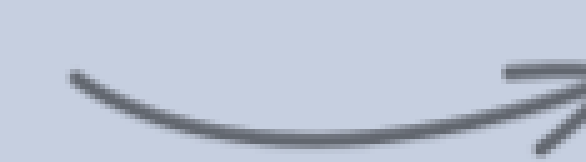


ARMY PUBLIC COLLEGE, SADDAR KARACHI

**JOIN
FOR
HSC-II Notes
MORE!!!**



2021-2022



What is the difference between human languages and computer languages?

Human languages

- Human stores data in brain
- Human communicates with words
- gestures and expressions
- human identify data in words
- human language has infinite words

Computer Language

- computer stores data in memory
- computer communication with ones and zeros
- basic data types of letters, numbers decimals, symbols
- computer identify data with keywords
- computer languages having finite keywords

What is the difference between a compiler and interpreter? Why each is used?

We generally write a computer program using a high-level language. A high-level language is one which is understandable by us humans. It contains words and phrases from the English (or other) language. But a computer does not understand high-level language. It only understands program written in 0's and 1's in binary, called the machine code. A program written in high level language is called a source code. We need to convert the source code into machine code and this is accomplished by compilers and interpreters. Hence, a compiler or an interpreter is a program that converts program written in high-level language into machine code understood by the computer.

The difference between an interpreter and a compiler is given below:

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.

Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming language like Python, Ruby use interpreters.	Programming language like C, C++ use compilers.

What is definition of a program?

In computing, a **program** is a specific set of ordered operations for a computer to perform. ... Typically, the **program** is put into a storage area accessible to the computer. The computer gets one instruction and performs it and then gets the next instruction.

What is definition of a programming?

Programming is the implementation of logic to facilitate specified computing operations and functionality. It occurs in one or more languages, which differ by application, domain and programming model. Programming language semantics and syntax are used when constructing an application. Thus, programming requires knowledge of application domains, algorithms and programming language expertise.

Explain the steps in a programming project?

Before going on to discuss the five steps of creating a program it is important to determine what exactly a program is. A program is a list of instructions that contain data for a computer to follow. Different programs are written with different languages.

Defining the Problem

Knowing the objective is the first consideration. Is it a payroll or editing program? Knowing who the end user will be is also important. Determining the inputs and outputs is next. How will the program operate and what data is needed to make it happen Finally, if the project is a go, then one must take measures to ensure the project is properly documented and analysed

Design the Program

Programs use *algorithms* which are like equations that tell the computer what task to perform. The aim of the programmer is to create algorithms that are clear and simple.

- Determine program logic through top down approach and modularization, using a hierarchy chart
- Design details using pseudocode and/or flowcharts, preferably involving control structure.



Code the Program

After the program has been designed it must be coded or written. Using the *pseudocode* and logic requirements from step two an appropriate programming language must be selected.

- Select the appropriate high-level programming language
- Code the program in that language following the syntax carefully

Test the Program

After the program is written it then enters the programming debugging and testing phase of the Program Development Life Cycle (PDLC). During this time the programmer will be looking for errors in both logic and syntax, as well as exploring other areas that may cause the program to either not work properly or to not run at all.

Document and Maintain

Documentation should be ongoing from the very beginning because it is needed for those involved with program now and future. Upon completion *User Documentation* for commercial use, *Operator Documentation* for people who run computer systems, and *Programmer Documentation* for programmers charged with maintenance.

Distinguish between syntax errors and logic error.

Program Errors

- Syntax Errors
 - Found during compilation
 - Usually caused by incorrect capitalization or spelling mistakes
- Run-Time Errors
 - Found during execution
 - Usually caused by invalid data such as dividing by zero
- Logic Errors
 - Found during testing
 - Usually caused by human error such as using the wrong equation, wrong strategy, etc.



What were the reason behind the development of high-level programming languages?

PREPARED BY SANA MAHMOOD 3

Chapter no. 01 2nd Year



The main advantage of high-level languages over low-level languages is that they are easier to **read**, write, and maintain. Ultimately, programs **written** in a high-level language must be translated into machine language by a compiler or interpreter. The first high-level programming languages were designed in the 1950s.

What is the difference between procedure-oriented languages and object-oriented languages?

Divided Into In POP, program is divided into small parts called functions.

Importance In POP, Importance is not given to **data** but to functions as well

Approach POP follows Top Down approach. OOP follows Bottom Up approach.

Access

Specifiers

POP does not have any access specifier. OOP has access specifiers named Public, Private, Protected, etc.

Data Moving In POP, Data can move freely from function to function in the system.

Expansion To add new data and function in POP is not so easy.

Data Access In POP, Most function uses Global data for sharing that can be accessed freely from function to function in the system.

Overloading In POP, Overloading is not possible. In OOP, overloading is possible in the form of

Function Overloading and Operator Overloading.

Examples Example of POP are: C, VB, FORTRAN, Pascal.

as sequence of actions to be done.

In OOP, program is divided into parts called objects.

In OOP, Importance is given to the data rather than procedures or functions because it works as a real world.

Data Hiding POP does not have any proper way for hiding data so it is **less secure**.

In OOP, objects can move and communicate with each other through member functions.

OOP provides an easy way to add new data and function.

In OOP, data can not move easily from function to function, it can be kept public or private so we can control the access of data.

OOP provides Data Hiding so provides more security.

Example of OOP are: C++, JAVA, VB.NET, C#.NET.

Define source code and object code? (2013) Or

What is the difference between source program and an object program? (2018)

Object code is the output of a compiler after it processes **source code**. **Source code** is the version of a computer program as it is originally written (i.e., typed into a computer) by a human in a programming language. A compiler is a specialized program that converts **source code** into **object code**.



Chapter No. 02: Past paper questions

What is an algorithm? (2013)

what is a programming algorithm? You can think of a **programming algorithm** as a recipe that describes the exact steps needed for the computer to solve a problem or reach a goal. We've all seen food recipes - they list the ingredients needed and a set of steps for how to make the described meal. Well, an algorithm is just like that. In computer lingo, the word for a recipe is a **procedure**, and the ingredients are called **inputs**. Your computer looks at your procedure, follows it to the letter, and you get to see the results, which are called **outputs**. A programming algorithm describes how to do something, and your computer will do it exactly that way every time. Well, it will once you convert your algorithm into a language it understands!.

Let's say that you have a friend arriving at the airport, and your friend needs to get from the airport to your house. Here are four different algorithms that you might give your friend for getting to your home:

The taxi algorithm:

1. Go to the taxi stand.
2. Get in a taxi.
3. Give the driver my address.

The rent-a-car algorithm:

1. Take the shuttle to the rental car place.
2. Rent a car.
3. Follow the directions to get to my house.

The bus algorithm:

1. Outside baggage claim, catch bus number 70.
2. Transfer to bus 14 on Main Street.
3. Get off on Elm street.
4. Walk two blocks north to my house.



Chapter no. 02 Past paper questions with answers 2nd year

All of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way. Each algorithm also has a different cost and a different travel time.

Write different algorithm standard notations. (2013)

Standard notations of an algorithm refers to defining the mathematical framing of its run-time performance. Using asymptotic analysis, we can very well conclude the best case, average case, and worst case scenario of an algorithm.

- **Best Case** – Minimum time required for program execution.
- **Average Case** – Average time required for program execution.
- **Worst Case** – Maximum time required for program execution.

Big Oh Notation, O

The notation $O(n)$ is the formal way to express the upper bound of an algorithm's running time. It measures the worst case time complexity or the longest amount of time an algorithm can possibly take to complete.

Omega Notation, Ω

The notation $\Omega(n)$ is the formal way to express the lower bound of an algorithm's running time. It measures the best case time complexity or the best amount of time an algorithm can possibly take to complete.

Theta Notation, θ

The notation $\theta(n)$ is the formal way to express both the lower bound and the upper bound of an algorithm's running time. It is represented as follows

What is flowchart? Describe different symbols used while drawing a flowchart. (2012)

A flowchart is a type of diagram that represents an algorithm, workflow or process. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem.



Chapter no. 02 Past paper questions with answers 2nd year

S.No.	Name	Symbols	Meanings
i.	Start/Stop	Oval	It indicates the beginning and ending of the flowchart.
ii.	Input/output	Parallelogram	It indicates the input/output operations.
iii.	Processing	Rectangle	It indicates the calculation or manipulate of data.
iv.	Decision	Diamond	It indicates the decision making and branching.
v.	Flow lines		It indicates the direction of flow of instruction.
vi.	Connector	Circle	It joins one part of the flowchart with another part.

What is an algorithm? What is its importance in computer programming? (2012)

Algorithms are very important in computer Science. The best-chosen algorithm makes sure computer will do the given task at best possible manner. In cases where efficiency matters a proper algorithm is really vital to be used. An algorithm is important in optimizing a computer program according to the available resources.

Ultimately when anyone decides to solve a problem through better algorithms then searching for the best combination of program speed and least amount of memory consumption is desired.



What is meant by control structure? Discuss selection control structure? (2017)

Control structures:

A **control structure** is a block of programming that analyses variables and chooses a direction in which to go based on given parameters. The term flow **control** details the

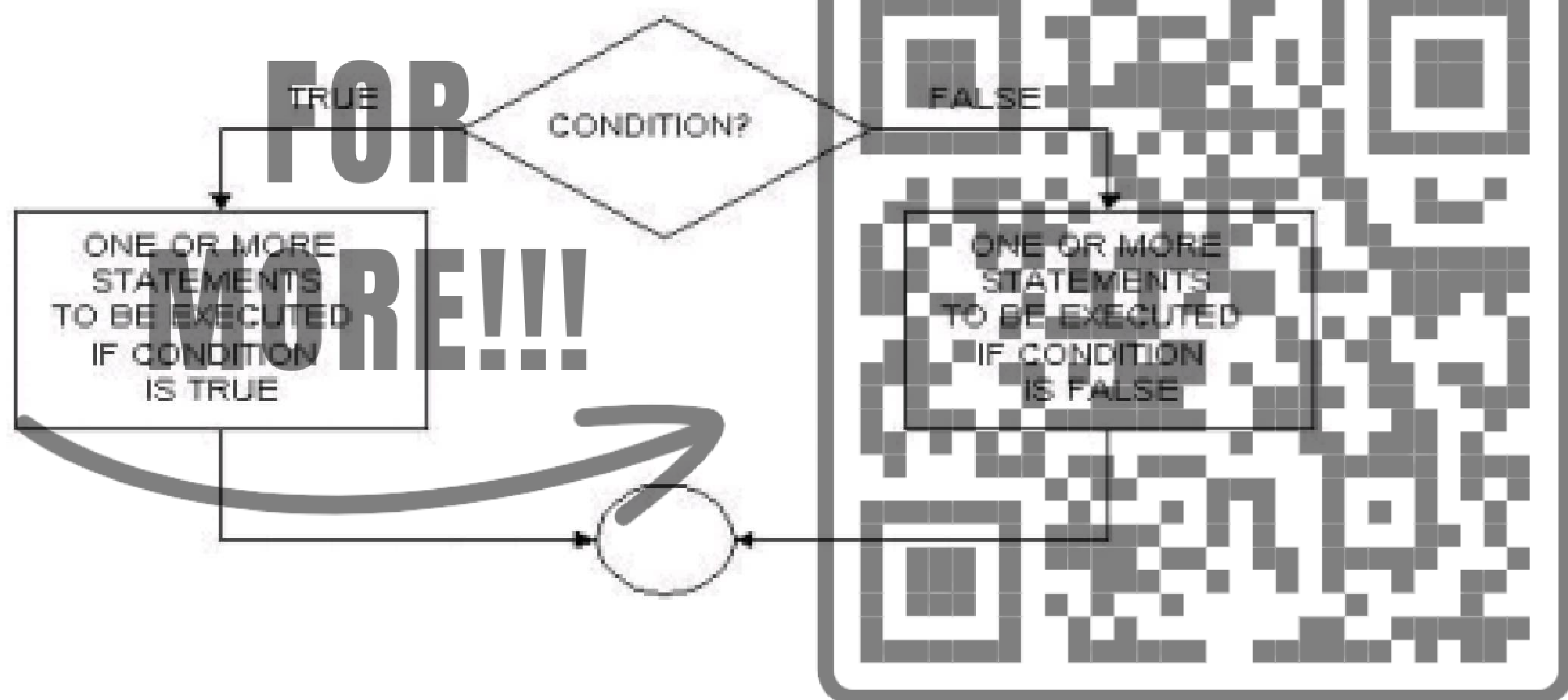
direction the program takes (which way program **control** "flows").How the programs executes.

The Selection Control Structure

The selection control structure allows one set of statements to be executed if a condition is true and another set of actions to be executed if a condition is false. A selection structure, also called an "If-Then-Else" structure. The selection control structure employs a number of condition which lead to a selection of one out of several modules.

PREPARED BY SANA MAHMOOD 3

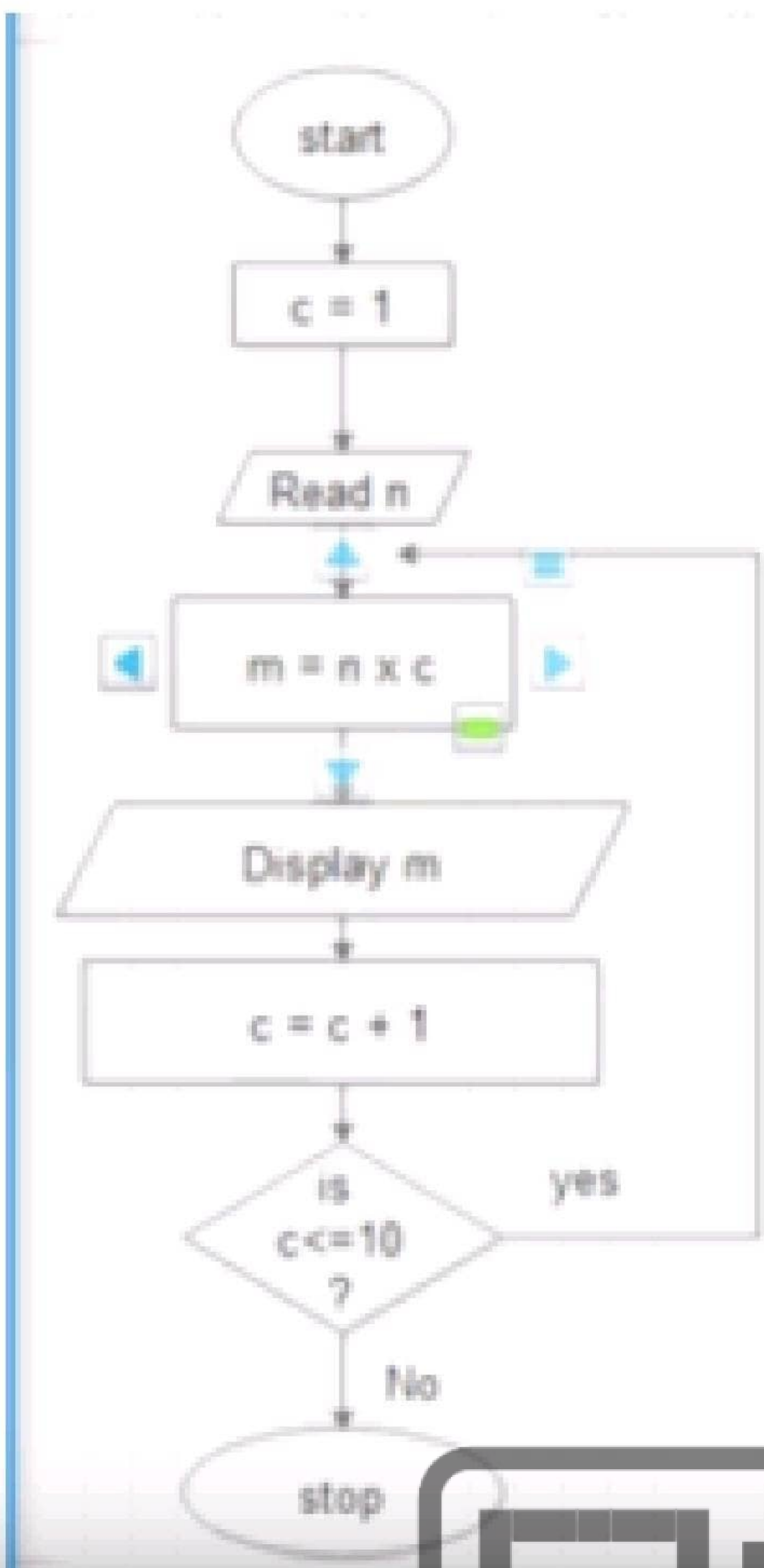
Chapter no. 02 Past paper questions with answers 2nd year



Draw a flowchart that prints the table of any inputted number. (2016)



n	x	c	=	m
5	x	1	=	5
5	x	2	=	10
5	x	3	=	15
5	x	4	=	20
5	x	5	=	25
5	x	6	=	30
5	x	7	=	35
5	x	8	=	40
5	x	9	=	45
5	x	10	=	50



**JOIN
FOR
MORE!!!**



PREPARED BY SANA MAHMOOD 4

Chapter no. 02 Past paper questions with answers 2nd year

What is a sequence control structure?

Condition not in use, only one step is performed at once, none is repeated.



Name the control structure used in c language and discuss iterative control structure with syntax and example. (2015)

There are four types of control statements in C:

- Decision making statements.
- Selection statements.
- Iteration statements.
- Sequence statements.

Iteration

The iterative control structures are used for repetitively executing a block of code multiple times; the following are the basic iterative control structures.

•❖ **for:**

A for loop execute the statements for a specific number of times mentioned in the condition. Syntax:

```
for(initialization; condition; iteration)
{
    Statements;
}
```

•❖ **while:**

A while loop executes statement repeatedly until the condition satisfies.

Syntax:

```
While(condition)
{
    Statements;
}
```

•❖ **do-while:**

A do-while loop works similar to while loop, the only difference is it executes the statement once before checking the condition.

Syntax:

PREPARED BY SANA MAHMOOD 5

Chapter no. 02 Past paper questions with answers 2nd year




```

Do
{
    Statements;
}
While (condition);

```

What repetitive control structures used in C language. Discuss for loop with example (2011)

Iterative control instruction is also known as repetitive control instruction or loop. Sometimes it is desirable to executed same statement again and again. This can be done with the help of loops

For loop:

In computer science, a **for-loop** (or simply **for loop**) is a control flow statement for specifying iteration, which allows code to be executed repeatedly.

```

for (initialization; condition test; increment or decrement)
{
    //Statements to be executed repeatedly
}

```



PREPARED BY SANA MAHMOOD 6



**JOIN
FOR
MORE!!!**



JOIN
FOR
MORE!!!



Chapter No.03 Past Paper questions with answers 2nd Year

Chapter no. 03: Past paper questions with answers

What is the purpose of the statement `#include<stdio.h>` in a C program? What is a statement terminator?

#include<stdio.h>:

The lines that begin with a # are called pre-processing directives. The #include directive tells the computer to use the information in the header file called stdio.h. The initial stdio stand for standard input/output and h stand for header file. The stdio.h file contains all the instructions the compiler needs to work with disk files and send information to the screen. This header file is provided by the Turbo C++ compiler. The angle brackets around <stdio.h> indicate that this file to be found in the “usual place”, which is system independent. Another example of header file is <conio.h> in which conio stand for console input/output.

Statement terminator:

Every statement must end with semi-colon (also known as statement terminator). You can put several statements on a line. For clarity's sake usually one statement is put on a line.

What are escape sequences? Give an example

You can control the way the cursor moves on your monitor, and some other computer functions, by using special codes, called escape sequences. An escape sequence consists of the escape character (the backslash '\') and a second character. The backslash causes an escape from the normal interpretation of a string, so that the next character is recognised as one having a special meaning.

Example:

The escape sequence \n performs a new line command and is most commonly used at the end of an instructions. It can appear anywhere inside the double quotes and moves the cursor to the start of the next line. These commands

```
printf("this is\n");  
printf("my first\n");  
printf("C program");
```



Chapter No.03 Past Paper questions with answers 2nd Year

displays three lines of text:

this is

my first

C program”

What is the basic structure of C program? (2014)

Basic Structure of C Program

1. Documentation Section
2. Link Section
3. Definition Section
4. Global Declaration Section
5. main() Function Section
6. Subprogram Section

Documentation Section

This section consists of comment lines which include the name of the program, the name of the programmer, the author and other details like time and date of writing the program.

Documentation section helps anyone to get an overview of the program. /*Documentation Section:

Program Name: Program to find the area of circle

Author: Rumman Ansari

Date : 12/01/2013

Time : 10 AM

*/

Link Section

The link section consists of the header files of the functions that are used in the program. It provides instructions to the compiler to link functions from the system library such as using the #include directive.

#include"stdio.h" //link section




```
#include"conio.h" //link section/
```

PREPARED BY SANA MAHMOOD 2

Chapter No.03 Past Paper questions with answers 2nd Year

Definition Section

All the symbolic constants are written in the definition section. Macros are known as symbolic constants.

```
#define PI 3.14 //definition section
```

Every C program must have a `main()` function which is the starting point of the program execution.

Global Declaration Section

The global variables that can be used anywhere in the program are declared in the global declaration section. This section also declares the user defined functions.

```
float area; //global declaration section
```

Subprogram Section

The subprogram section contains all the user defined functions that are used to perform a specific task. These user defined functions are called in the `main()` function. **void** message()

```
{  
printf("This Sub Function \n");  
printf("we can take more Sub Function \n");  
}
```

What are escape sequences? Mention all escape sequences. (2012-13) You can control the way the cursor moves on your monitor, and some other computer functions, by using special codes, called escape sequences. An escape sequence consists of the escape character (the



backslash '\') and a second character. The backslash causes an escape from the normal interpretation of a string, so that the next character is recognised as one having a special meaning.

PREPARED BY SANA MAHMOOD 3

Chapter No.03 Past Paper questions with answers 2nd Year

See book page no 66. Table 3.1 (mention all escape sequences)

Define IDE and discuss how it is useful in C programming. (2011)

An **IDE** typically contains a code editor, a compiler or interpreter, and a debugger, accessed through a single graphical user interface (GUI). The user writes and edits source code in the code editor. The compiler translates the source code into a readable language that is executable for a computer.

Benefits of using IDEs

An IDE can improve the productivity of software developers thanks to fast setup and standardization across tools.

Without an IDE, developers spend time deciding what tools to use for various tasks, configuring the tools and learning how to use them. Many or even all of the necessary dev-test tools are included in one integrated development environment.

IDEs are also designed with all their tools under one user interface. An IDE can standardize the development process by organizing the necessary features for software development in the UI.

What is meant by C is a Case sensitive language?

C language is a case sensitive language. Commands and functions names are always written in lowercase characters, so we use `printf()` rather than `PRINTF()`. Uppercase is reserved for things called constants and symbolic constant.

What is pre-processor directives? Explain any two pre-processor directives in C program with example.

Pre-processor directives are lines included in a program that begin with the character `#`, which make them different from a typical source code text. They are invoked by the compiler to process some programs before compilation. Pre-processor directives change the text of the source code and the result is a new source code without these directives.



Chapter No.03 Past Paper questions with answers 2nd Year

`#include`

The `#include` pre-processor directive is used to paste code of given file into current file. It is used include system-defined and user-defined header files. If included file is not found, compiler renders error.

A macro is a segment of code which is replaced by the value of macro. Macro is defined by `#define` directive.

Syntax

`#define` token value

**JOIN
FOR
MORE!!!**



**JOIN
FOR
MORE!!!**



PREPARED BY SANA MAHMOOD 5



Chapter no.04 Past paper questions with answers 2nd Year

Chapter no. 04: Past paper questions with answers

Explain the data types used in C language with examples. How much space is reserved for each data type in memory? (2014-13-12)

Or

Explain the data types used in C language with examples. (2011)

In computer science and computer programming, a data type or simply type is a classification of data which tells the compiler or interpreter how the programmer intends to use the data.

JOIN FOR MORE!!

Basic Data Type

Data Type	Bytes Occupied in RAM
char	1 byte
int	2 bytes
float	4 bytes
double	8 bytes



Additional Data Types

Data Type	Byte Occupied in RAM
unsigned char	1 byte
short int	1 byte
unsigned short int	1 byte
unsigned int	2 bytes
long int	4 bytes
unsigned long int	4 bytes
long double	10 bytes

1

Chapter no.04 Past paper questions with answers 2nd Year

Example of valid integers are:

4 -87 +9345 -18239

Examples of valid float variables

-78.294 395.20587 -95.0 4.39

Examples of char are

‘m’ ‘>’ ‘H’ ‘2’ [‘ ‘!’ ‘F’

Examples of Strings are

“school-786 Abc”



What are format specifiers and how do we use them in C program? (2013)

Format specifiers can be defined as the operators which are used in association with printf() function for printing the data that is referred by any object or any variable. When a value is stored in a particular variable, then you cannot print the value stored in the variable straightforwardly without using the format specifiers. You can retrieve the data that are stored in the variables and can print them onto the console screen by implementing these format specifiers in a printf() function.



%d	Integer Format Specifier

%f	Float Format Specifier
%c	Character Format Specifier
%s	String Format Specifier
%u	Unsigned Integer Format Specifier
%ld	Long Int Format Specifier

2

Chapter no.04 Past paper questions with answers 2nd Year

Integer Format Specifier

%
d

The format specifier is implemented for representing integer values. This is used with

printf() function for printing the integer value stored in the variable.

Float Format Specifier

%
f

The format specifier is implemented for representing fractional values. This is implemented

within printf() function for printing the fractional or floating value stored in the variable. Whenever you need to print any fractional or floating data, you have to use format specifier.

%
f

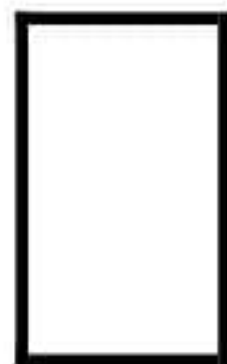
Character Format Specifier

%

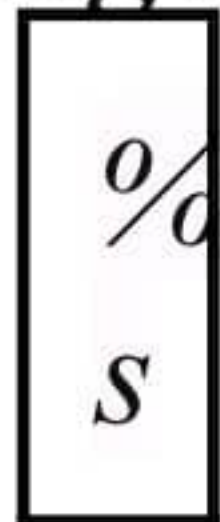
The %c format specifier is implemented for representing characters. This is used with printf()



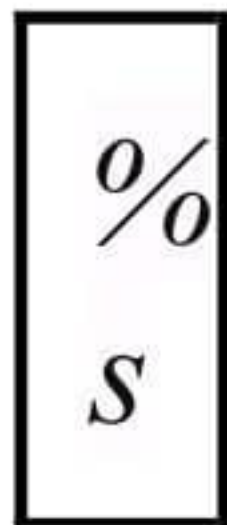
function for printing the character stored in a variable. When you want to print a character data, you should incorporate the %c format specifier.



String Format Specifier



The format specifier is implemented for representing strings. This is used in printf() function for printing a string stored in the character array variable. When you have to print a string, you should implement the format specifier.



3

**JOIN
FOR
MORE!!!**



Character set

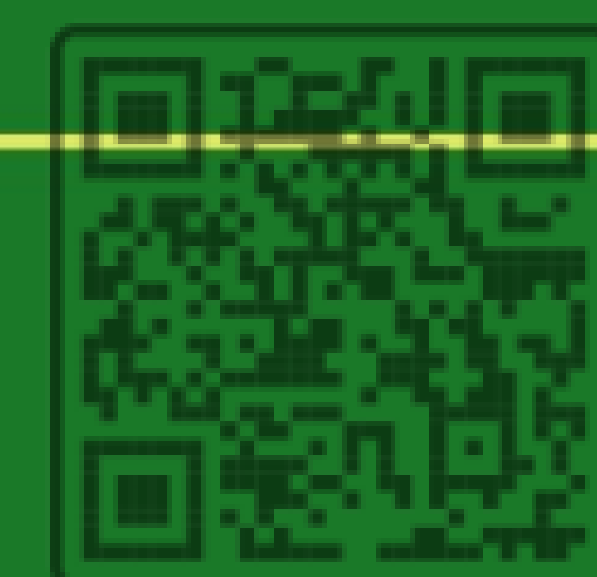
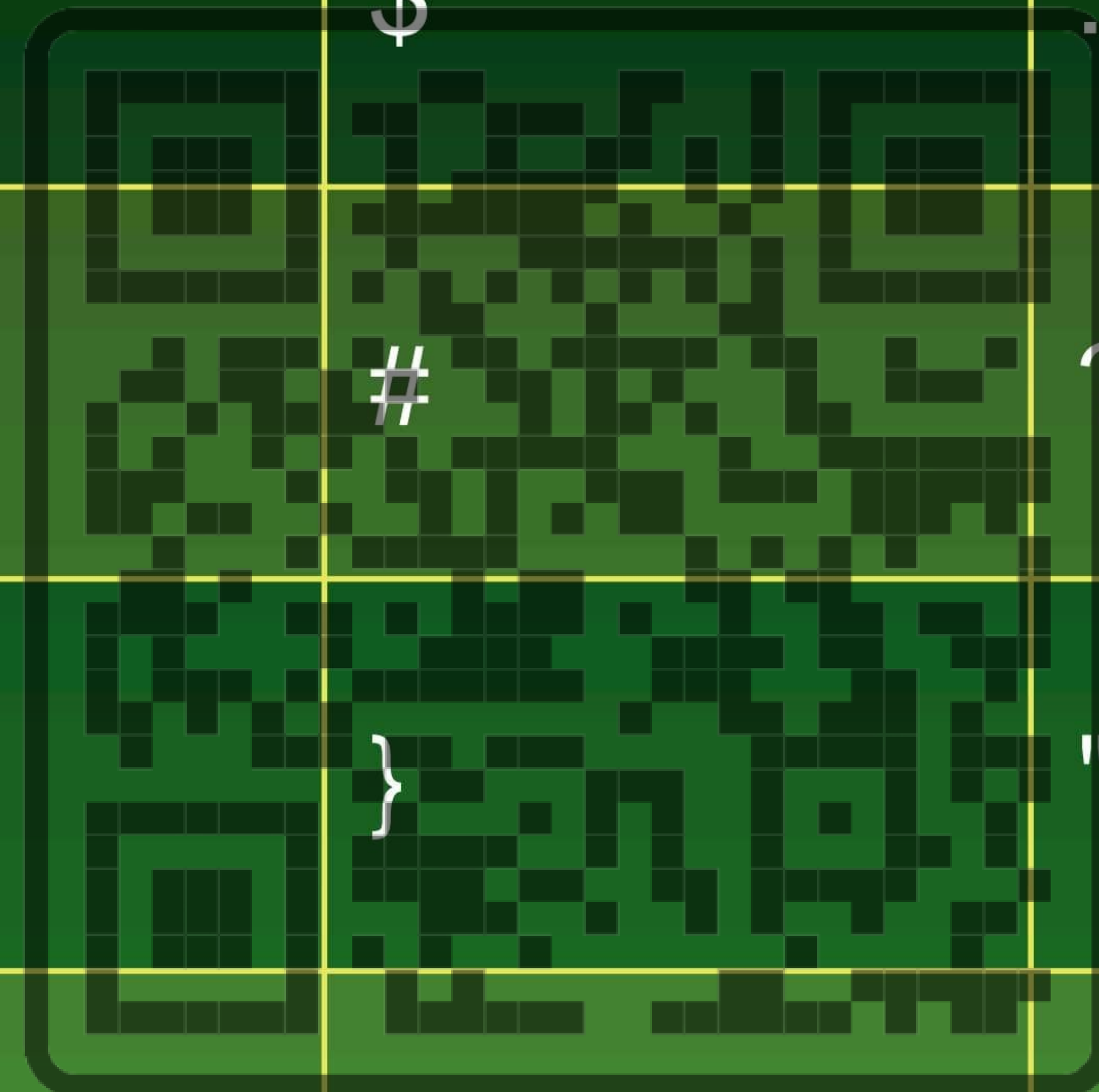
- Character set is a set of alphabets, letters and some special characters that are valid in C language.
- **Alphabets**
- Uppercase: A B C X Y Z
- Lowercase: a b c x y z
- C accepts both lowercase and uppercase alphabets as variables and functions.
- **Digits**
- 0 1 2 3 4 5 6 7 8 9
- **White space Characters**
- blank space, new line, horizontal tab, carriage return



Special Characters in C Programming

,	<	>	.	_
(	)	;	\$	:
%	[	]	#	?
'	&	{	}	"
^	!	*	/	
-	\	~	+	

**JOIN
FOR
MORE!!!**



What is Identifiers

JOIN

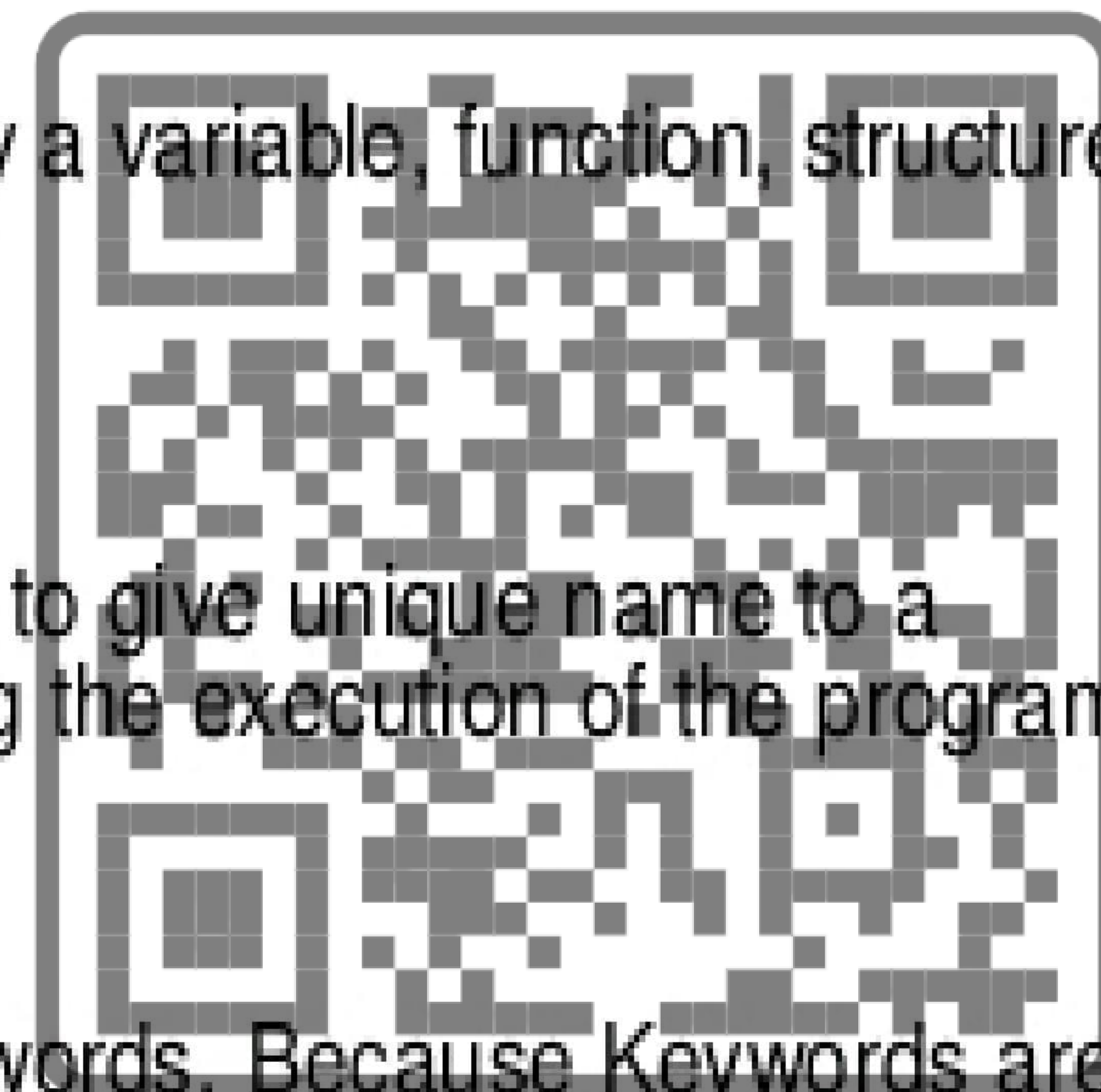
FOR

MORE!!!

A C **identifier** refers to name used to identify a variable, function, structures or any other user-defined item or entity.

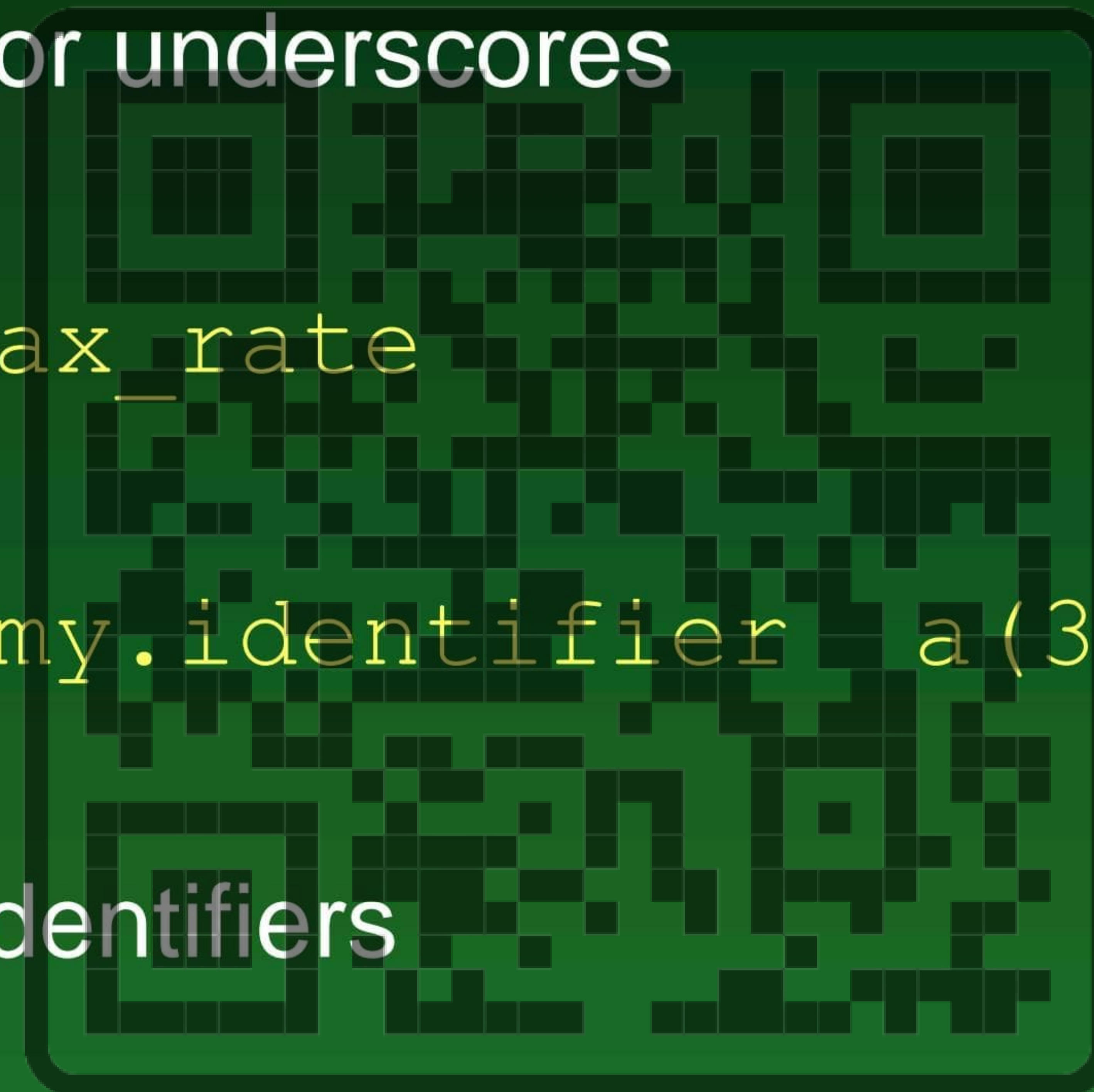
Identifier must be unique. They are created to give unique name to a entity/item/variable name to identify it during the execution of the program

identifier names must be different from keywords. Because Keywords are predefined, reserved words used in programming .As for **Example** :You cannot use **double** as an identifier because **double** is a keyword.

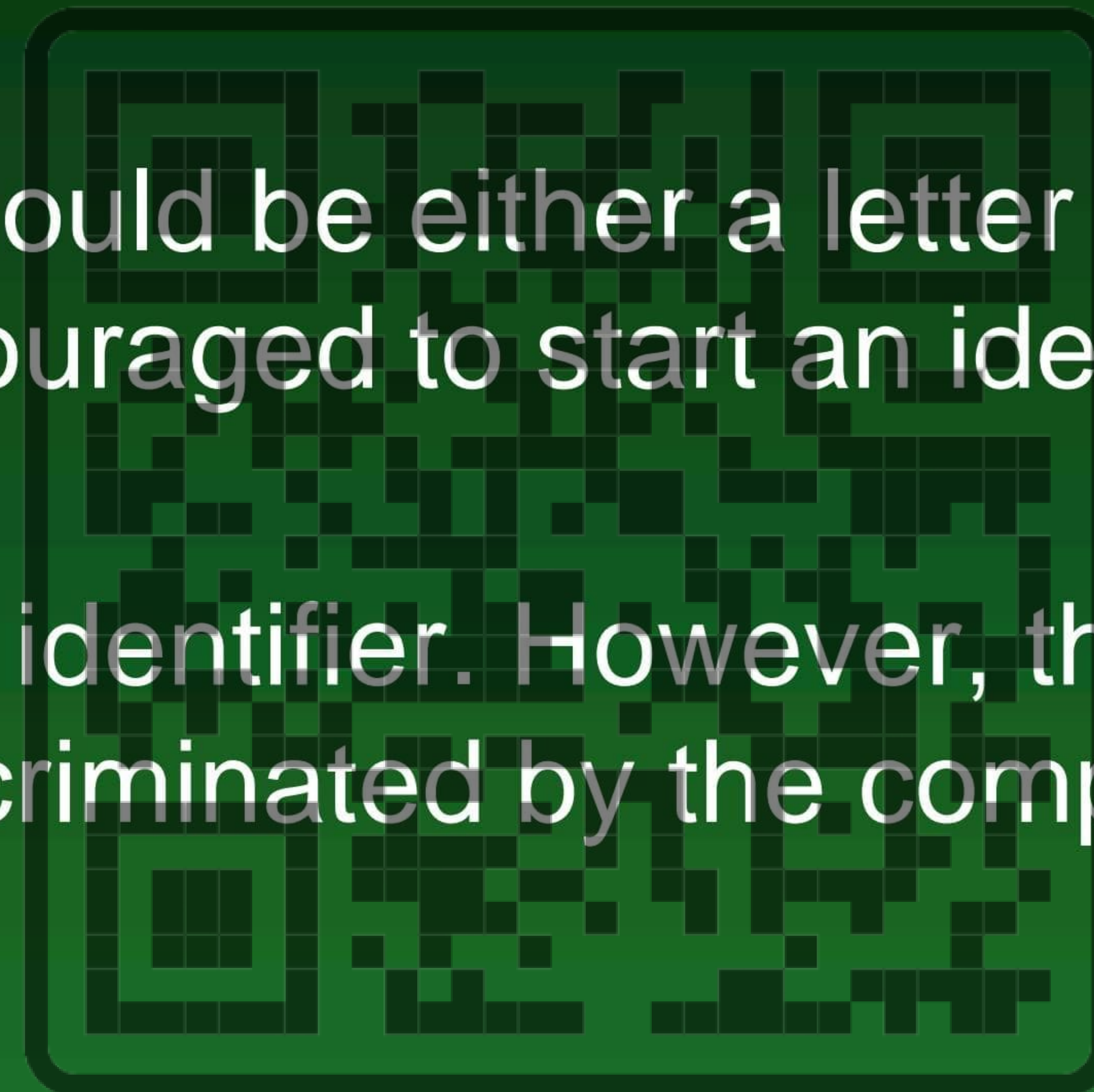


Identifiers

- *identifier* – name of a variable or any other named construct
- identifier must start with a letter or underscore symbol (), the rest of the characters should be letters, digits or underscores
- legal identifiers:
x x1 x_1 _abc sum tax_rate
- illegal identifiers, why?
13 3X %change data-1 my.identifier a(3)
- C++ is *case sensitive*:
MyVar and myvar are different identifiers



- **Rules for writing an identifier**
- A valid identifier can have letters (both uppercase and lowercase letters), digits and underscores.
- The first letter of an identifier should be either a letter or an underscore. However, it is discouraged to start an identifier name with an underscore.
- There is no rule on length of an identifier. However, the first 31 characters of identifiers are discriminated by the compiler.



Keywords

- **C Keywords.** **Keywords** are predefined, reserved words used in programming that have special meanings to the compiler. **Keywords** are part of the syntax and they cannot be used as an identifier.

keywords are identifiers reserved as part of the language

`int, return, float, double,
long, switch, break, return, char, if, for etc`

- cannot be used by the programmer to name constructs
- consist of lowercase letters only
- have special meaning to the compile



Keywords (cont.)

asm	do	if	return	typedef
auto	double	inline	short	typeid
bool	dynamic_cast	int	signed	typename
break	delete	long	sizeof	union
case	else	mutable	static	unsigned
catch	enum	namespace	static_cast	using
char	explicit	new	struct	virtual
class	extern	operator	switch	void
const	false	private	template	volatile
const_cast	float	protected	this	wchar_t
continue	for	public	throw	while
default	friend	register	true	union
delete	goto	reinterpret_cast	try	unsigned

JOIN
FOR
MORE!!!



Data type

- In computer science and computer programming, a data type or simply type is a classification of data which tells the compiler or interpreter how the programmer intends to use the data.

MORE!!

Basic Data Type

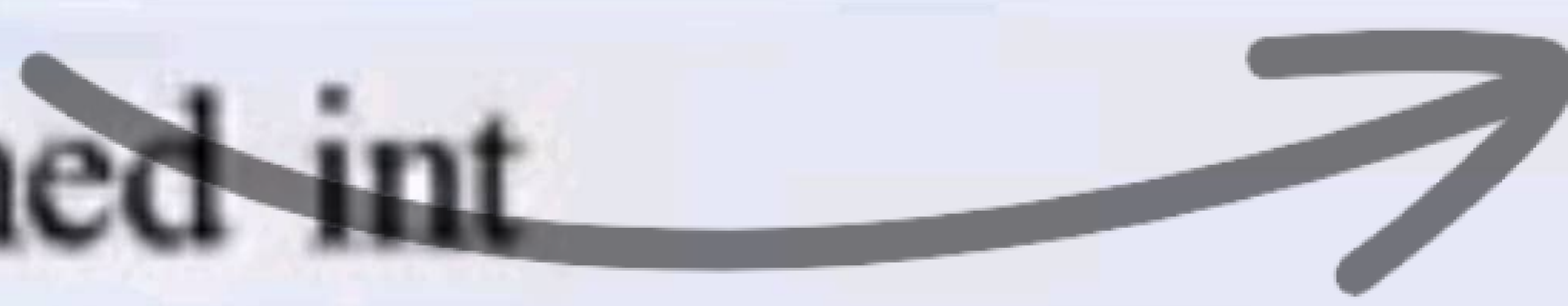
Data Type	Bytes Occupied in RAM
char	1 byte
int	2 bytes
float	4 bytes
double	8 bytes



Additional Data Types

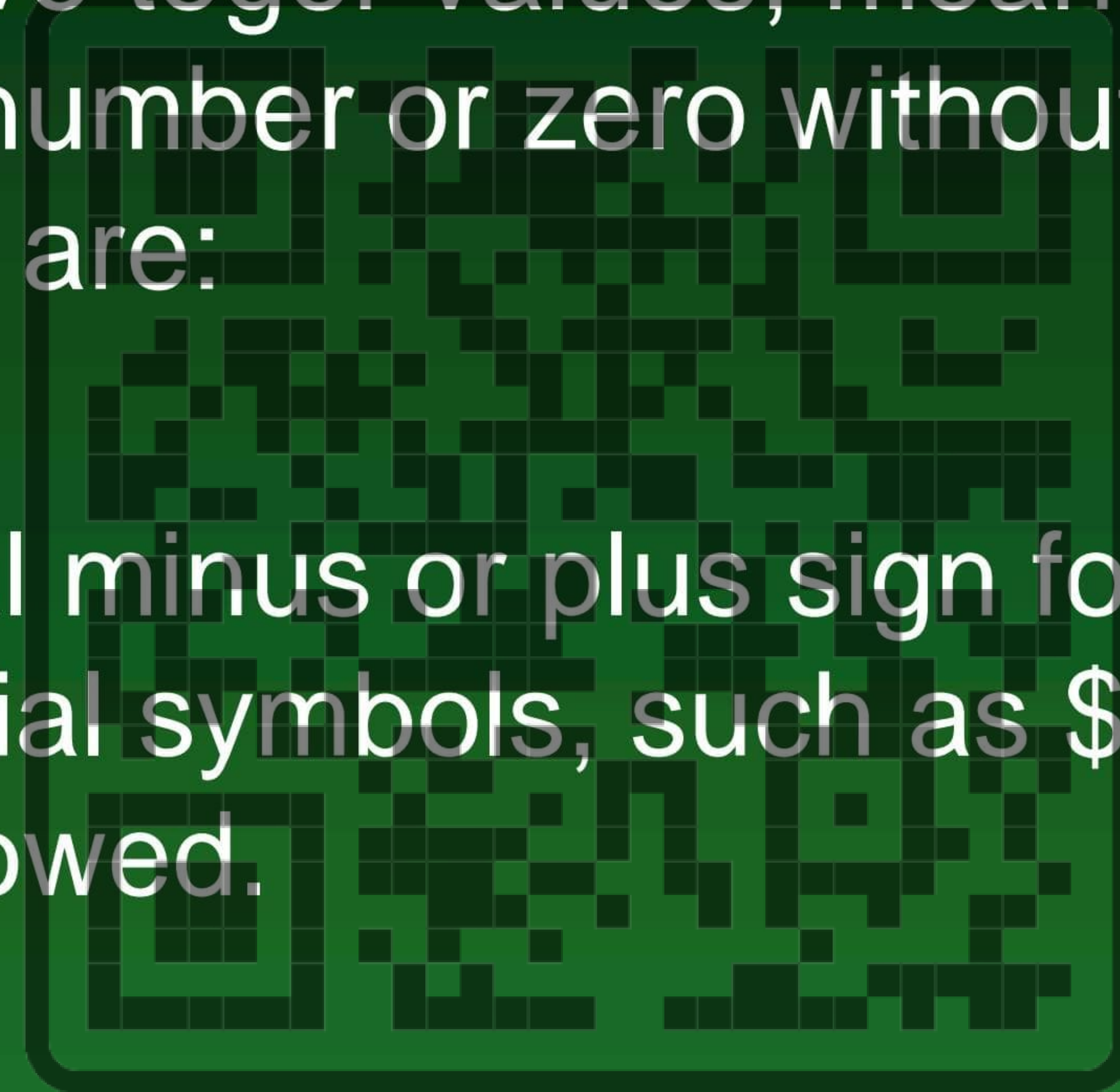
Data Type	Byte Occupied in RAM
unsigned char	1 byte
short int	2 bytes
unsigned short int	2 bytes
unsigned int	4 bytes
long int	4 bytes
unsigned long int	4 bytes
long double	10 bytes

JOIN
FOR
MORE!!!

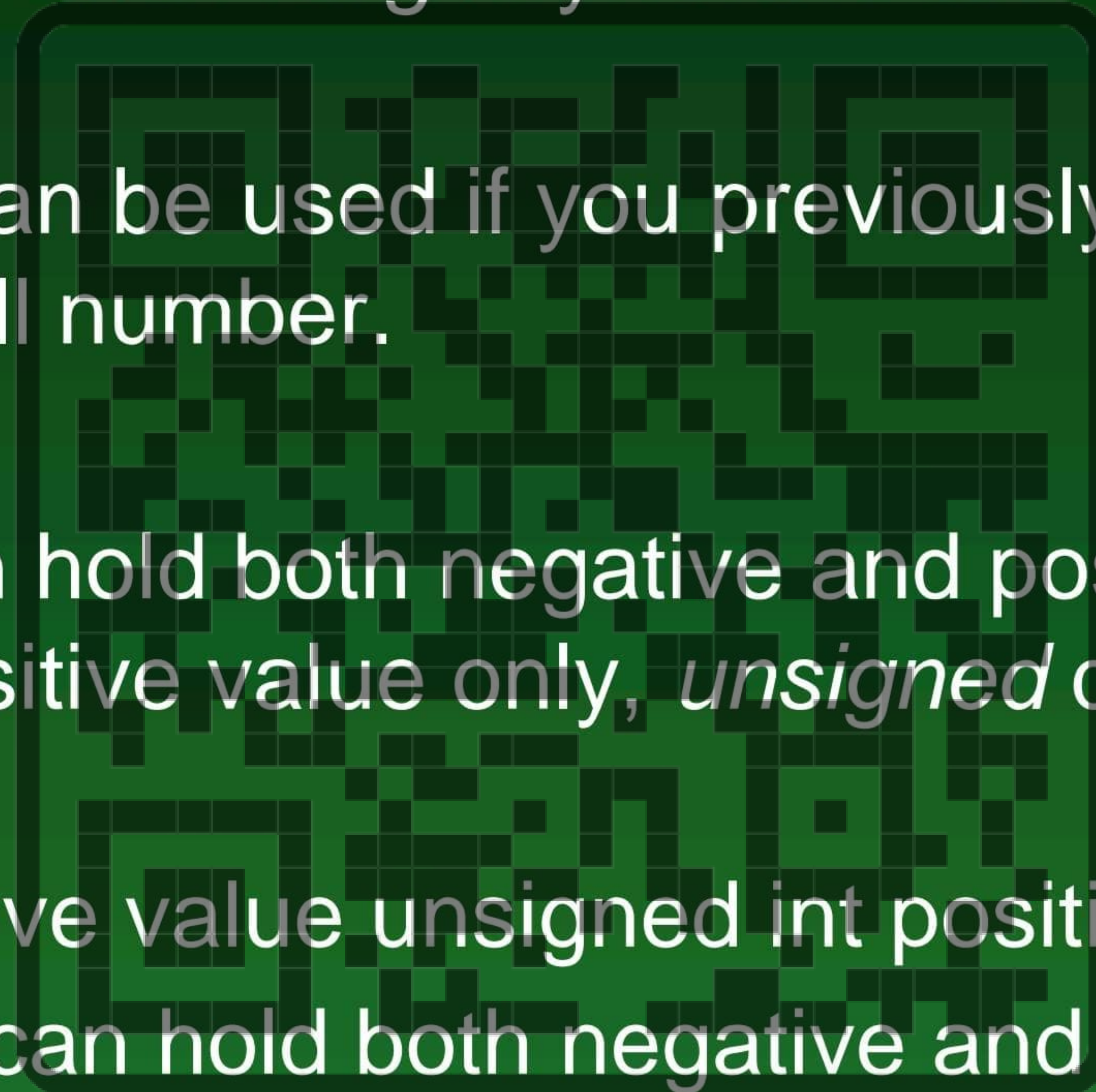


Int Data Type

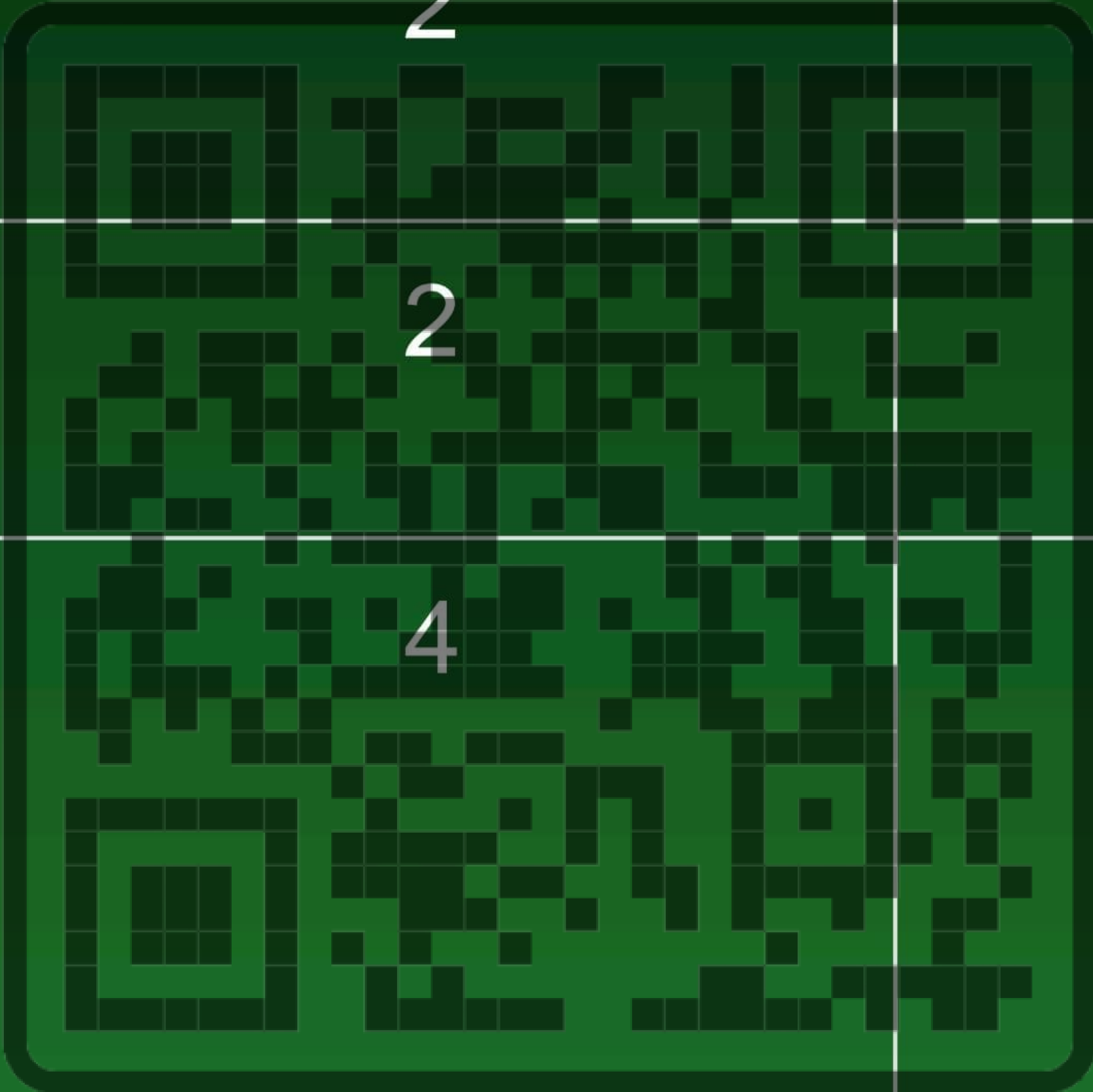
- The int data type represents integer values, meaning any positive and negative values, number or zero without decimal point. Example of valid integers are:
- 4 -87 +9345 -18239
- An integer consists of an optional minus or plus sign followed by a series of digits. Commas, special symbols, such as \$, or e for exponential notation are not allowed.



- **C Qualifiers**
- Qualifiers alters the meaning of base data types to yield a new data type.
- **Size qualifiers**
- Size qualifiers alters the size of a basic type. There are two size qualifiers, *long* and *short*. For example:
 - long double i;
 - The size of *double* is 8 bytes. However, when *long* keyword is used, that variable becomes 10 bytes.
 - There is another keyword *short* which can be used if you previously know the value of a variable will always be a small number.
- **Sign qualifiers**
- Integers and floating point variables can hold both negative and positive values. However, if a variable needs to hold positive value only, *unsigned* data types are used. For example:
 - // unsigned variables cannot hold negative value unsigned int positiveInteger;
 - There is another qualifier *signed* which can hold both negative and positive only. However, it is not necessary to define variable *signed* since a variable is signed by default.
 - An integer variable of 4 bytes can hold data from -2^{31} to $2^{31}-1$. However, if the variable is defined as unsigned, it can hold data from 0 to $2^{32}-1$.
 - It is important to note that, sign qualifiers can be applied to *int* and *char* types only.

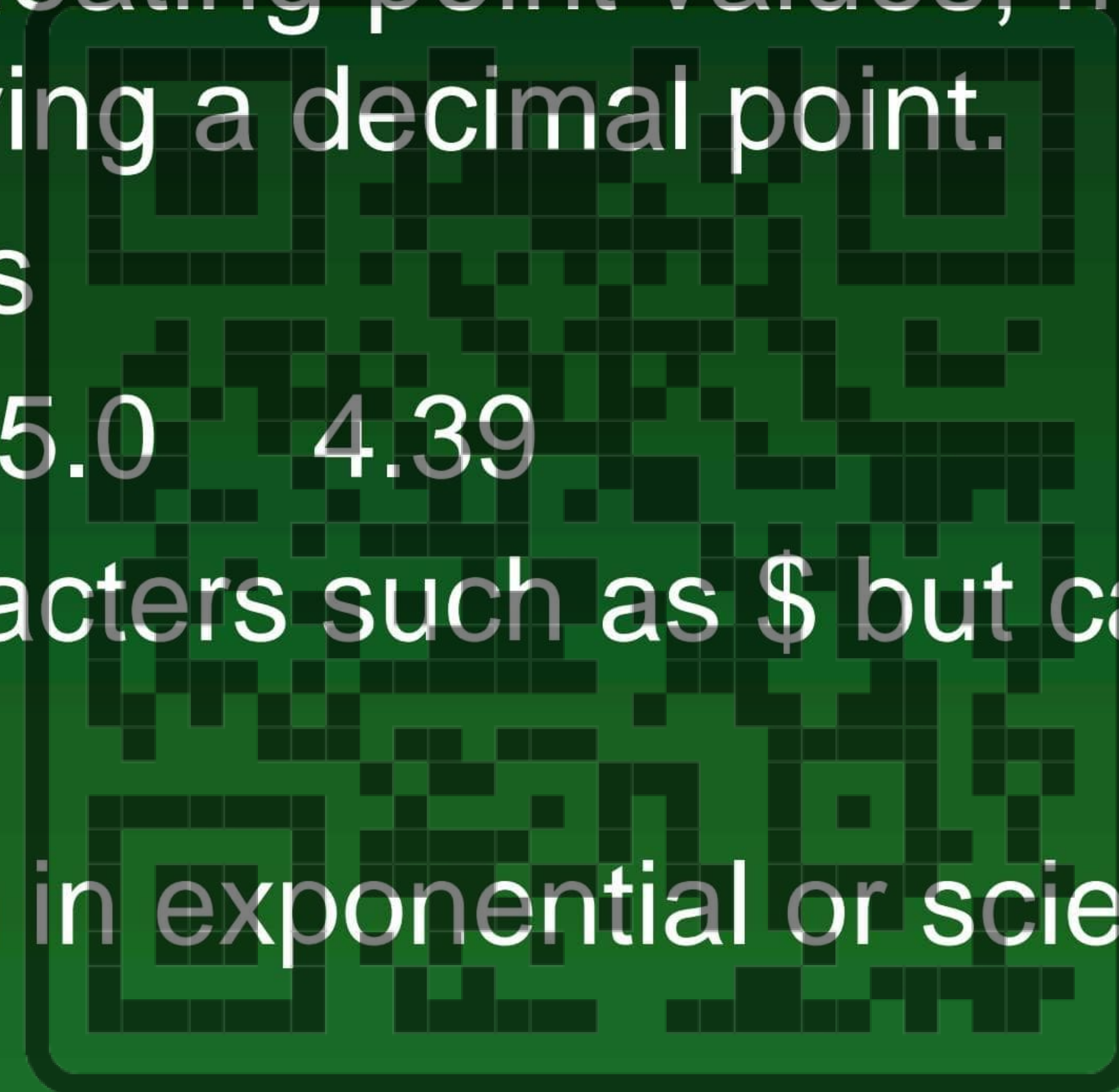


Data Type	Range	Bytes width	Format
Short signed int	-32768 to +32767	2	%d
Short unsigned int	0 to 65535	2	%u
Long signed int	-2,147,483,648 to 2,147,483,647	4	ld%
Long unsigned int	0 to 4,294,967,295	4	%lu



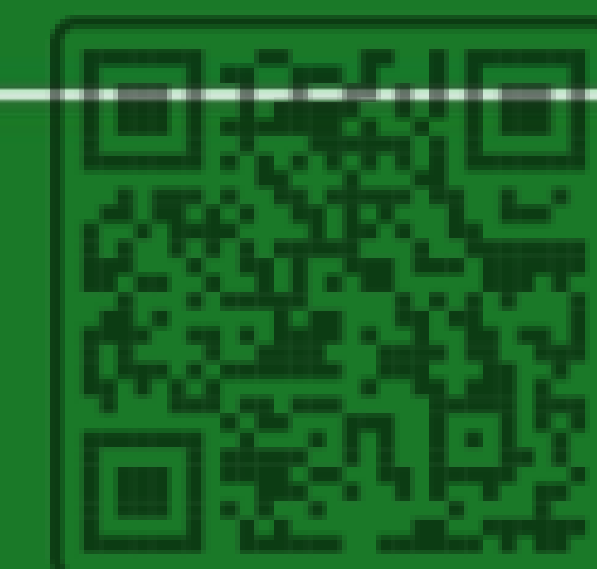
Float Data Typed

- The float data type represents floating-point values, meaning any signed or unsigned number having a decimal point.
- Examples of valid float variables
- -78.294 395.20587 -95.0 4.39
- It cannot contain special characters such as \$ but can include letter e for exponential.
- Float and double can be written in exponential or scientific notation, if needed.
- See book page 76



Floating data type

Data Type	Range	Bytes width	Format
float	4bytes	$3.4e - 38$ to $3.4e + 38$ (7 digits)	%f
double	8bytes	$1.7e - 308$ to $1.7e + 308$ (15 digits)	%lf
long double	8bytes	$3.4e - 4932$ to $1.1e + 4932$	%Lf



Char Data Type

- C also recognizes nonnumeric character data. A char represents individual characters. Hence, the char type will generally require only one byte of memory. A char is any valid ASCII character enclosed in single character quotes. Examples of char are
- 'm' '>' 'H' '2' '[' '!' 'F'
- Note that a char data item can also be a single numeric digit, such as 1,2,3. but C makes a distinction between the character '1' and the number 1. as character '1' cannot be used in a mathematical operation because it does not have a numeric value.
- See in book, page 77



Data type	Bytes width	Range	Format
char	1byte	-128 to 127 or 0 to 255	%c
unsigned char	1byte	0 to 255	%c
signed char	1byte	-128 to 127	%c

See example 4.3
Page 77



Constants

**JOIN
FOR
MORE!!!**

1. Integer constant
2. Real or Floating Point Constant
3. Character Constant
4. String Constant



Numeric Constants



Constant

- **C - Constants and Literals.** **Constants** refer to fixed values that the **program** may not alter during its execution. These fixed values are also called literals. **Constants** can be of any of the basic data types like an integer **constant**, a floating **constant**, a character **constant**, or a string literal.

Integer Constant

MORE!!!

An integer constant is signed or unsigned whole number.

Ex.: 25, -56, +30 etc



- See book page 79



Real or Floating Point Constant

An signed or unsigned number with fractional part is called real or floating point constant.

Ex: 0.25, -2.56, 0.23, 0.34e3 etc
where e = exponential

Character Constant

Any letter or character enclosed in single apostrophe is called, single character string constant or character constant.

Ex: 'y' '+' '\$' etc



String Constant

Any string of characters consisting of letter, digits and symbols enclosed in double quotes is called, string of character constant or string constant.

Ex: "Total amount is" "Average =" "Jail No. – 420"

JOIN
FOR
MORE!!!



See book page 81



Symbolic Constant

**JOIN
FOR
MORE!!!**



```
1  /* Symbolic Constants */
2  #include<stdio.h>
3  #define two 3
4  int main()
5  {
6      int a, b, c, d;
7      a = 1;
8      c = 4;
9      b = two + a; //3+1
10     d = two + c; //3+4
11     printf("%d\n", b);
12     printf("%d\n", d);
13 }
```




```
#include<stdio.h>
```

```
#define two 3
```

```
int main()
```

```
{
```

```
int a, b, c, d;
```

```
a = 1;
```

```
c = 4;
```

```
b = two + a; // 3+1
```

```
d = two + c; // 3+4
```

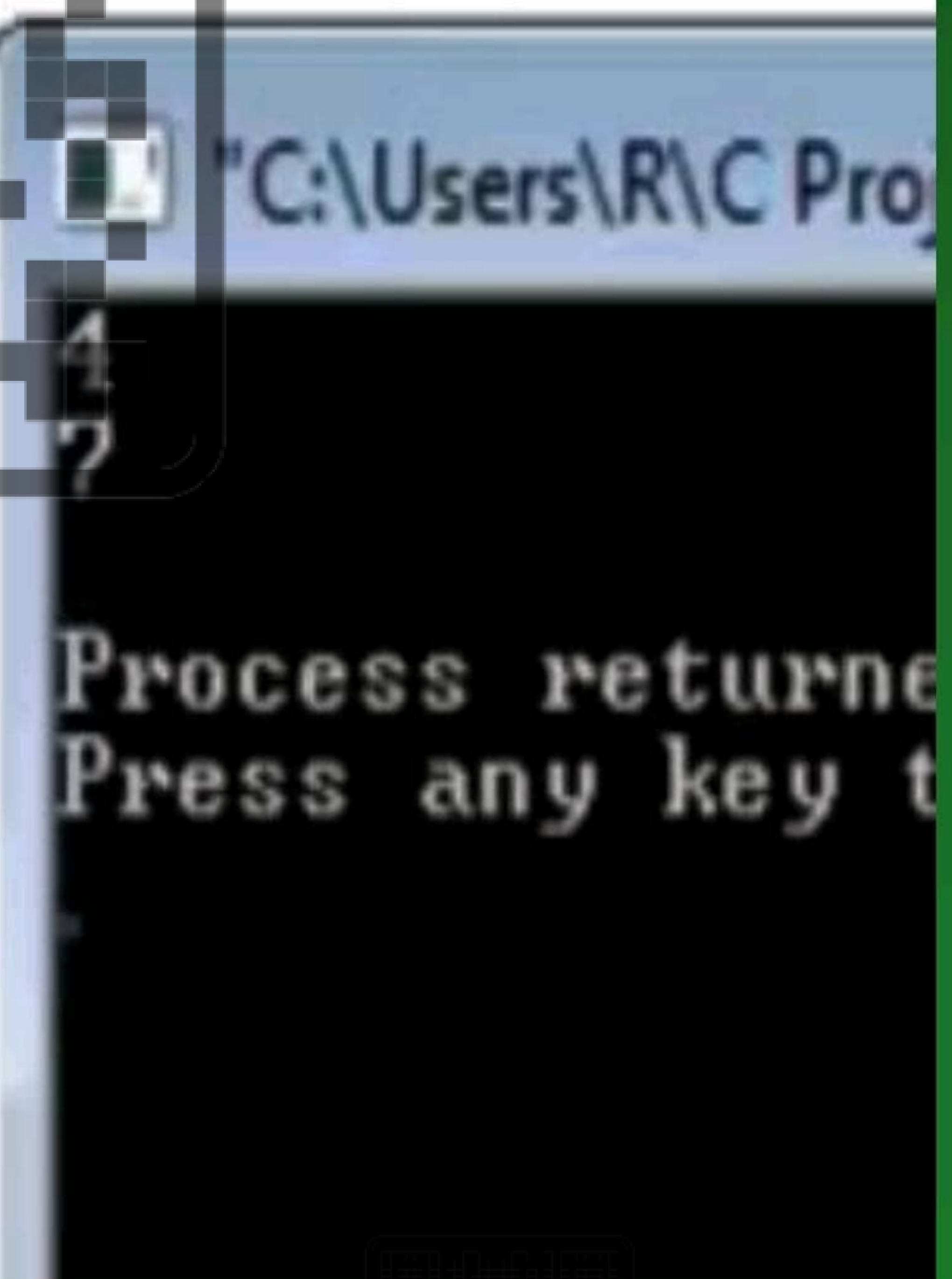
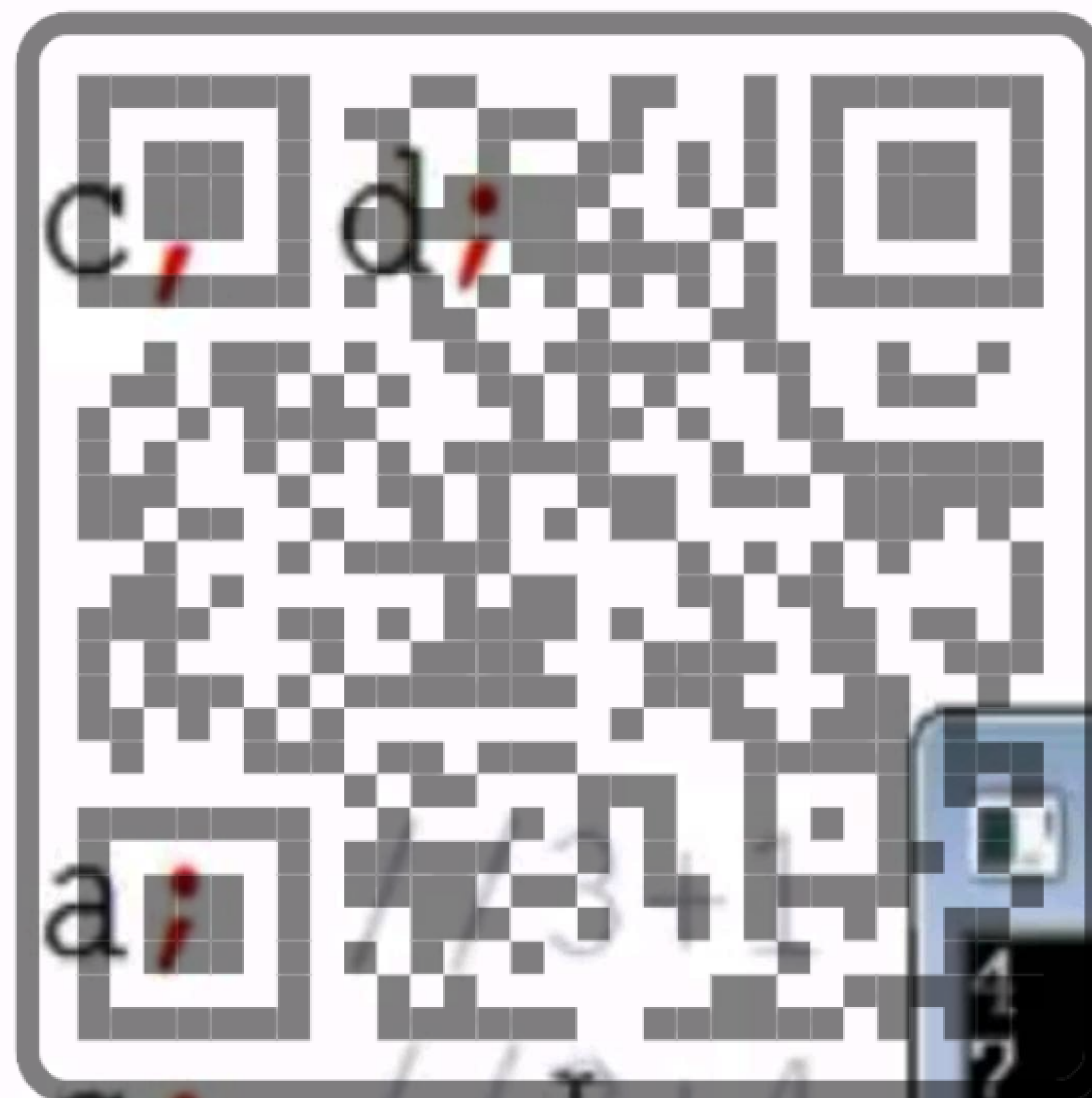
```
printf("%d\n", b);
```

**JOIN
FOR
MORE!!**




```
3  #define two 3
4  int main()
5  {
6      int a, b, c, d;
7      a = 1;
8      c = 4;
9      b = two + a; // 3+1
10     d = two + c; // 3+4
11     printf("%d\n", b);
12     printf("%d\n", d);
```

JOIN
FOR
MORE!!!




```
#define two 656565476765
```

```
int main(6)
```

```
{
```

```
int a, b, c, d;
```

```
a = 1;
```

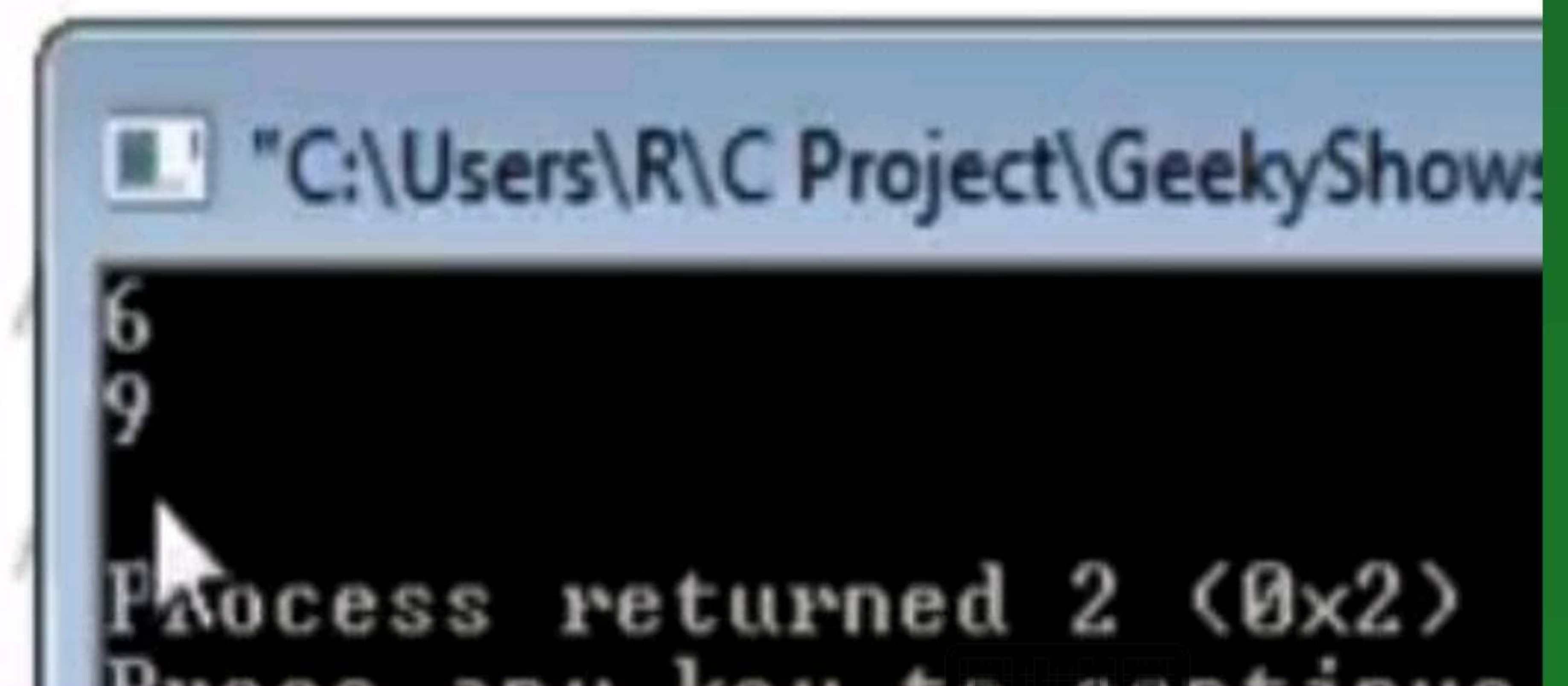
```
c = 4;
```

```
b = two + a; //3+1
```




```
1  /* Symbolic Constants */
2  #include<stdio.h>
3  #define two 5
4  int main()
5  {
6      int a, b, c, d;
7      a = 1;
8      c = 4;
9      b = two + a;
10     d = two + c;
```

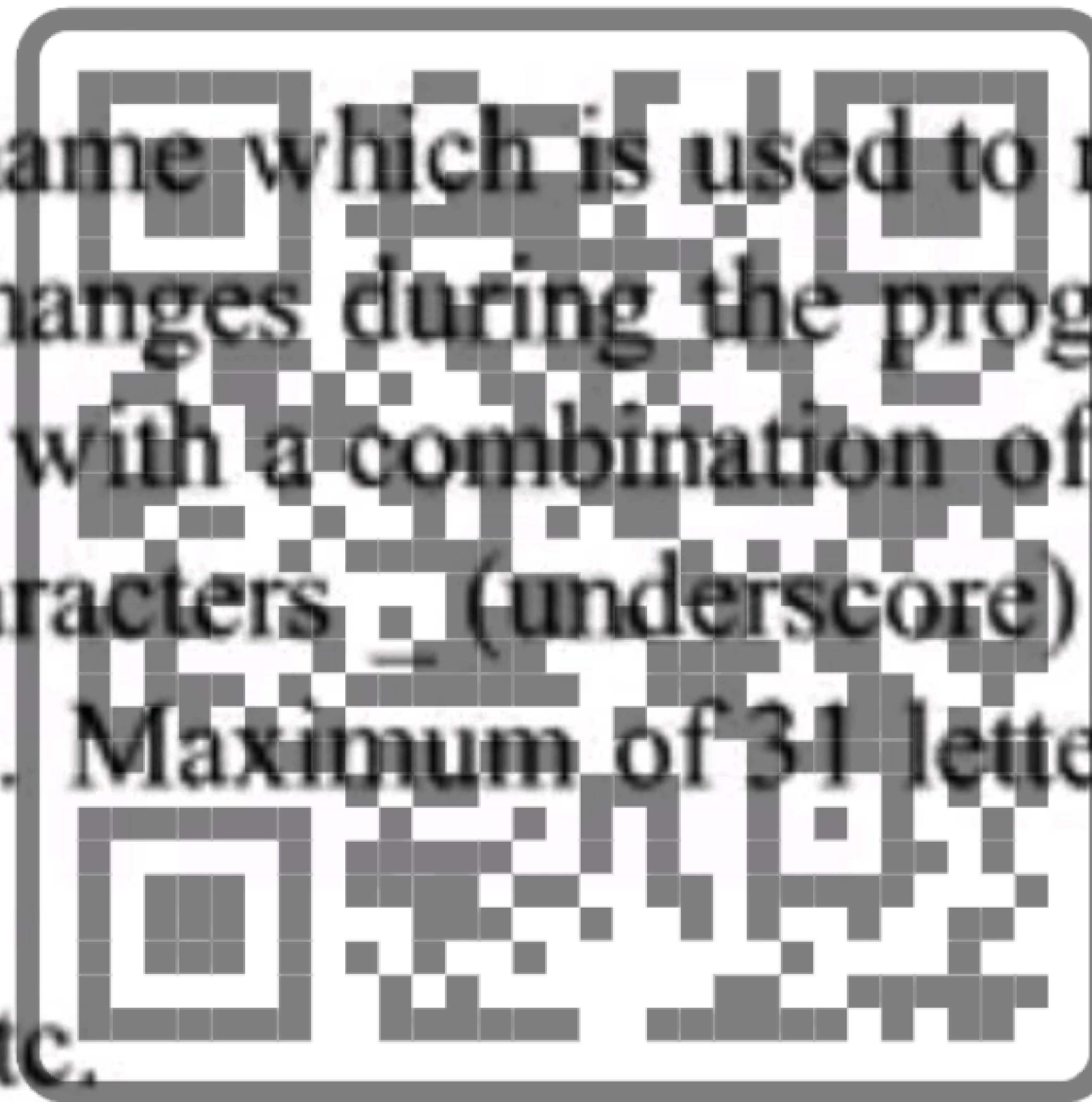
JOIN
FOR
MORE!!!



Variables

JOIN FOR MORE!!!

A variable is an identifier or a name which is used to refer a value and this value varies or changes during the program execution. A variable is written with a combination of letters, numbers and special characters _ (underscore) with the first letter being an alphabet. Maximum of 31 letters can be used to write a variable.
Ex: c, fact, b33, total_amount etc.



Variables

JOIN
FOR

MORE!!!

y	12.5	1001
		1002
		1003
		1004
		1005
Temperature	32	1006
Letter	'c'	1007
Number	—	1008
		1009

- *variable* is a named memory location
- variable *value* is data stored in variable
 - variable always has a value
- compiler removes variable name and assigns memory location
 - however, it is convenient to think that memory locations are labeled with variable names



Variable Declaration

- before use, every variable in C++ program needs to be *declared*
- *declaration* – introduction of the name to the compiler
- *type* – the kind of data stored in a variable
- variable declaration statement specifies

- type
- name

- declaration syntax: `type id, id, ..., id;`

- two commonly used numeric types are:

- `int` - whole positive or negative numbers:

`1, 2, -1, 0, -288, etc.`

- `double` - positive or negative numbers with fractional part:

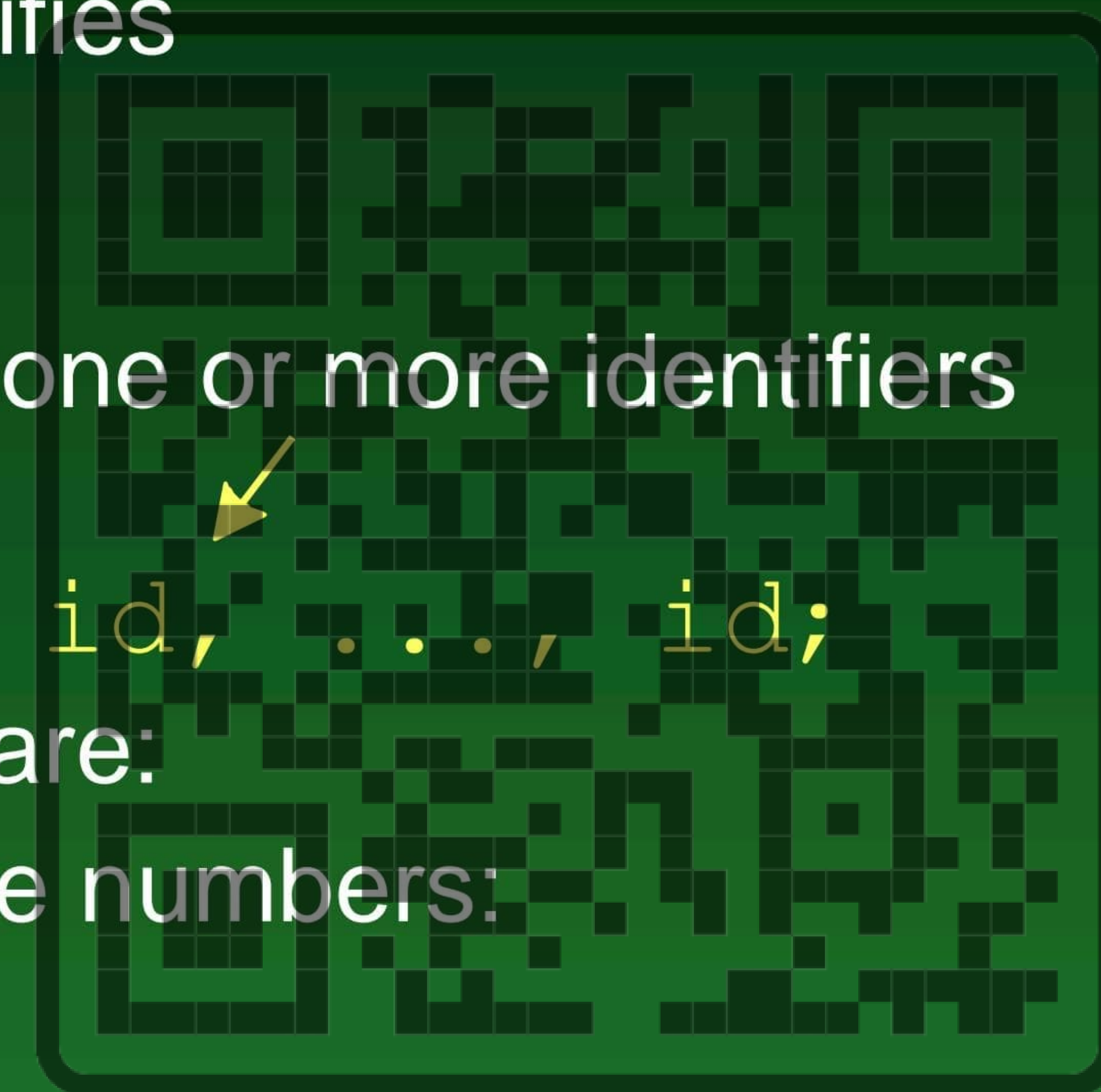
`1.75, -0.55`

- example declarations:

`int numberOfBars;`

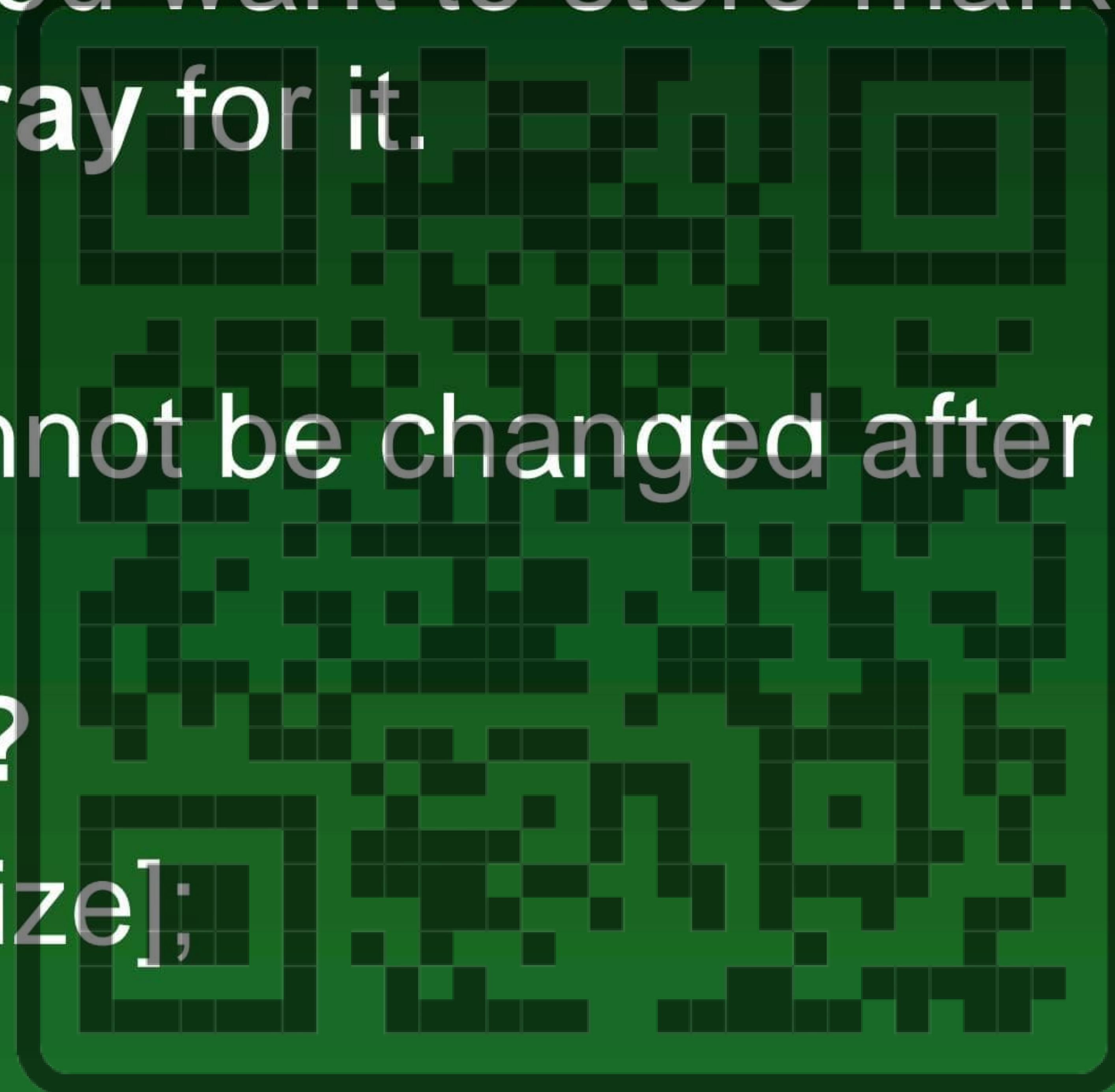
`double weight, totalWeight;`

one or more identifiers



Arrays

- **C Programming Arrays.** ... You will learn to declare, initialize and, access **array** elements with the help of examples.
An **array** is a collection of data that holds fixed number of values of same type. For example: if you want to store marks of 100 students, you can create an **array** for it.
- `float marks[100];`
- The size and type of arrays cannot be changed after its declaration.
- **How to declare an array in C?**
- `data_type array_name[array_size];`
- **For example,**
- `float mark[5];`
- Here, we declared an array, mark, of floating-point type and size 5. Meaning, it can hold 5 floating-point values.



- **Elements of an Array and How to access them?**
- You can access elements of an array by indices.
- Suppose you declared an array mark as above. The first element is mark[0], second element is mark[1] and so on.
- **Few key notes:**
- Suppose the starting address of mark[0] is 2120d. Then, the next address, a[1], will be 2124d, address of a[2] will be 2128d and so on. It's because the size of a float is 4 bytes.
- **How to initialize an array in C programming?**
- It's possible to initialize an array during declaration. For example,
- `int mark[5] = {19, 10, 8, 17, 9};`
- Another method to initialize array during declaration:
- `int mark[] = {19, 10, 8, 17, 9};`
-

mark[0]	mark[1]	mark[2]	mark[3]	mark[4]
19	10	8	17	9



Exercise

- 4.10: Answer see slides 15 & 16
- 4.11:

JOIN FOR MORE!!!
To assign value to 'char' variable we use single quotation ("),but to assign value to 'string' variable we use double quotation("").An example is given below.

char c1='\$';

string str="Hello string";



We can add two string variable using the plus (+) sign, we can also add two char variable using the plus(+) sign but their output is different.

```
string st="3" , st1 ;  
char c='3' , c1 ;  
  
st1=st+st; //give "33" as the output  
c1=c+c ; //give 'f' as the output
```

Adding two strings is same as concatenating the two strings. So if we concatenate "3" to "3" we get "33". Hence, st1 gives "33".

Adding two char variable add the integer value of the character. Which means '3'+'3' is same as '51+51' ('3' has the integer value 51 in ASCII table). Now this gives 102 which stands for the character 'f' in ASCII table. Hence 'c1' gives 'f'.



- 4.12: A char data item can also be a single numeric digit, such as 1,2,3. but C makes a distinction between the character '3' and the number 3. as character '3' cannot be used in a mathematical operation because it does not have a numeric value.
- 4.14: A program need to use long int data type when increase the range of the program because long int which is 8 byte for 62 bit processor.
- 4.15: see page 78, third paragraph.
- 4.16: see slide 8
- 4.17: see slide 26 to 28



- What is the difference between arithmetic operator and arithmetic assignment operator. (2018)
- What are the different types of operator used in C language? Define any two types. (2018)
- What is operator precedence? Write the precedence of arithmetic operator. (2017) Define an increment or decrement statement in C. (2017)
- Define operators. Explain different types of operators in C language. (2016)
- Define logical operator in C language. (2015)
- Explain all types of the operators in C language. (2014)
- What are the different operators used in C language? Write their precedence also. (2012)
- Define relational operators with suitable examples. (2011)

What is the difference between arithmetic operator and arithmetic assignment operator. (2018)

Arithmetic Operators:

Arithmetic operators are used to perform mathematical operations. An arithmetic expression is made of constant, variable, a combination of both connected by arithmetic operators.

Operation	Description	Type	Example
+	Addition	Binary	a + b
-	Subtraction	Binary	a - b
/	Division	Binary	a / b
*	Multiplication	Binary	a * b
%	Modulus / Remainder	Binary	a % b

ARITHMETIC OPERATORS IN C		EXAMPLE
+	Addition	10 + 2 = 12
-	Subtraction	10 - 2 = 8
*	Multiplication	10 * 2 = 20
/	Division	10 / 2 = 5
%	Modulus – It returns the remainder after the division	10 % 2 = 0 (Here remainder is zero). If it is 10 % 3 then it will be 1.

Arithmetic Assignment Operators

The **Arithmetic Operators**. The C programming language provides **operators** that perform addition, subtraction, multiplication, and division. ... You can also combine the **arithmetic operators** with the simple **assignment operator** to create compound **assignments**. For example, x+=1; and x=x+1; both increment the value of x by 1.

Chapter 06 Past paper questions with answers 2nd Year

Operation	Examples of expression	Description
+=	num += 5;	num = num + 5;
-=	num -= 5;	num = num - 5;
*=	num *= 5;	num = num * 5;
/=	num /= 5;	num = num / 5;
%=	num %= 5;	num = num % 5;

What are the different types of operator used in C language? Define any two types. (2018)
 What are the different operators used in C language? (2012)
 Define relational operators with suitable examples. (2011)

Classifications of operators:

C is very rich in operators and is sometimes called “the language of operators”. It has as many as 45 different operators. Some commonly used operators are:

- 1) Arithmetic operators
- 2) Assignment operators
- 3) Relational operators
- 4) Logical/Boolean operators
- 5) Increment and decrement operators

Logical /Boolean operators

The concept of logical operators is simple. They allow a program to make a decision based on multiple conditions. Each operand is considered a condition that can be evaluated to a true or false value.

Operator	Operation	example
&&	Logical AND	1 if a=b && c=d; else 0
	Logical OR	1 if a=b; c>d; else 0
!	Logical NOT	1 if !a; else 0

Relational operators:

Relational operators are used to compare two numeric operands. The operands can be variables, constant or expression that ultimately get evaluated to a numerical value . Since character can be represented as integers using ASCII code, therefore, operands to be compared can be

COMPILED BY SANA MAHMOOD 2

Chapter 06 Past paper questions with answers 2nd Year

character as well. All relational operator is made of two arithmetic expressions connected by a relational operator. It returns 0 when answer is false and 1 when answer is true.

Relational Operators			
Operation	Description	Examples of Expression	Value
<	Less than	6 < 9	1 (true)
<=	Less than or equal to	5 <= 5	1 (true)
>	Greater than	2 > 6	0 (false)
>=	Greater than or equal to	9 >= 5	1 (true)
==	Equal to	7 == 5	0 (false)
!=	Not equal to	6 != 5	1 (true)

What is operator precedence? Write the precedence of arithmetic operator. (2017)
? Write their precedence also. (2012)

Order of precedence:

Arithmetic Expression

Order of Precedence:

High: * / %

Low: + -

All operators have a precedence level. High precedence level operators are evaluated before lower ones. Operators of the same precedence level are evaluated from left to right

Arithmetic Expression

An expression may contain 2 or more arithmetic operators

Main issue:

ORDER OF PRECEDENCE



JOIN
FOR
MORE!!!

Chapter 06 Past paper questions with answers 2nd Year

1 st	(B) rackets
2 nd	(O) rder / Power / Indices
3 rd	(D) ivision
4 th	(M) ultiplication
5 th	(A) ddition
6 th	(S) ubtraction

B RACKETS () [] { }
O RDER POWER OF $\sqrt{\quad}$ ()²
D IVIDE / $\div$
M ULTIPLY * X
A DDITION +
S UBTRACTION -

COMPILED BY SANA MAHMOOD 3

Arithmetic Expression

Example:

$$\therefore (9 - (3 + 2)) * 3 = 12$$

Define an increment or decrement statement in C. (2017)



Increment and Decrement

This operation contains only one operand, that is, the operand which value will be incremented/ decremented

Symbol	Description	Examples of Expression	Description
++	Increment operand by 1	i++	i = i + 1
--	Decrement operand by 1	i--	i = i - 1

Prefix and Postfix

Increment and Decrement operators can be either in prefix or postfix forms

Expression	Description
i++	Value of i is incremented after being used in the expression
++i	Value of i is incremented before being used in the expression
i--	Value of i is decremented after being used in the expression
--i	Value of i is decremented before being used in the expression

Prefix and Postfix

Example:

j = i++ - 2

similar to

→ j = i - 2;

→ i = i + 1;

i 6
j 3

Prefix and Postfix

Example:

j = ++i - 2

similar to

→ i = i + 1;

→ j = i - 2;

i 6
j 4

Define operators. Explain different types of operators in C language. (2016)

Operators:

The processing phase of a program converts input data into information that is output. This process of converting data into information takes many forms. When working with numbers, the process often involves some form of mathematical operation. Operators are needed to perform such type of calculations that transform data into information.



Explain different types of operators in C language.

Explain all types of the operators in C language. (2014)

See in notes (chap 05)

Define logical operator in C language. (2015)

Logical /Boolean operators

The concept of logical operators is simple. They allow a program to make a decision based on multiple conditions. Each operand is considered a condition that can be evaluated to a true or false value.

Operator	Operation	example
&&	Logical AND	1 if a=b && c=d; else 0
	Logical OR	1 if a=b; c>d; else 0
!	Logical NOT	1 if !a; else 0

JOIN
FOR
MORE!!!



Chapter No. 05: Operators And Expressions

- **Operators:** The processing phase of a program converts input data into information that is output. This process of converting data into information takes many forms. When working with numbers, the process often involves some form of mathematical operation. Operators are needed to perform such type of calculations that transform data into information.

Example:

Marks: computer 75, Mathematics 80, Urdu 60, English 70

$75+80+60+70=285/400=0.7125*100=71.25\%= \text{A Grade}$

Operands: Operators are used to perform some 'operation' on the 'operands'. For example $a=b+c$; here b and c are operands, and the operation is being performed is 'addition' by using '+' operator.



Operator and Operand

What are operator and operand?

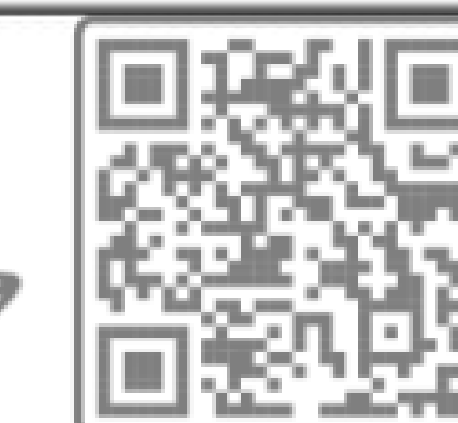
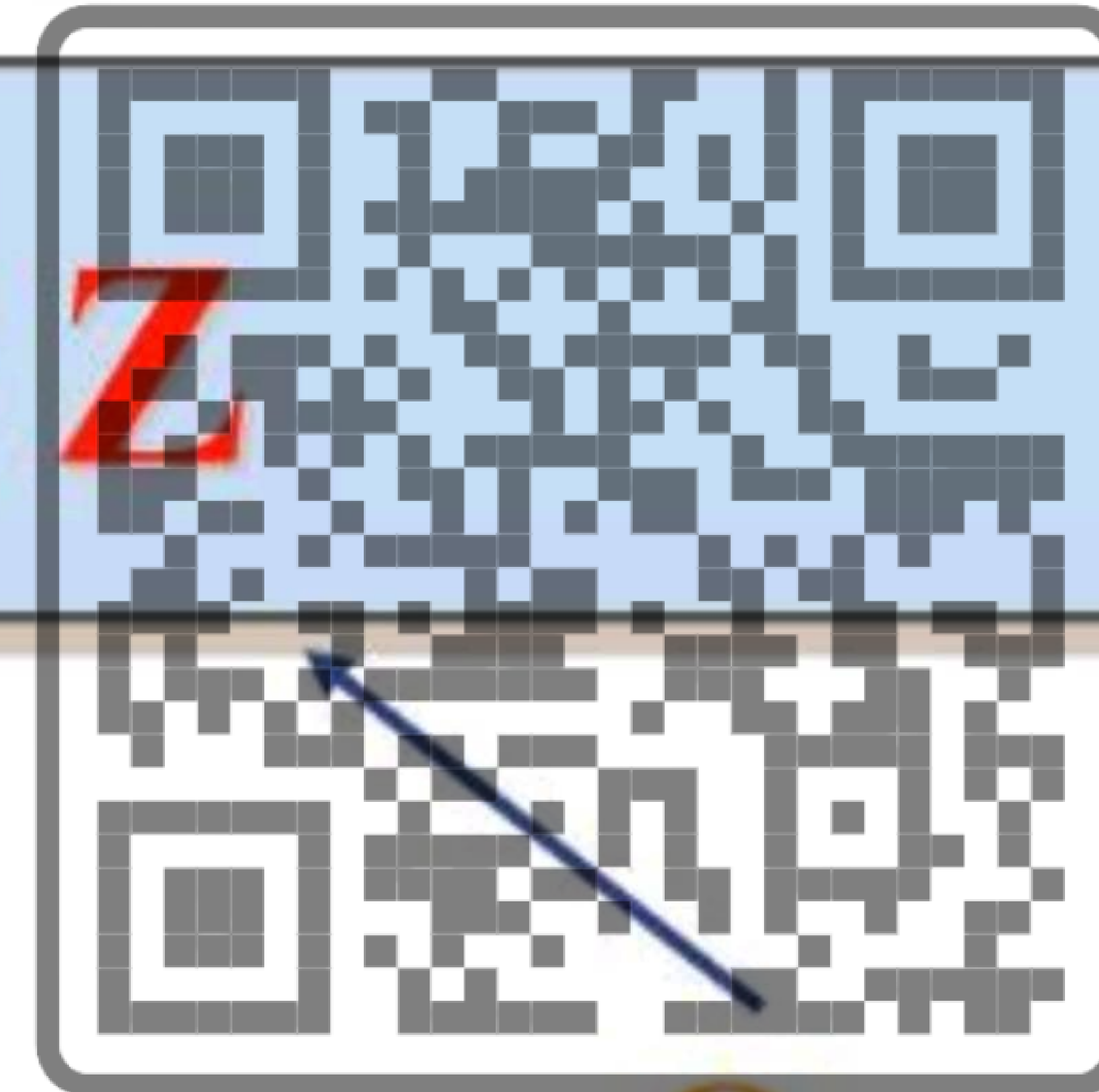
Example:

W + Z

Operand

Operator

Operand



Expression:

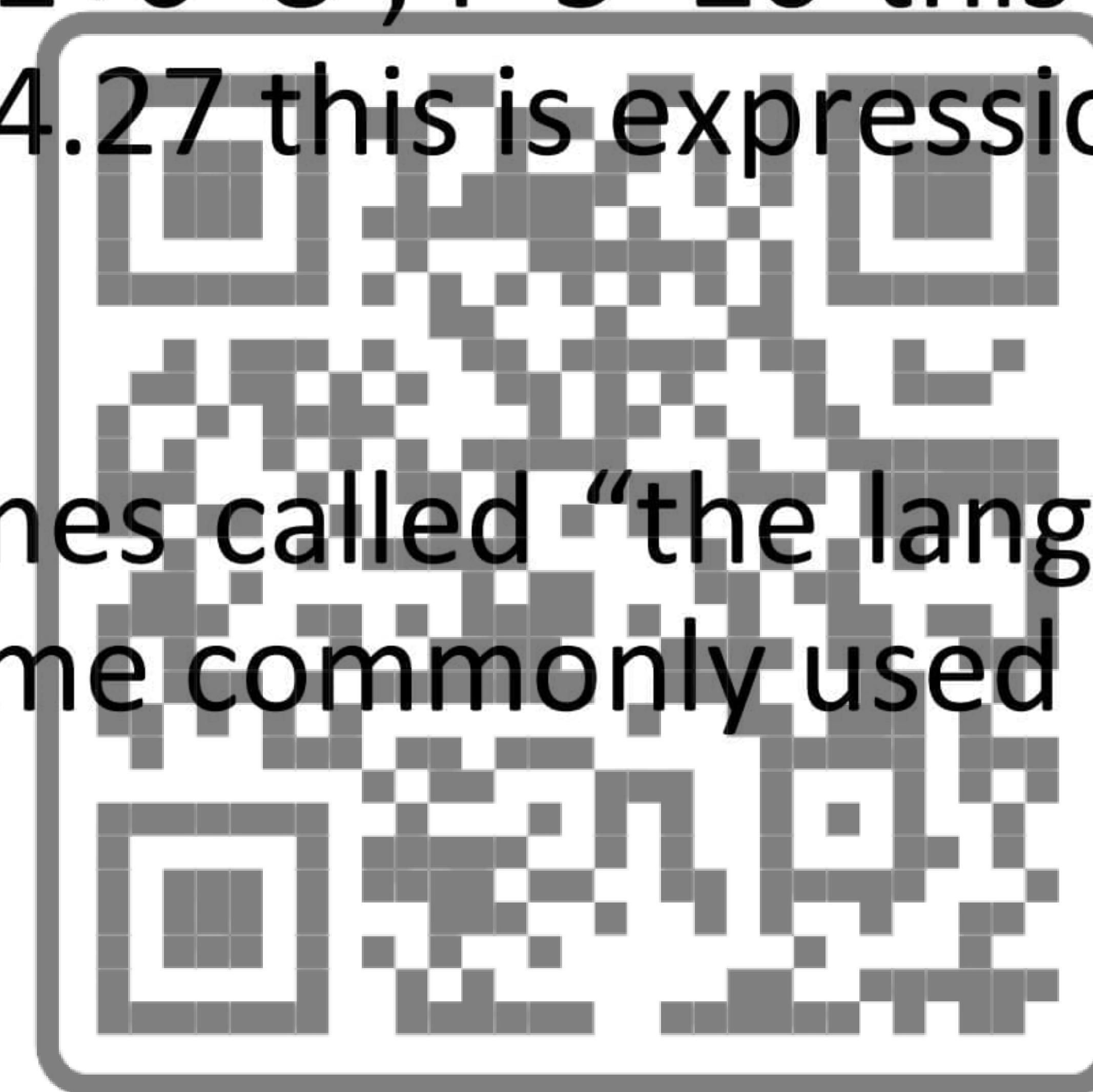
- An expression is a collection of operands and operators that can be evaluated to a single value. This is called expression.

Example: $2+6$ evaluates to 8 this means $2+6=8$, $4*5=20$ this is expression, $48/8=6$ this is expression. $6 + (7+6) (i)/(3*j/-k) - 84.27$ this is expression.

Classifications of operators:

C is very rich in operators and is sometimes called “the language of operators”. It has as many as 45 different operators. Some commonly used operators are:

- 1) Arithmetic operators
- 2) Assignment operators
- 3) Relational operators
- 4) Logical/Boolean operators
- 5) Increment and decrement operators



Arithmetic Operators:

- Arithmetic operators are used to perform mathematical operations. An arithmetic expression is made of constant, variable, a combination of both connected by arithmetic operators.

Operation	Description	Type	Example
+	Addition	Binary	$a + b$
-	Subtraction	Binary	$a - b$
/	Division	Binary	a / b
*	Multiplication	Binary	$a * b$
%	Modulus / Remainder	Binary	$a \% b$

ARITHMETIC OPERATORS IN C		OPERATION	EXAMPLE
+		Addition	$10 + 2 = 12$
-		Subtraction	$10 - 2 = 8$
*		Multiplication	$10 * 2 = 20$
/		Division	$10 / 2 = 5$
%		Modulus – It returns the remainder after the division	$10 \% 2 = 0$ (Here remainder is zero). it is $10 \% 3$ then it will be 1.

- The operands acted upon by arithmetic operators must represent numeric values.
- The remainder operators (%) requires that both operands be integers and the second operand be nonzero. $12\%0$ -----wrong
- Example: $5\%4.5$ ————— wrong
- $5\%4$ ————— correct

Example: **$12 \% 5 = 2$**

an integer (pointing to 12)

an integer (pointing to 5)

result (pointing to 2)

remainder (pointing to 2)

the result is the remainder of 12/5 (pointing to the entire equation)

an integer (pointing to 2)

JOIN FOR MORE!!!

Handwritten calculation showing 12 divided by 5 with a remainder of 2:

$$\begin{array}{r} 2 \\ 5 \overline{)12} \\ \underline{10} \\ 2 \end{array}$$

Example: **$12.0 \% 3 = ??$**

a float (pointing to 12.0)

an integer (pointing to 3)

INVALID! (pointing to the result ??)

Handwritten calculation showing 12.0 divided by 3 with a remainder of 0:

$$\begin{array}{r} 4 \\ 3 \overline{)12.0} \\ \underline{12} \\ 0 \end{array}$$


- Similarly the division operator (/) requires the second operand be non zero. 5/0--
-----wrong.



- Division of one integer quantity by another is referred to as integer division. This always results in a truncated quotient.(the decimal portion of the quotient will be dropped. 7/2= 3.5 but decimal portion will be dropped and answer is 3.



- On the other hand , if a division is carried out with two floating point numbers or the result will be floating point. Example if result 2 then will take 2.00.
- Examples: now suppose that i and j are integer variables whose values are 7 and 2 respectively.

- Expression

- $i+j$
- $i-j$
- $i*j$
- i/j
- $i\%j$

value

=9

=5

=14

=3

=1

**JOIN
FOR
MORE!!!**



See book page 92 for further examples.



Order of precedence

Arithmetic Expression

- An expression may contain 2 or more arithmetic operators
- Main issue:

**JOIN
FOR
MORE!!!**

ORDER OF PRECEDENCE

Arithmetic Expression

Examples:

$$5 + 6 = 11$$

$$5 + 6 * 2 = 17 \text{ or } 17?$$

$$2.5 + 6 - 2 * 2 = ??$$

$$12 / 6.0 - 2 * 2 = ??$$



Arithmetic Expression

Order of Precedence:

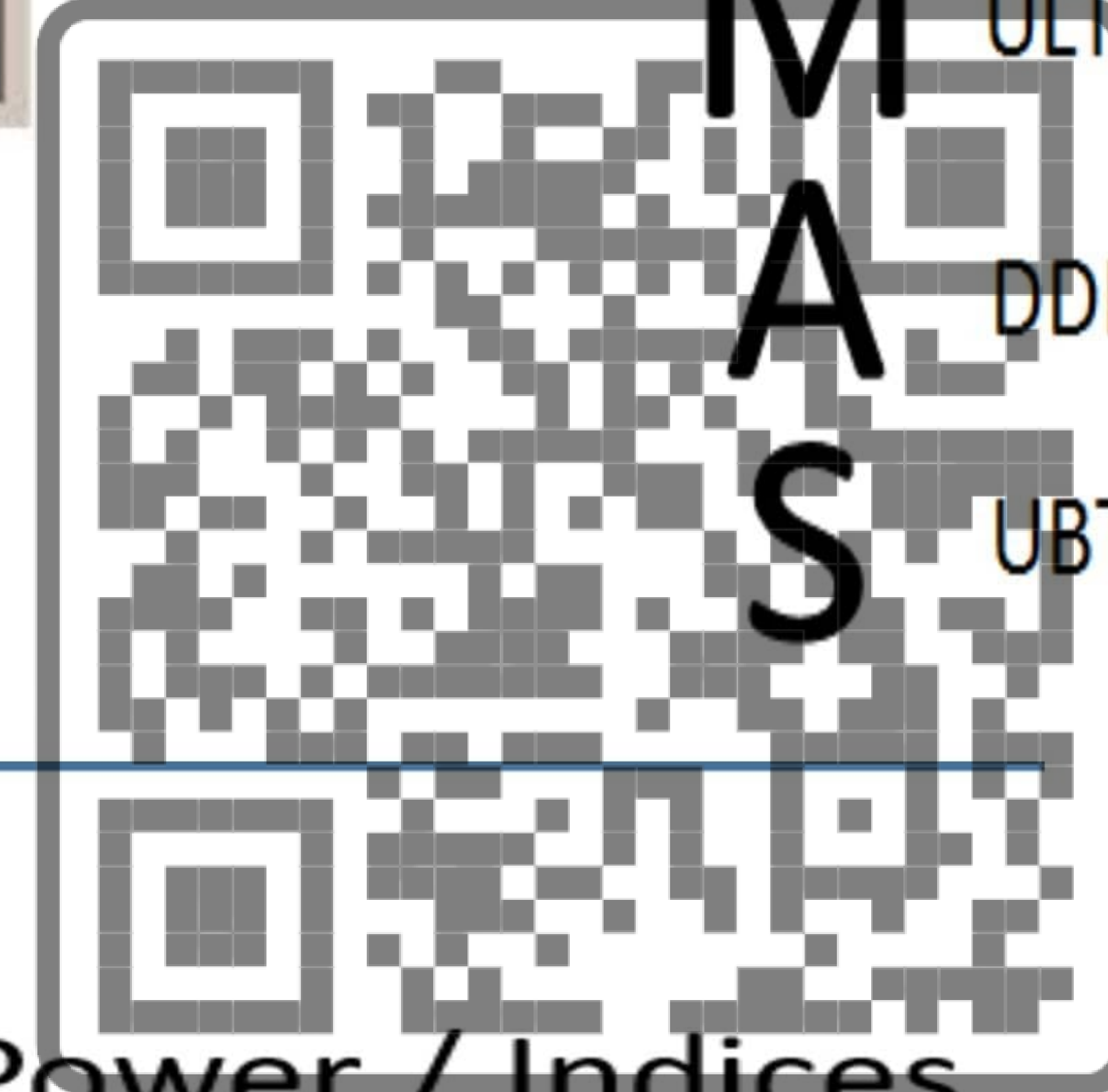
High:	$*$ / $\%$
Low:	$+$ -

All operators have a precedence level. High precedence level operators are evaluated before lower ones. Operators of the same precedence level are evaluated from left to right

JOIN
FOR
MORE!!!

- 
- | | |
|-----------------|----------------------------|
| 1 st | (B) rackets |
| 2 nd | (O) rder / Power / Indices |
| 3 rd | (D) ivision |
| 4 th | (M) ultiplication |
| 5 th | (A) ddition |
| 6 th | (S) ubtraction |

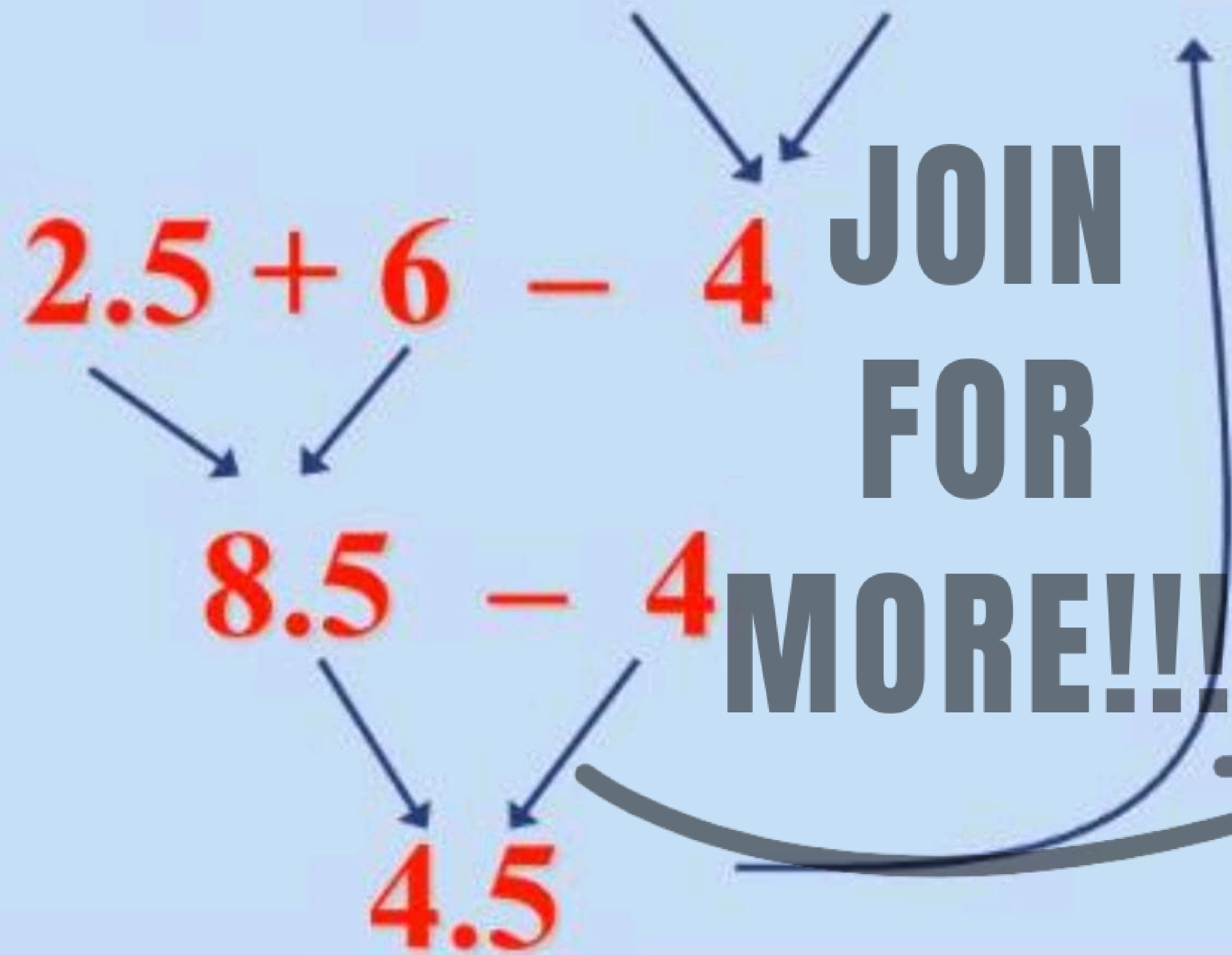
B	RACKETS	() [] { }
O	ORDER POWER OF	$\sqrt{\quad}$ $(\quad)^2$
D	DIVIDE	/ ÷
M	MULTIPLY	* X
A	ADDITION	+
S	SUBTRACTION	-



Arithmetic Expression

Example:

$$2.5 + 6 - 2 * 2 = 4.5$$



Try it!

Example:

$$12 + 6.0 - 2 * 2 = ??$$

What's the answer??



Arithmetic Expression

- All expressions in parentheses (brackets) must be evaluated prior to values outside brackets
- Nested parenthesized expressions must be evaluated from the inside out, with the innermost expression evaluated first

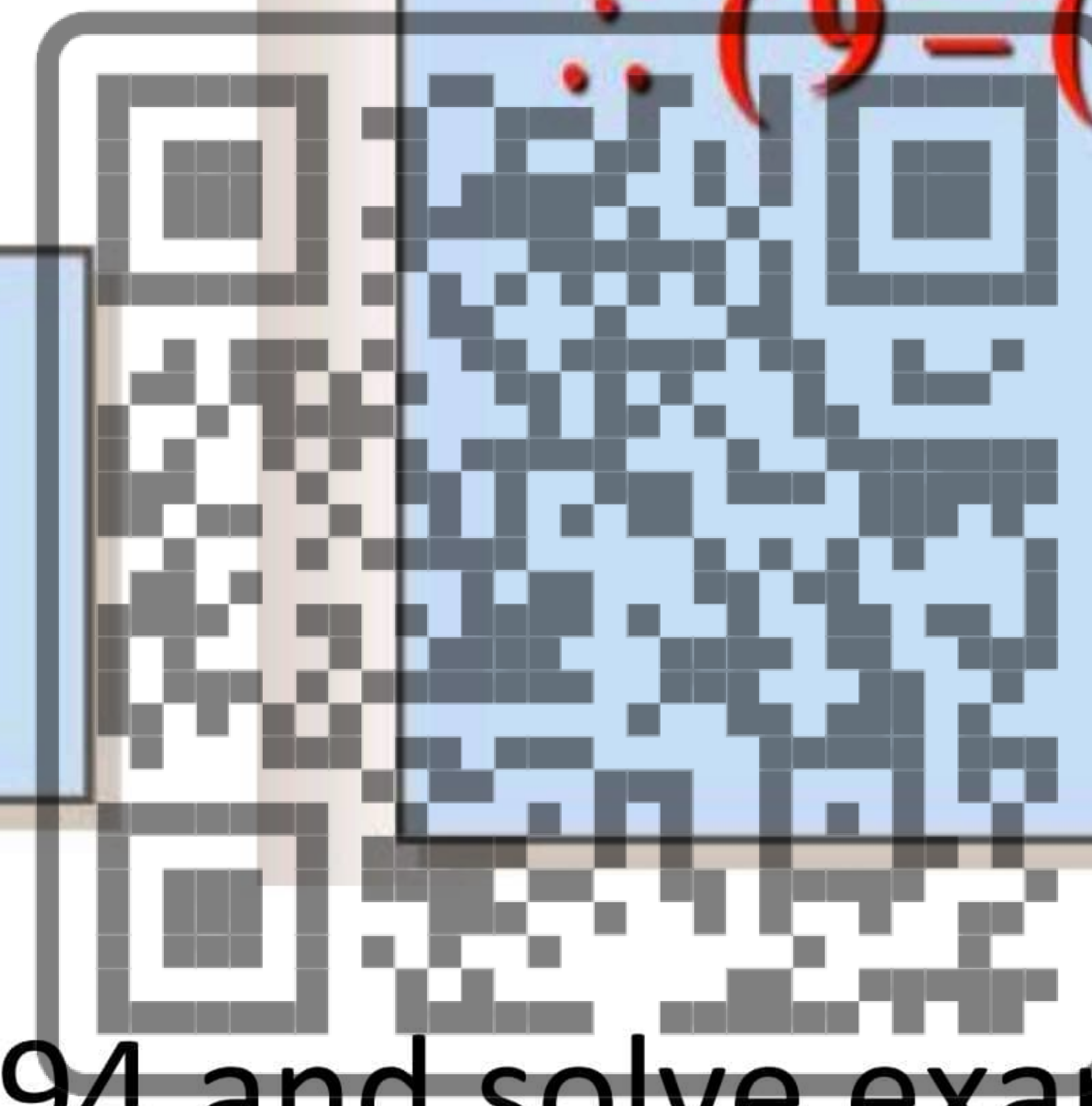
Example:

$$(9 - (3 + 2)) * 3 = ???$$

Arithmetic Expression

Example:

$$\therefore (9 - (3 + 2)) * 3 = 12$$



See book page 93 fig 5.1 and page 94 and solve example 5.1



Assignment Statement

There are 3 types of assignment:

- Simple
- Multiple
- Shorthand

- the basic assignment operator is =. This is used to assign value of an expression to an variable (identifier).

Simple Assignment

Syntax:

**JOIN
FOR
MORE!!!**

variable = expression;

Don't forget the semicolon !!



- The assignment operator has right to left associativity for example
- Expression meaning
- `A=5` assign integer value 5 to variable a.
- For further example see book page 95.
- here `a=b=c=d=1` -----meaning first the value 1 is assigned to d , then d assigned to c, then c assigned to b and finally b is assigned to a.
- Another example: `int i;`
- `double f;`
- `f=i=3.5 ;`-----meaning assigns 3.5 to f and 3 to i.

JOIN
FOR
MORE!!!



Operation	Examples of expression	Description
+=	num += 5;	num = num + 5;
-=	num -= 5;	num = num - 5;
*=	num *= 5;	num = num * 5;
/=	num /= 5;	num = num / 5;
%=	num %= 5;	num = num % 5;

Example:

pay += hour * rate * 2

similar to pay = pay + (hour * rate * 2)

→ pay + (hour * rate * 2)

→ pay + (8 * 5.00 * 2)

→ 100.00 + 80.00

→ 180.00

pay180.00

hour8

rate5.00

Example:

num += 5;

similar to num = num + 5

shorthand assignment operator

For example i=i*5 can be written as i*=5

see book page 96 for exercise practice.



Relational operators:

- Relational operators are used to compare two numeric operands . The operands can be variables , constant or expression that ultimately get evaluated to a numerical value . Since character can be represented as integers using ASCII code, therefore, operands to be compared can be character as well.
- All relational operator is made of two arithmetic expressions connected by a relational operator . It returns 0 when answer is false and 1 when answer is true.

Relational Operators

Operation	Description	Examples of Expression	Value
<	Less than	6 < 9	1 (true)
<=	Less than or equal to	5 <= 5	1 (true)
>	Greater than	2 > 6	0 (false)
>=	Greater than or equal to	9 >= 5	1 (true)
==	Equal to	7 == 5	0 (false)
!=	Not equal to	6 != 5	1 (true)

Must See book 97 and 98 for practice

Confusing equality (==) and assignment (=) operators
See book page 98 and 99

**JOIN
FOR
MORE!!!**



Logical /Boolean operators

- The concept of **logical operators** is simple. They allow a program to make a decision based on multiple conditions. Each operand is considered a condition that can be evaluated to a true or false value.



Operator	Operation	example
&&	Logical AND	1 if a=b && c=d; else 0
	Logical OR	1 if a=b; c>d; else 0
!	Logical NOT	1 if !a; else 0

**JOIN
FOR
MORE!!!**



Mantic/Logical Operators

See book page 99, 100 & 101

Symbol

&&

||

!

Description

AND

OR

NOT

a	b	a && b	a b	!a	!b
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	1	0	0

JOIN
FOR
MORE!!!



Example:

(a >= 1) && (b == 5)

~~→ (2 >= 1) && (b == 5)~~

~~→ 1 && (b == 5)~~

~~→ 1 && (5 == 5)~~

~~→ && 1~~

~~→ 1~~



Example:

!(c > a)

→ !(15 > 2)

→ !(1)

→ 0

a 2
b 5
c 15
d 17

a 2
b 5
c 15
d 17

JOIN
FOR
MORE!!!

Example:

$(c \geq (b * 3)) \parallel (a == 3)$

$\rightarrow (c \geq (5 * 3)) \parallel (a == 3)$

$\rightarrow (15 \geq 15) \parallel (a == 3)$

$\rightarrow 1 \parallel (a == 3)$

$\rightarrow 1 \parallel (2 == 3)$

$\rightarrow \parallel 0$

$\rightarrow 1$

a	2
b	5
c	15
d	17

Example:

$!((a < b) \parallel (c > d))$

$\rightarrow !((2 < 5) \parallel (c > d))$

$\rightarrow !(1 \parallel (c > d))$

$\rightarrow !(1 \parallel (15 > 17))$

$\rightarrow !(1 \parallel 0)$

$\rightarrow 1$

$\rightarrow 0$

a	2
b	5
c	15
d	17



Increment and Decrement

This operation contains only one operand, that is, the operand which value will be incremented/decremented

Symbol	Description	Examples of Expression	Description
++	Increment operand by 1	i++	i = i + 1
--	Decrement operand by 1	i--	i = i - 1

Increment and Decrement

Example:

```
int num;  
printf("Key-in a number: ");  
scanf("%d", &num);  
printf("Value before being incremented: %d\n", num);  
num++;  
printf("Value after being incremented: %d", num);
```

num 27

Key-in a number: 26

Value before being incremented: 26

Value after being incremented: 27



Prefix and Postfix

Increment and Decrement operators can be either in prefix or postfix forms

Expression	Description
i++	Value of i is incremented after being used in the expression
++i	Value of i is incremented before being used in the expression
i--	Value of i is decremented after being used in the expression
--i	Value of i is decremented before being used in the expression

Must See book page 101,102 and 103

JOIN
FOR
MORE!!!



Prefix and Postfix

Example:

j = ++i - 2

similar to

→ **i = i + 1;**

→ **j = i - 2;**

i **6**
j **4**

Prefix and Postfix

Example:

j = i++ - 2

similar to

→ **j = i - 2;**

→ **i = i + 1;**

i **6**
j **3**



Conversion of data type

- Suppose that an arithmetic expression such as $i + j$ has both value and a data type. For example, if both i and j have type `int` then the expression $i + j$ also has type `int`. But if i has `int` and j has `float` so `int` would be automatically converted to a `float`. See page 103 for further example

Coersion

Coersion is used to compute a value that is equivalent to its operand's value (based on the stated data type)

Example:

```
int x = 10;
```

```
float y;
```

```
y = (float) x
```

→ (float) 10

→ 10.000000

x

10

y

10.000000

Example:

```
int total, number;  
float average;
```

```
average = total / number;
```

→ 15 / 2

→ 7

total

15

number

2

average

7.000000

Example:

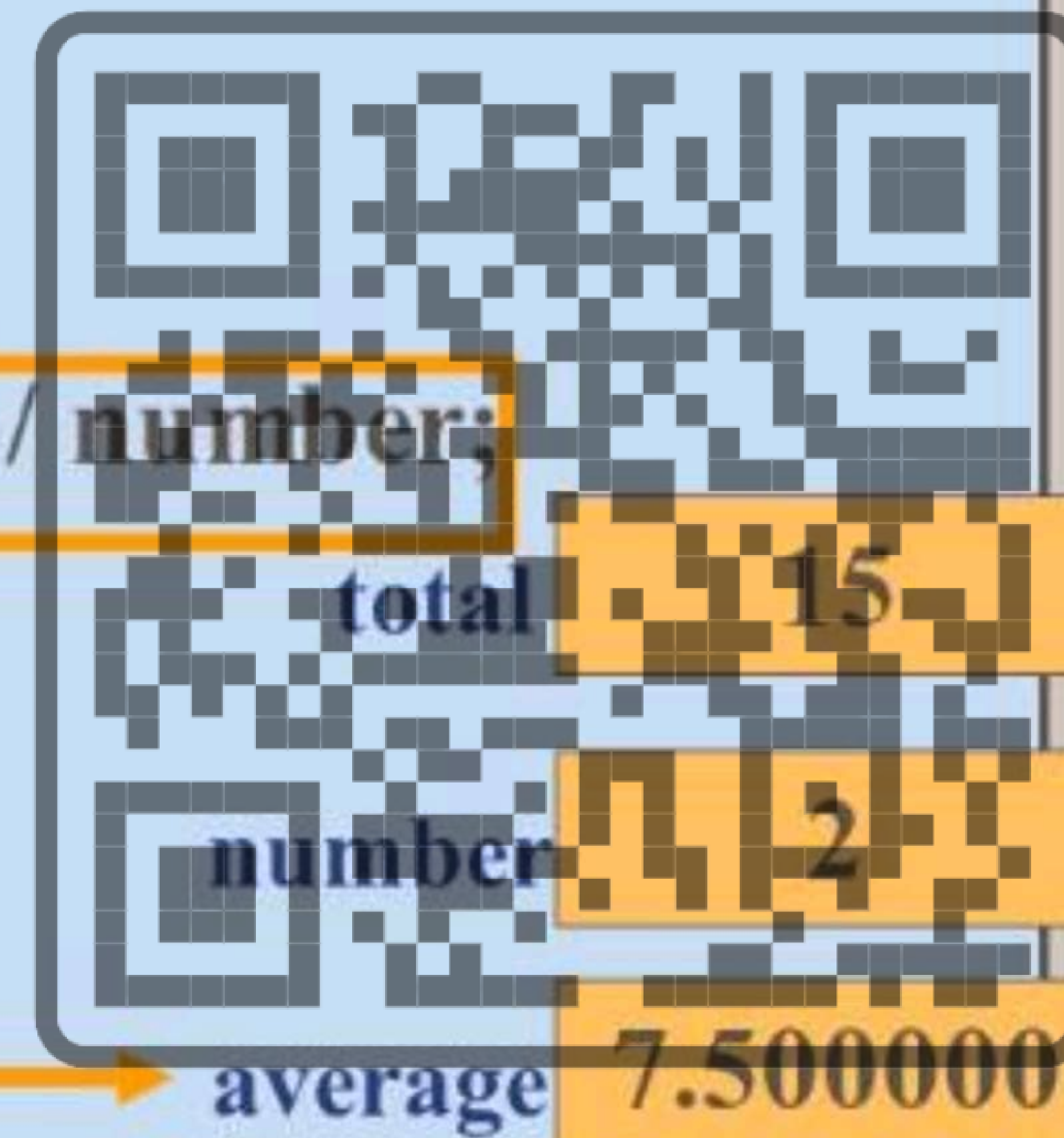
```
int total, number;  
float average;
```

...

```
average = (float) total / number;
```

→ 15.000000 / 2

→ 7.500000



Define different format specifiers used in any formatted function. (2018)

Define formatted I/O and unformatted I/O functions. (2017)

Define each format specifiers with an example. (2016)

a) %c b) %d c) %f d) %s e) %lf f) %ld

What are format specifiers and how do we use them in C language? (2013)

Define term function. discuss any five I/O functions used in C language. (2012)

Define three I/O functions with suitable examples. (2011)

What are library functions? Discuss any three library functions.

(2010) Define function. Why do we use it? Write down any five I/O functions.

Define different format specifiers used in any formatted function. (2018)

Define each format specifiers with an example. (2016)

a) %c b) %d c) %f d) %s e) %lf f) %ld

What are format specifiers and how do we use them in C language? (2013)

In C programming we need lots of format specifier to work with various data types. Format specifiers defines the type of data to be printed on standard output. Whether to print formatted output or to take formatted input we need format specifiers. Format specifiers are also called as format string. Each format specifier starts with the percent (%) symbol followed by a letter indicating the data type.

Scanf() format specifiers

Specifiers	Meaning
%d	Inputs an integer number
%u	Inputs an unsigned integer
%f	Inputs a float
%c	Inputs a character
%s	Inputs a string
%lf	Inputs a float long double
%ld	Inputs a signed integer long

Printf() format specifiers

Specifiers	Meaning
%d	Displays an integer number
%u	Displays an unsigned integer

%f	Displays a float
%c	Displays a character
%s	Displays a string
%lf	Displays a float long double
%ld	Displays a signed integer long

PREPARED BY SANA MAHMOOD 1

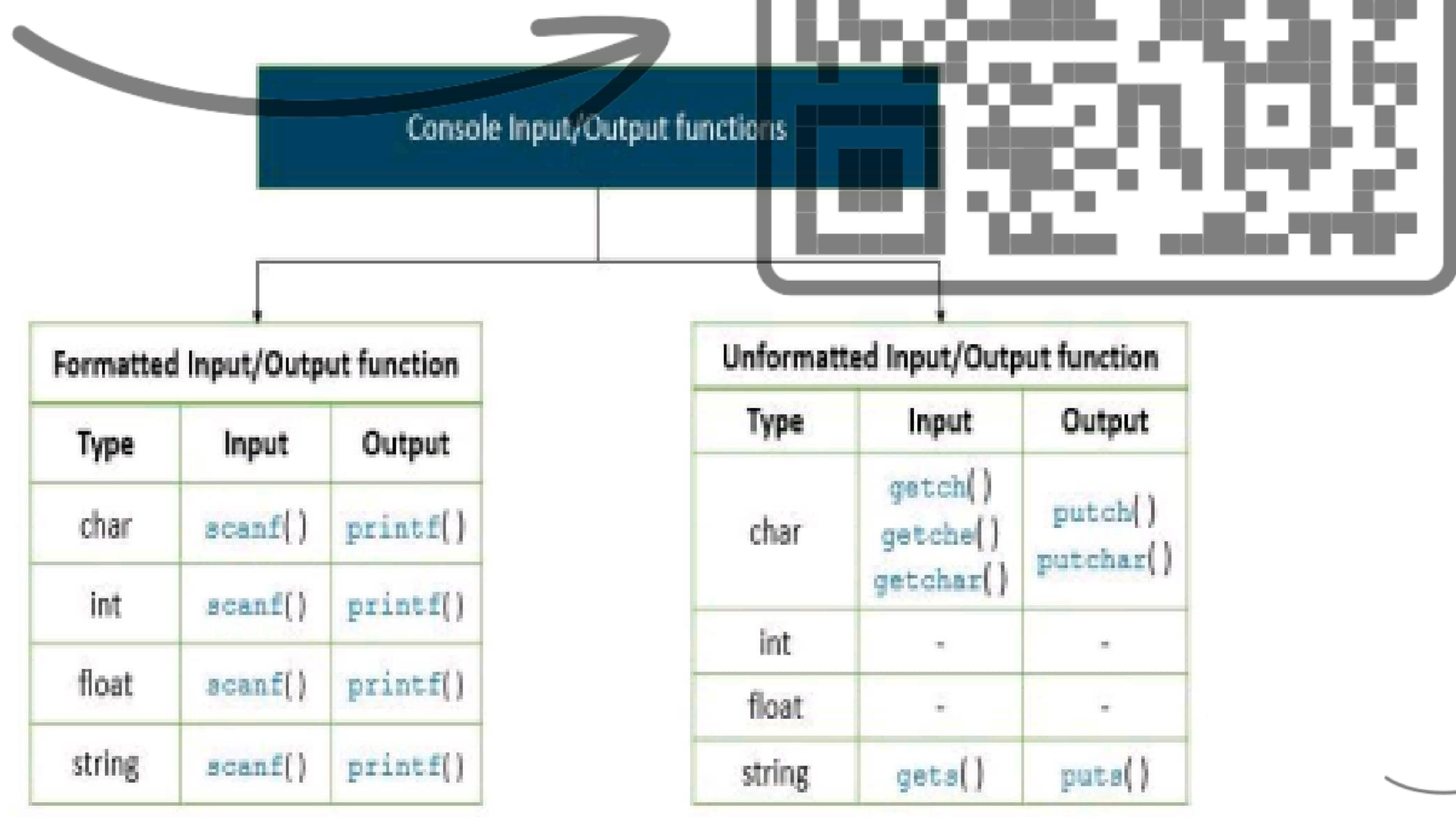
Chapter 06 Past paper with answers 2nd year

Define formatted I/O and unformatted I/O functions. (2017)

Formatted and Unformatted Input/Output

Unformatted Input/Output is the most basic form of input/output. Unformatted input/output transfers the internal binary representation of the data directly between memory and the file.

Formatted output converts the internal binary representation of the data to ASCII characters which are written to the output file. Formatted input reads characters from the input file and converts them to internal form. Formatted I/O can be either "Free" format or "Explicit" format, as described below.



Define term function. discuss any five I/O functions used in C language. (2012)

Define three I/O functions with suitable examples. (2011)

Define function. Why do we use it? Write down any five I/O functions.

Discuss any three library functions. (2010)

A function is a block of statements that performs a specific task. Suppose you are building an application in C language and in one of your program, you need to perform a same task more than once. In such case you have two options –

a) Use the same set of statements every time you want to perform the task b) Create a function to perform that task, and just call it every time you need to perform that task.

Example:

PREPARED BY SANA MAHMOOD 2

Chapter 06 Past paper with answers 2nd year

JOIN FOR MORE!!!



```
1 #include <stdio.h>
2
3 int main()
4 {
5     char ch;
6     ch = getchar();
7     printf("Charater entered is %c\n", ch);
8 }
9
```

Charater entered is a
Press any key to continue

getchar(): echos (display) character must follow the enter key. What we are type that comes in screen

```
1 #include <stdio.h>
2
3 int main()
4 {
5     char ch;
6     ch = getch();
7     printf("Charater entered is %c\n", ch);
8 }
9
```

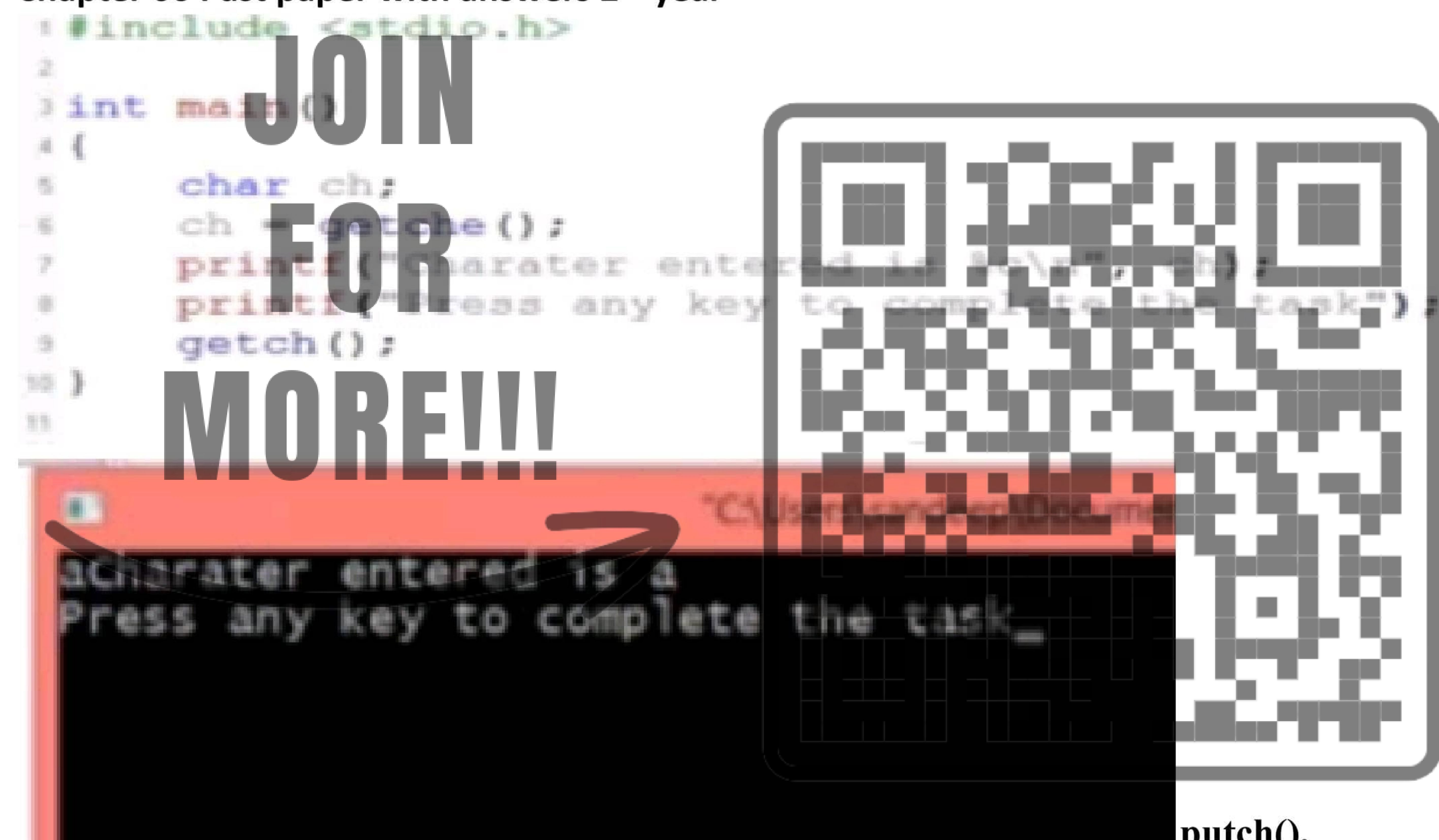


Charater entered is a
Press any key to continue . . .

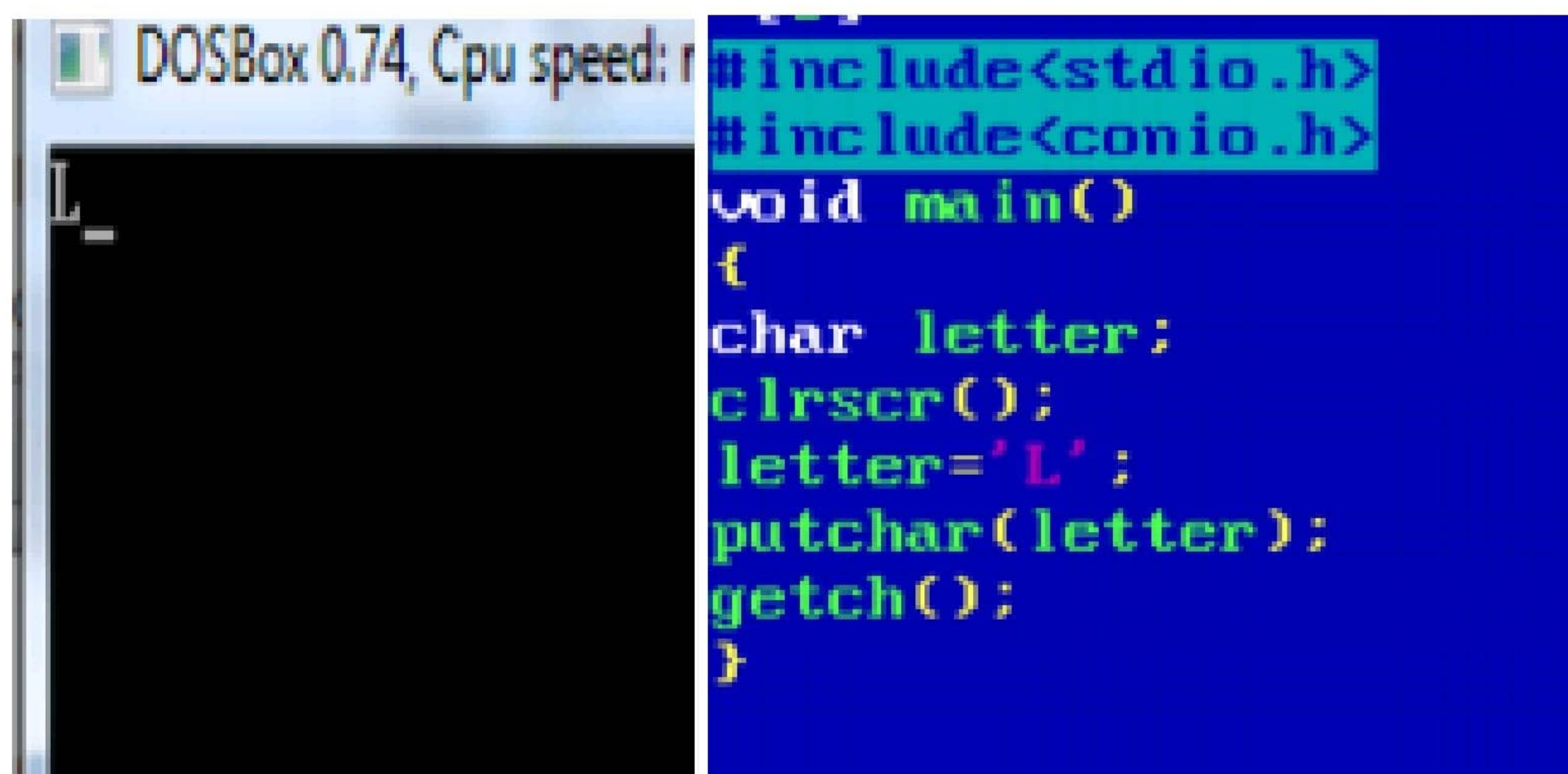
getche: echos character. A character is read instant without waiting hit enter key.
getche is same as getch but it echo the character now see

PREPARED BY SANA MAHMOOD 3

Chapter 06 Past paper with answers 2nd year

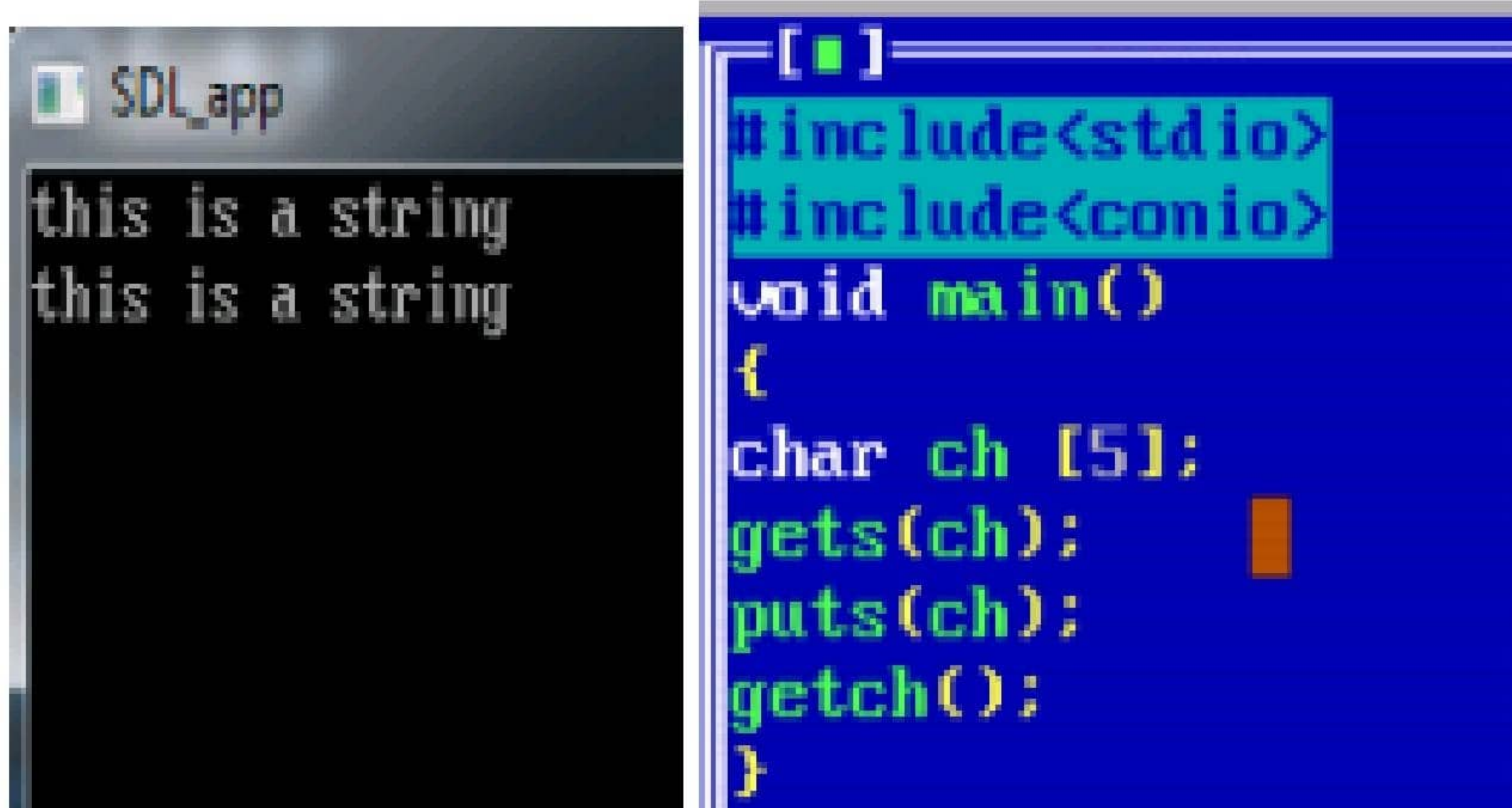


putch(), putchar(): Puts a single character on screen. Both the functions are exactly same.



gets(): Receives string of characters and put it in a char array. A string is entered from the keyboard, will terminate with a newline character.





```

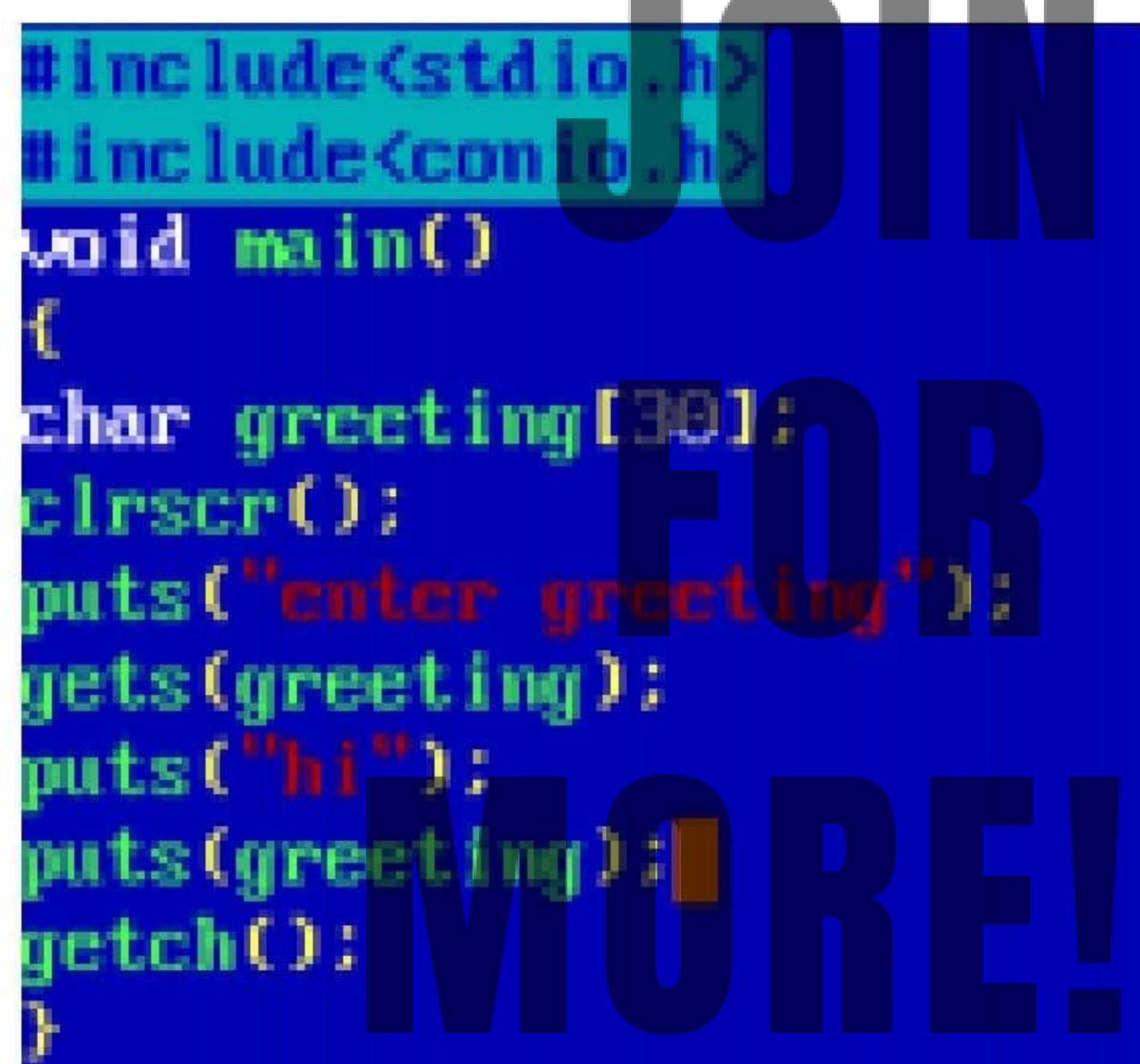
#include<stdio>
#include<conio>
void main()
{
    char ch [5];
    gets(ch);
    puts(ch);
    getch();
}

```

PREPARED BY SANA MAHMOOD 4

Chapter 06 Past paper with answers 2nd year

puts(): It puts backs them to screen



```

#include<stdio.h>
#include<conio.h>
void main()
{
    char greeting[30];
    clrscr();
    puts("enter greeting");
    gets(greeting);
    puts("hi");
    puts(greeting);
    getch();
}

```

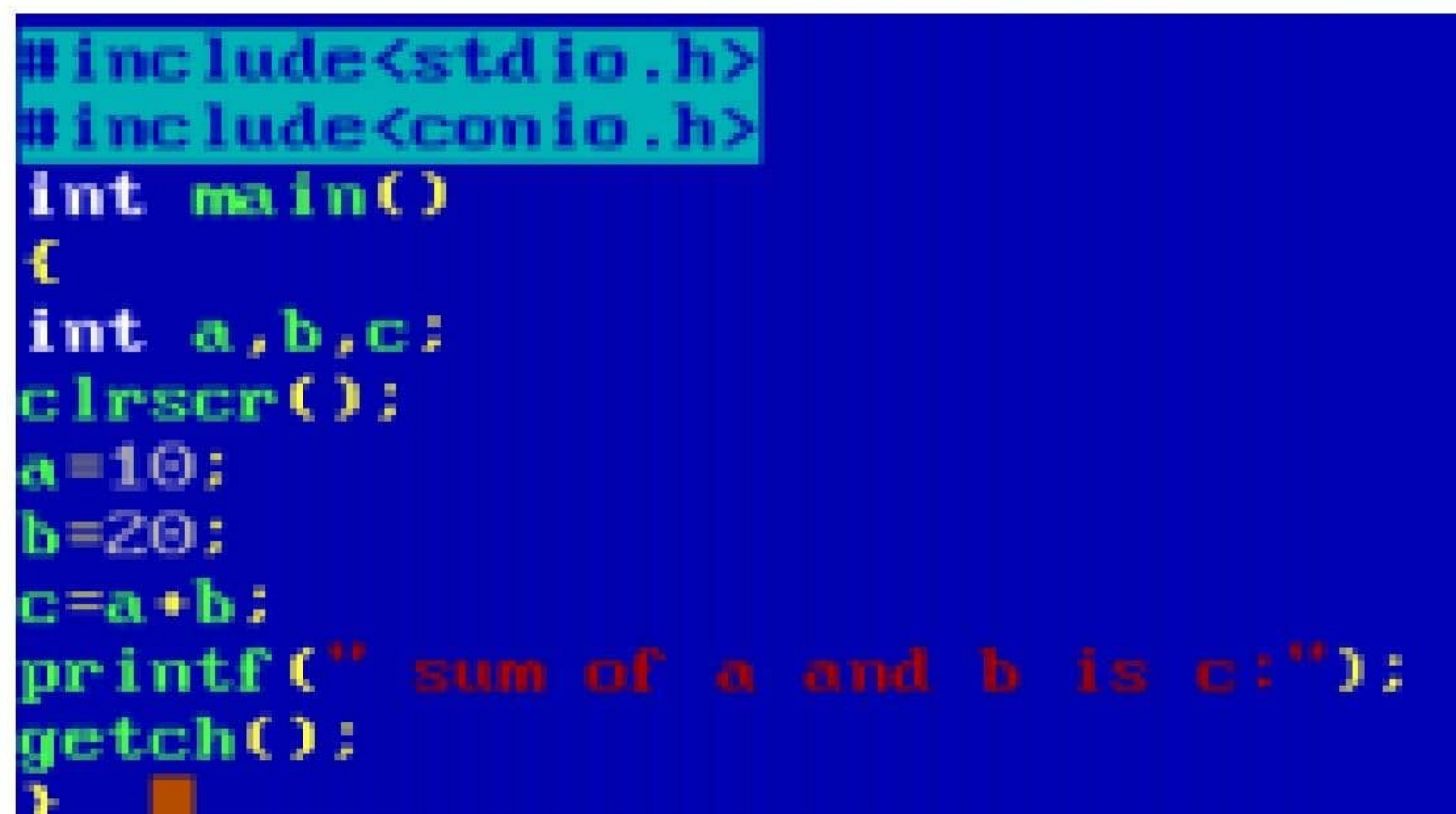


```

enter greeting
hi
hi
hi

```

The printf() Function: This function is most versatile way to display output on screen. When using printf(), you need to include the standard input/output header file, STDIO.H. `int printf(format-string [,arguments,...]);`



```

#include<stdio.h>
#include<conio.h>
int main()
{
    int a,b,c;
    clrscr();
    a=10;
    b=20;
    c=a+b;
    printf(" sum of a and b is c:");
    getch();
}

```

The scanf() function reads data from the keyboard according to a specified format and assigns the input data to one or more program variables .Like printf(), scanf() uses a format string to describe the format of the input.




```
type Integer,float,character,string
10
4.5
a
sana mahmood

Integer = 10
float = 4.500000
character = a
string = sana
```

```
void main()
{
    int num;
    float fnum;
    char letter, name[15];
    clrscr();
    printf("type Integer,float,character,string\n");
    scanf("%d %f %c %s",&num, &fnum, &letter, name);
    printf("\nInteger = %d",num);
    printf("\nfloat = %f",fnum);
    printf("\ncharacter = %c",letter);
    printf("\nstring = %s",name);
    getch();
}
```

What are library functions?

C library functions are the built-in functions that are accompanied along with the C compiler. they are defined and declared in their respective header files. they are the readily available functions which can be used directly without we are writing a code for it. ex: printf() and scanf () are the formatted input and output functions which are declared and defined in the header file <stdio.h>

**JOIN
FOR
MORE!!!**



PREPARED BY SANA MAHMOOD 5



Define the term function.

- A function is a block of statements that performs a specific task. Suppose you are building an application in C language and in one of your program, you need to perform a same task more than once. In such case you have two options –
 - a) Use the same set of statements every time you want to perform the task
 - b) Create a function to perform that task, and just call it every time you need to perform that task.



Console simply means screen and keyboard. There are two types of a console I/O functions:

- **Formatted input-output function**

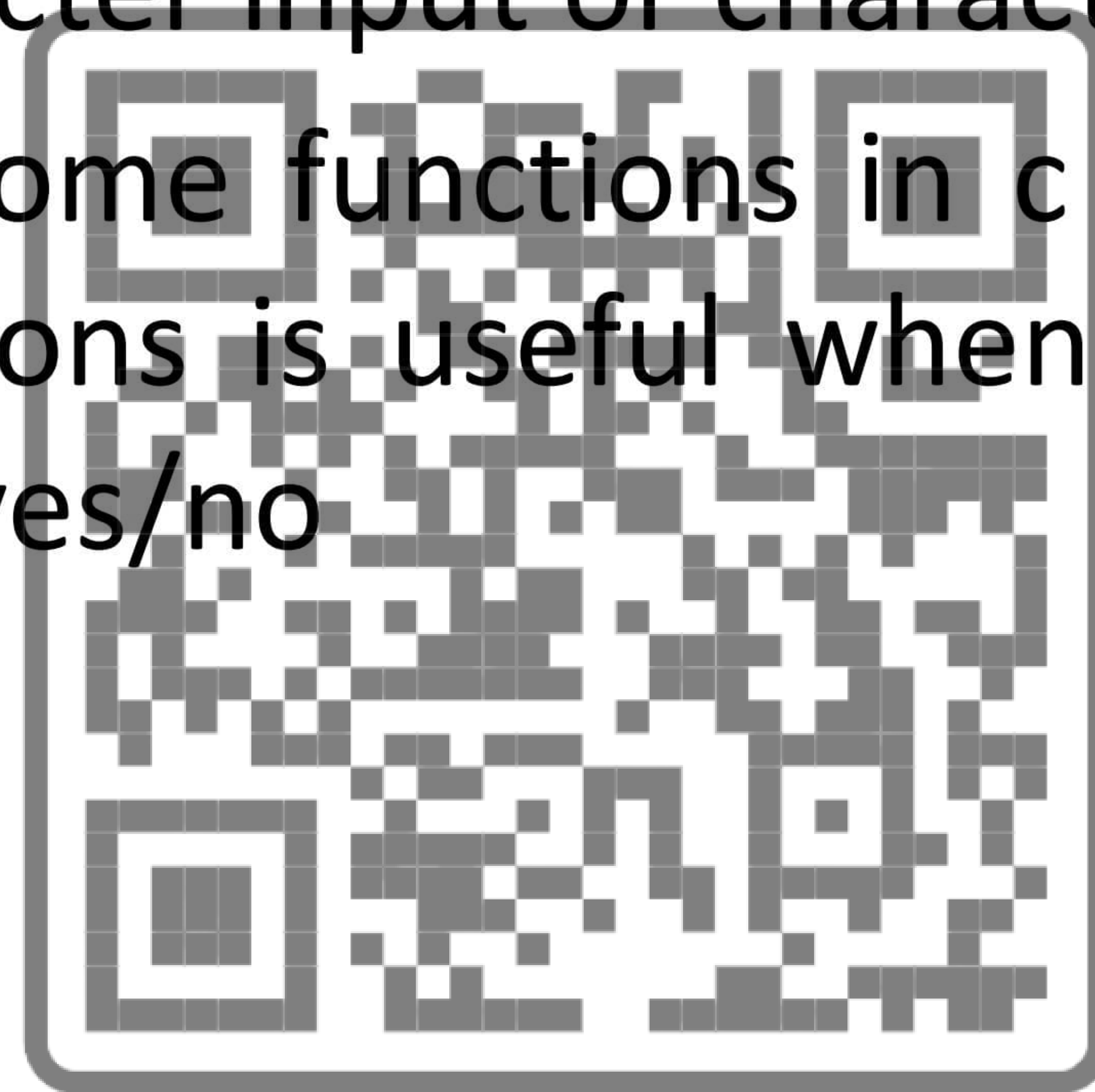
- **Unformatted input-output function**

- The major difference is that formatted function allows us to format the input from the keyboard and the output to be displayed on the screen.
- Unformatted Input/Output is the most basic form of input/output. Unformatted input/output transfers the internal binary representation of the data directly between memory and the file.
- Formatted output converts the internal binary representation of the data to ASCII characters which are written to the output file.



Formatted & Unformatted I/O functions

- Formatted input reads characters from the input file and converts them to internal form.
- This function is used for character input or character output.
- Single Character: there are some functions in c that handle characters only. These functions is useful when the user is **prompted (want)** to answer yes/no



Console Input/Output functions

JOIN
FOR
MORE!!!

Formatted Input/Output function		
Type	Input	Output
char	<code>scanf()</code>	<code>printf()</code>
int	<code>scanf()</code>	<code>printf()</code>
float	<code>scanf()</code>	<code>printf()</code>
string	<code>scanf()</code>	<code>printf()</code>

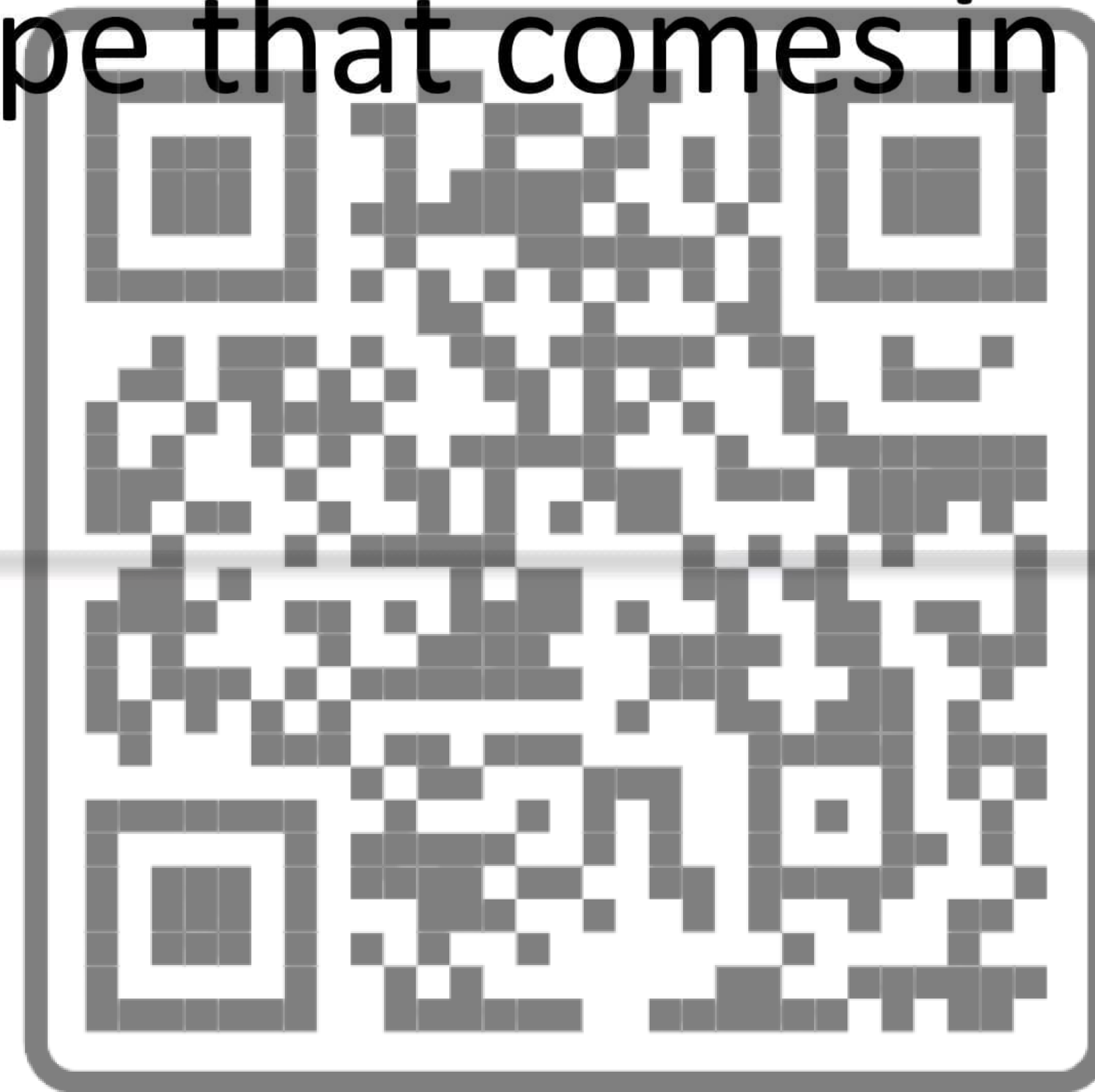
Unformatted Input/Output function		
Type	Input	Output
char	<code>getch()</code>	<code>putch()</code>
	<code>getche()</code>	<code>putchar()</code>
	<code>getchar()</code>	
int		-
float	-	-
string	<code>gets()</code>	<code>puts()</code>



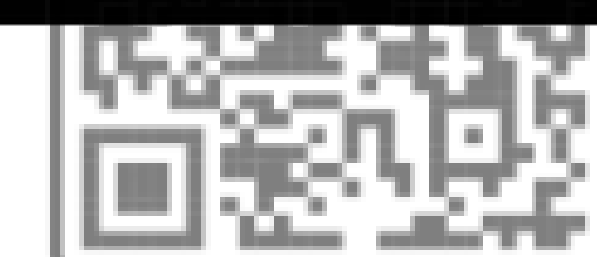
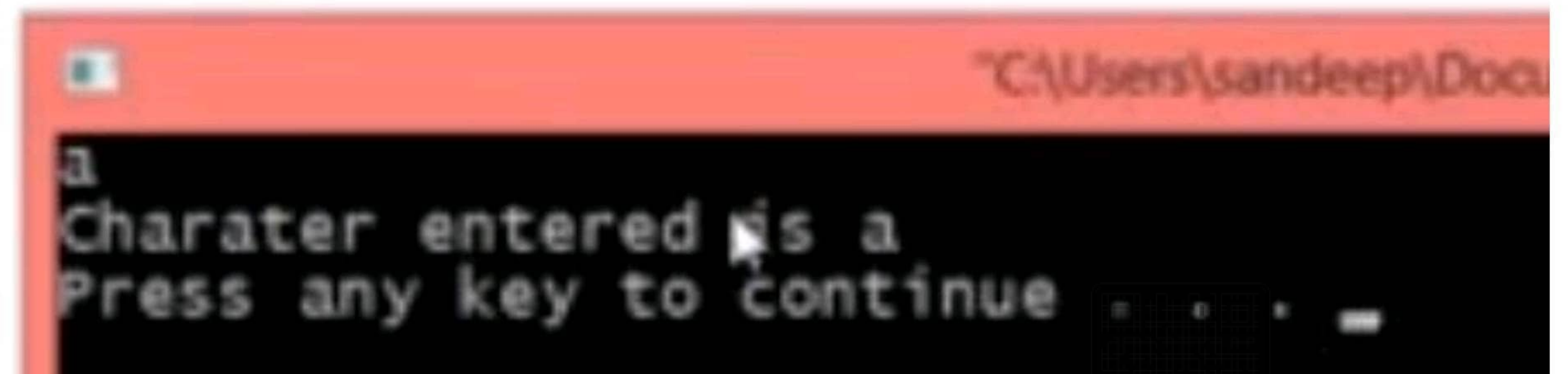
getchar(), getch(), getche(): Reading a single character using these functions.

- getchar(): echos (display) character must follow the enter key. What we are type that comes in screen
- Example:

**JOIN
FOR
MORE!!!**



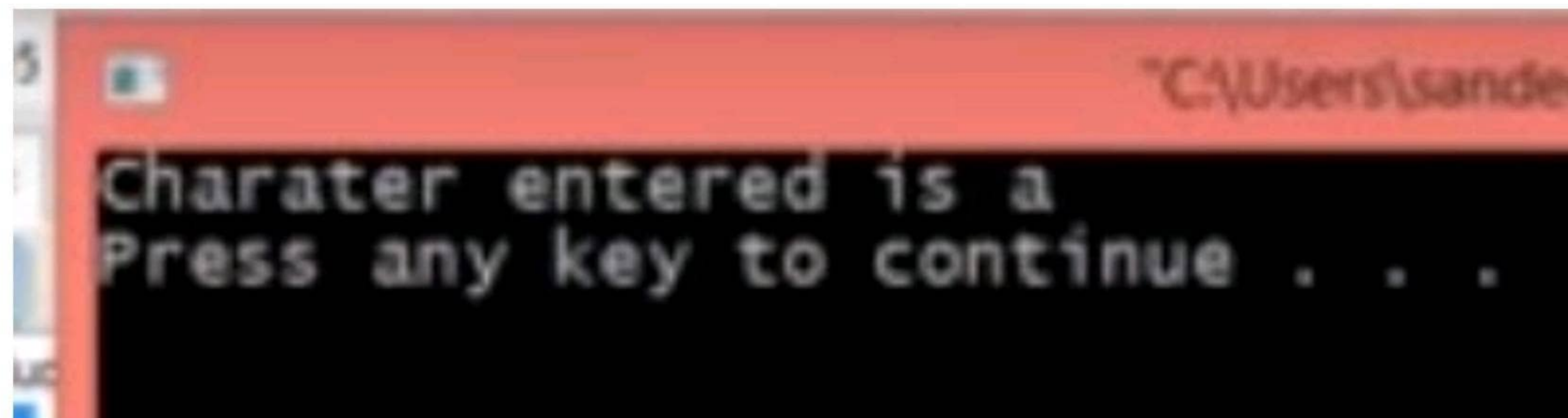
```
1 #include <stdio.h>
2
3 int main()
4 {
5     char ch;
6     ch = getchar();
7     printf("Charater entered is %c\n", ch);
8 }
9
```



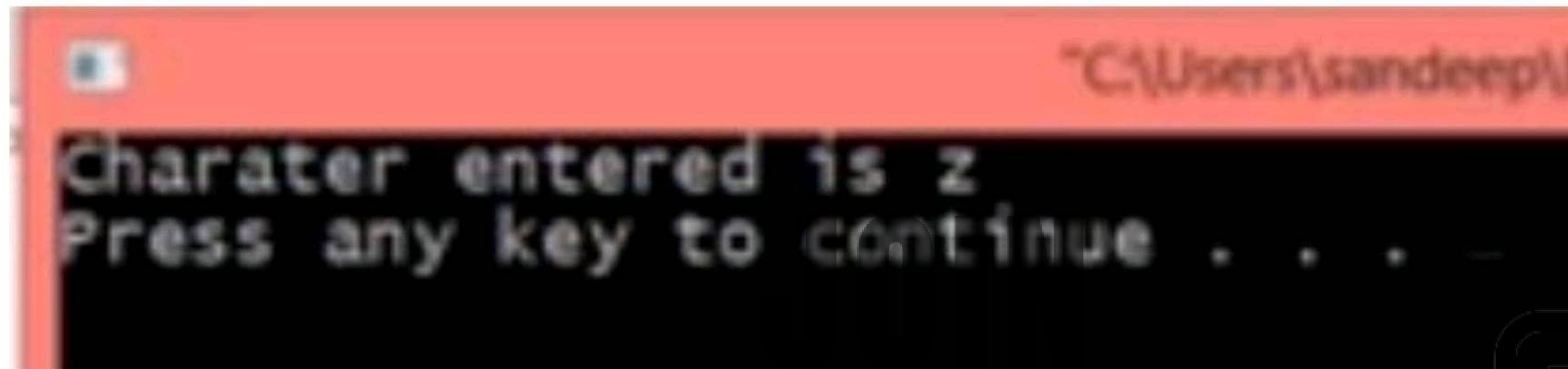
getch

- getch: Will not echo character. A character read instantly without waiting to hit enter key.

```
1 #include <stdio.h>
2
3 int main()
4 {
5     char ch;
6     ch = getch();
7     printf("Charater entered is %c\n", ch);
8 }
9
```



- When I was enter the character it is given to the ch but not echoed (displayed).



Moreover did not wait to press enter also.

getche

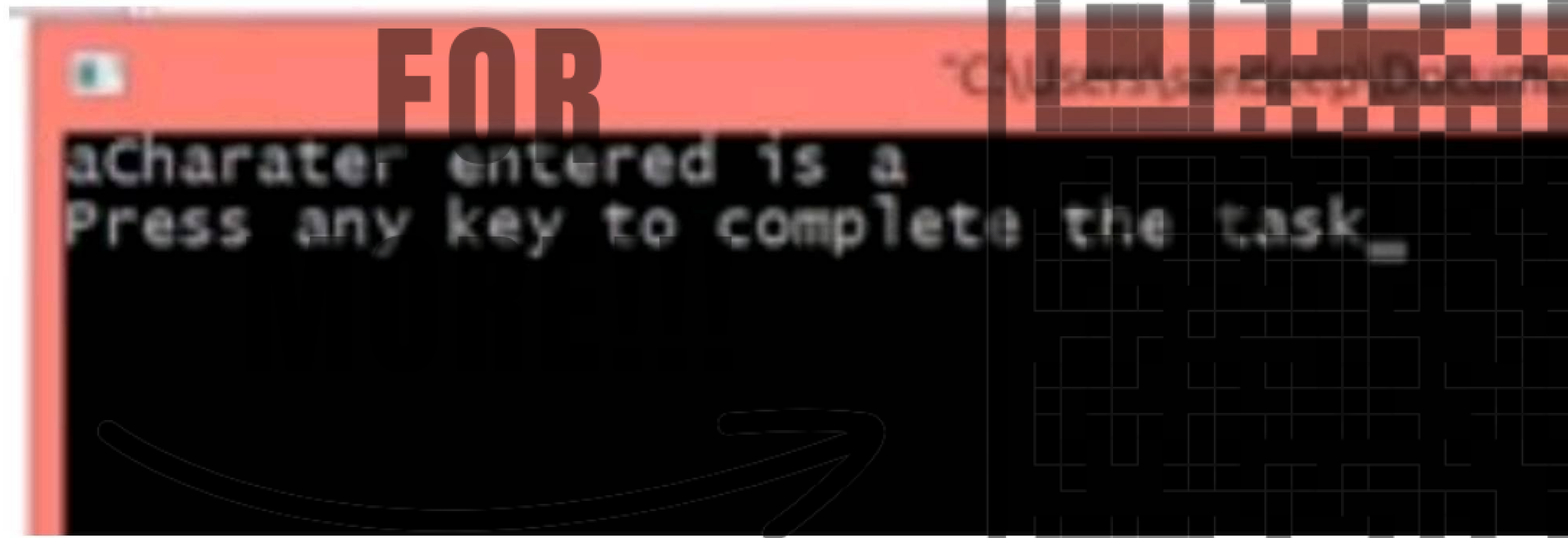
getche: echos character. A character is read instant without waiting hit enter key.

getche is same as getch but it echo the character now see




```
1 #include <stdio.h>
2
3 int main()
4 {
5     char ch;
6     ch = getch();
7     printf("Charater entered is %c\n", ch);
8     printf("Press any key to complete the task");
9     getch();
10 }
11
```

**JOIN
FOR**

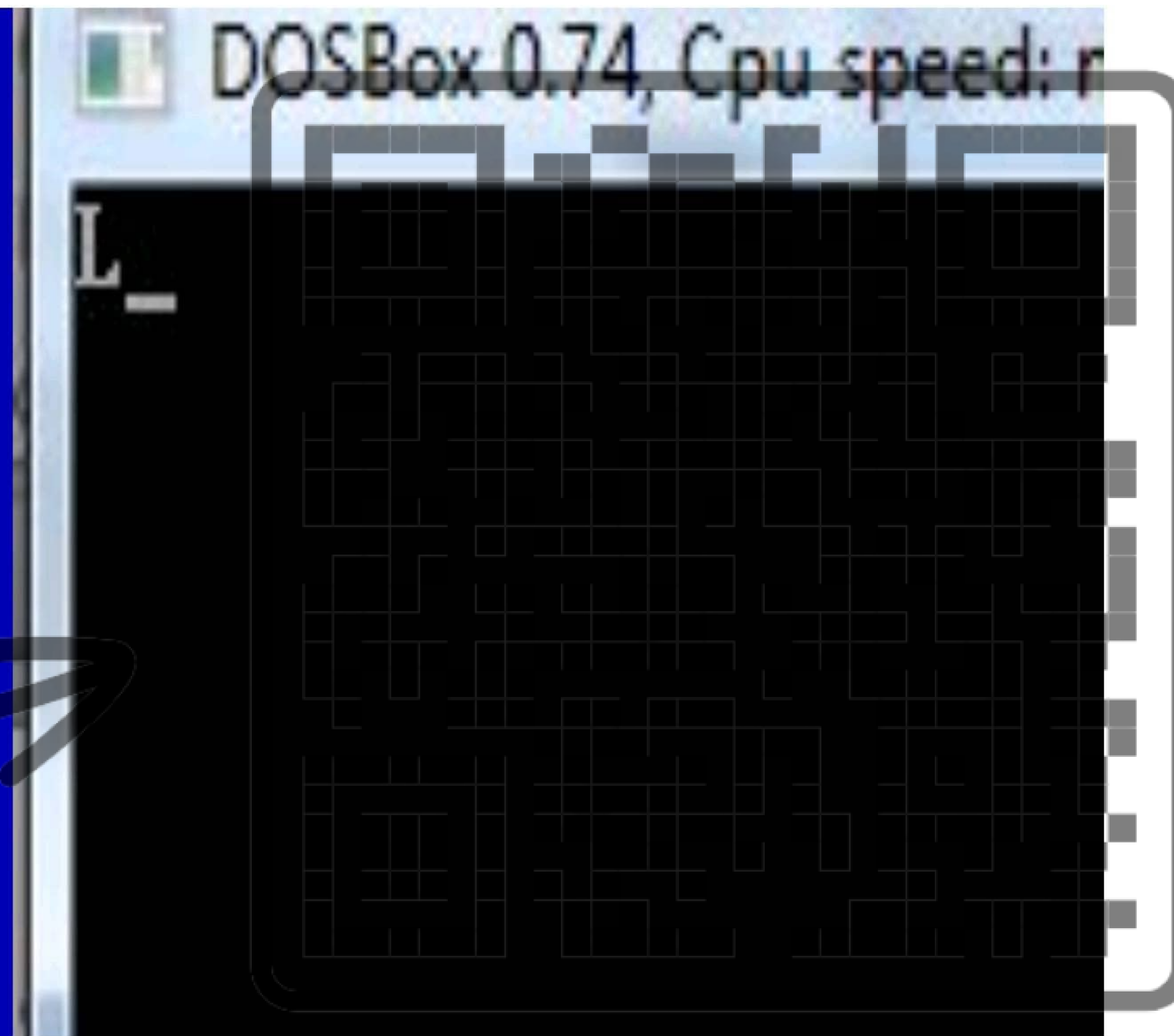


putch(), putchar()

- putch(), putchar(): Puts a single character on screen. Both the functions are exactly same.

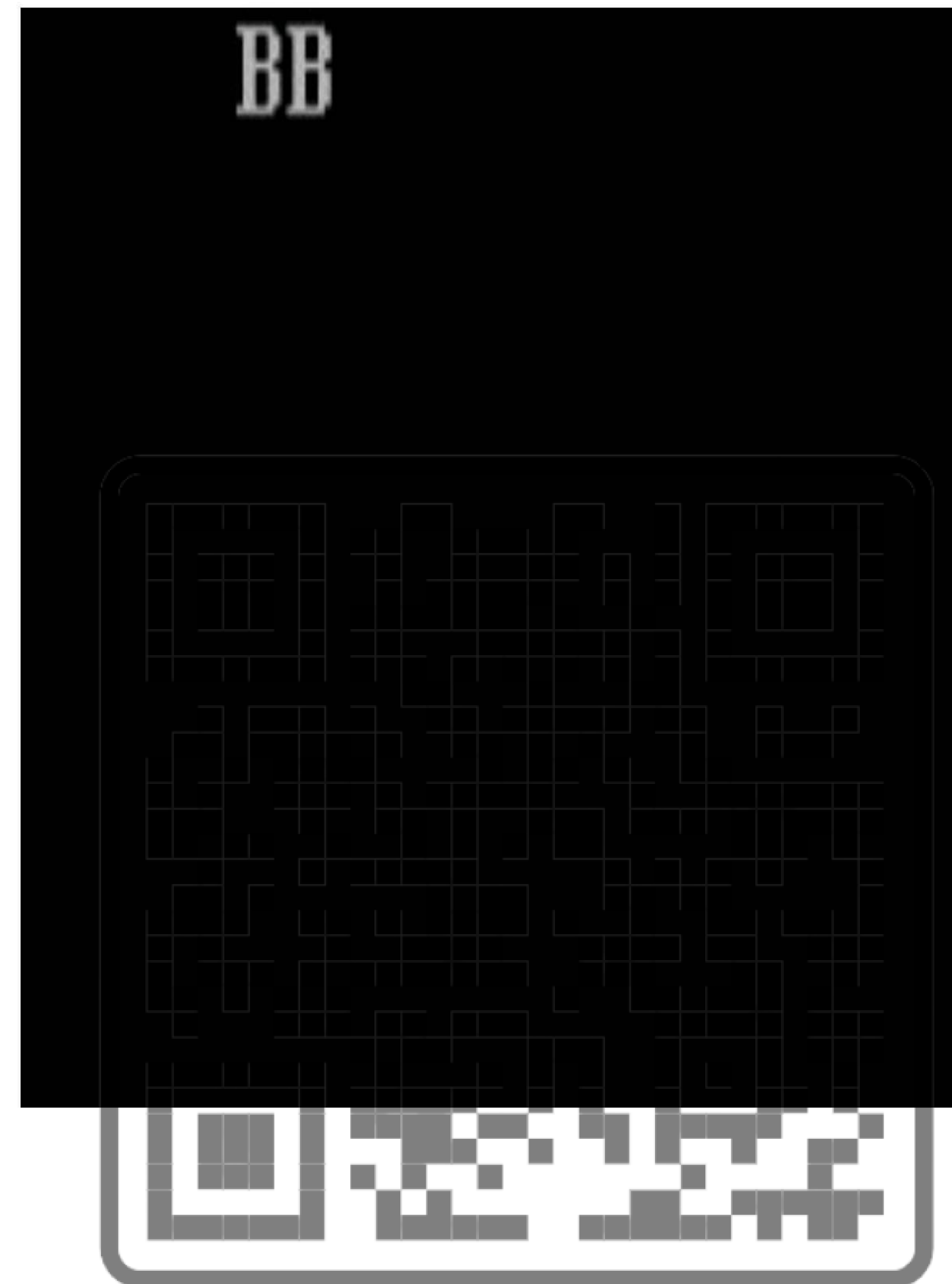
```
#include<stdio.h>
#include<conio.h>
void main()
{
char letter;
clrscr();
letter='L';
putchar(letter);
getch();
}
```

JOIN
FOR
MORE!!!




```
[■]  
#include<stdio.h>  
#include<conio.h>  
void main()  
{  
char ch;  
clrscr();  
ch=getche();  
putchar(ch);  
getch();  
}
```

JOIN
FOR
MORE!!!



gets():

- gets(): Receives string of characters and put it in a char array. A string is entered from the keyboard, will terminate with a newline character.

JOIN FOR MORE!!!

```
#include<stdio>
#include<conio>
void main()
{
char ch [5];
gets(ch);
puts(ch);
getch();
}
```

SDL_app

this is a string
this is a string



puts():

- puts(): It puts backs them to screen

JOIN FOR MORE!!!

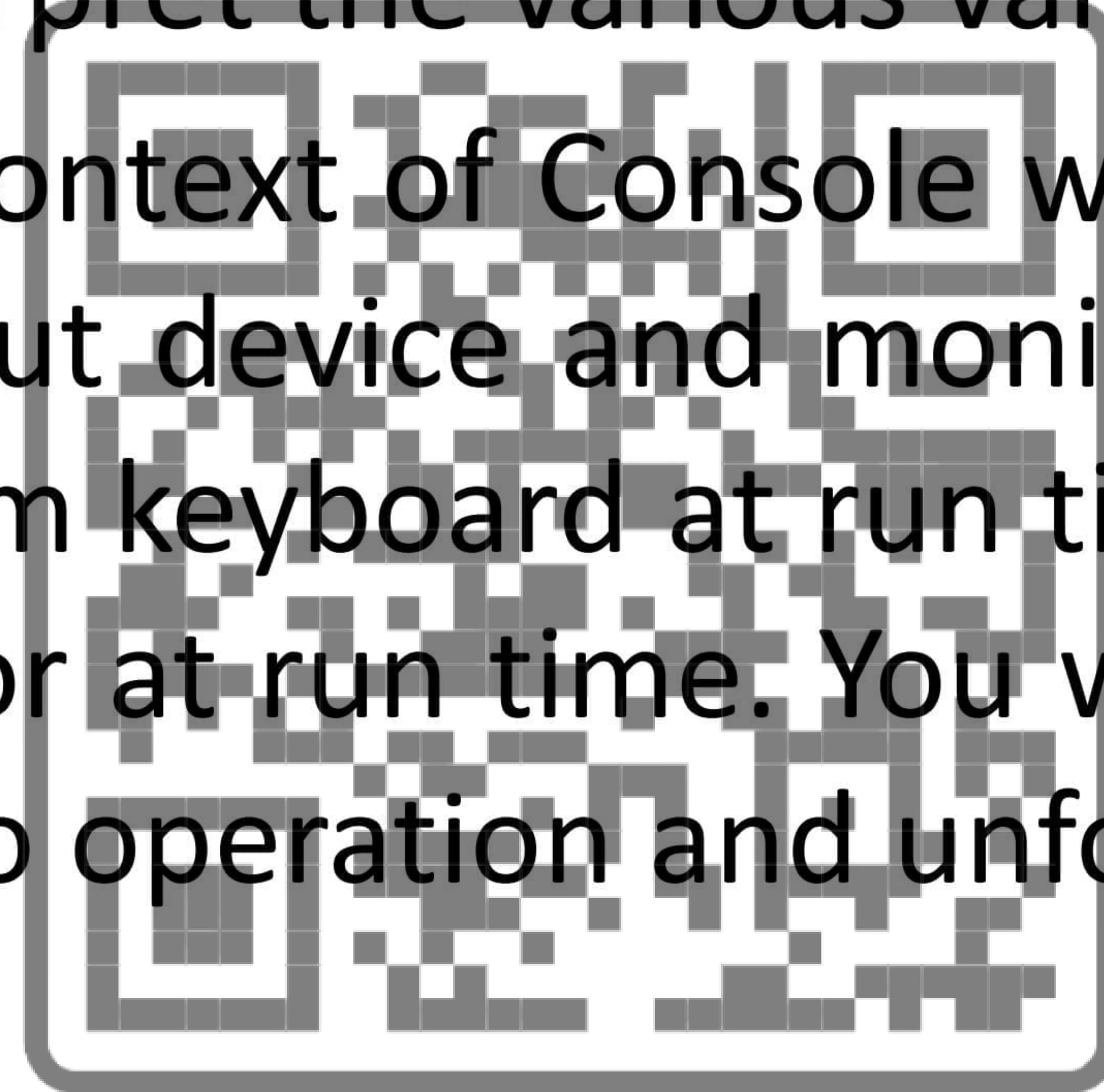
```
#include<stdio.h>
#include<conio.h>
void main()
{
char greeting[30];
clrscr();
puts("enter greeting");
gets(greeting);
puts("hi");
puts(greeting);
getch();
}
```

```
enter greeting
hi
hi
hi
```



Formatted I/O function

- The scanf() and printf() fall under the category of formatted input and output functions respectively. These functions instruct how to compiler interpret the various values.
- Console I/O operations: In context of Console what we need to learn, a keyboard is a input device and monitor is output device. How to read data from keyboard at run time and how to produce output on monitor at run time. You want to do in two ways one is formatted I/o operation and unformatted I/O operation.



Printf

- Printf is used for printing in the monitor. And scanf is used for reading from keyboard
 - The printf() Function: This function is most versatile way to display output on screen. When using printf(), you need to include the standard input/output header file, `STDIO.H`.
 - `int printf(format-string [,arguments,...]);`
 - Format string can have: Literal text, conversions pecifier, escape sequences.
- The format-string is required; however, arguments are optional.
For each argument, there must be a conversion specifier.




```
#include<stdio.h>
#include<conio.h>
int main()
{
int a,b,c;
clrscr();
a=10;
b=20;
c=a+b;
printf("sum of a and b is c:");
getch();
}
```

JOIN
FOR
MORE!!!

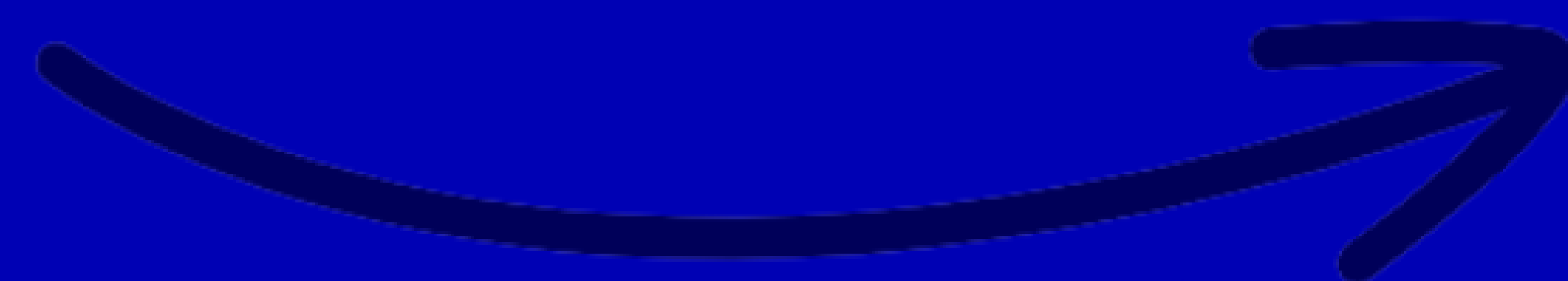


sum of a and b is c: _




```
#include<stdio.h>
#include<conio.h>
int main()
{
int a,b;
int c;
clrscr();
a=10;
b=20;
c=a+b;
printf("sum of %d and %d is: %d",a,b,c);
getch();
}
```

**JOIN
FOR
MORE!!!**



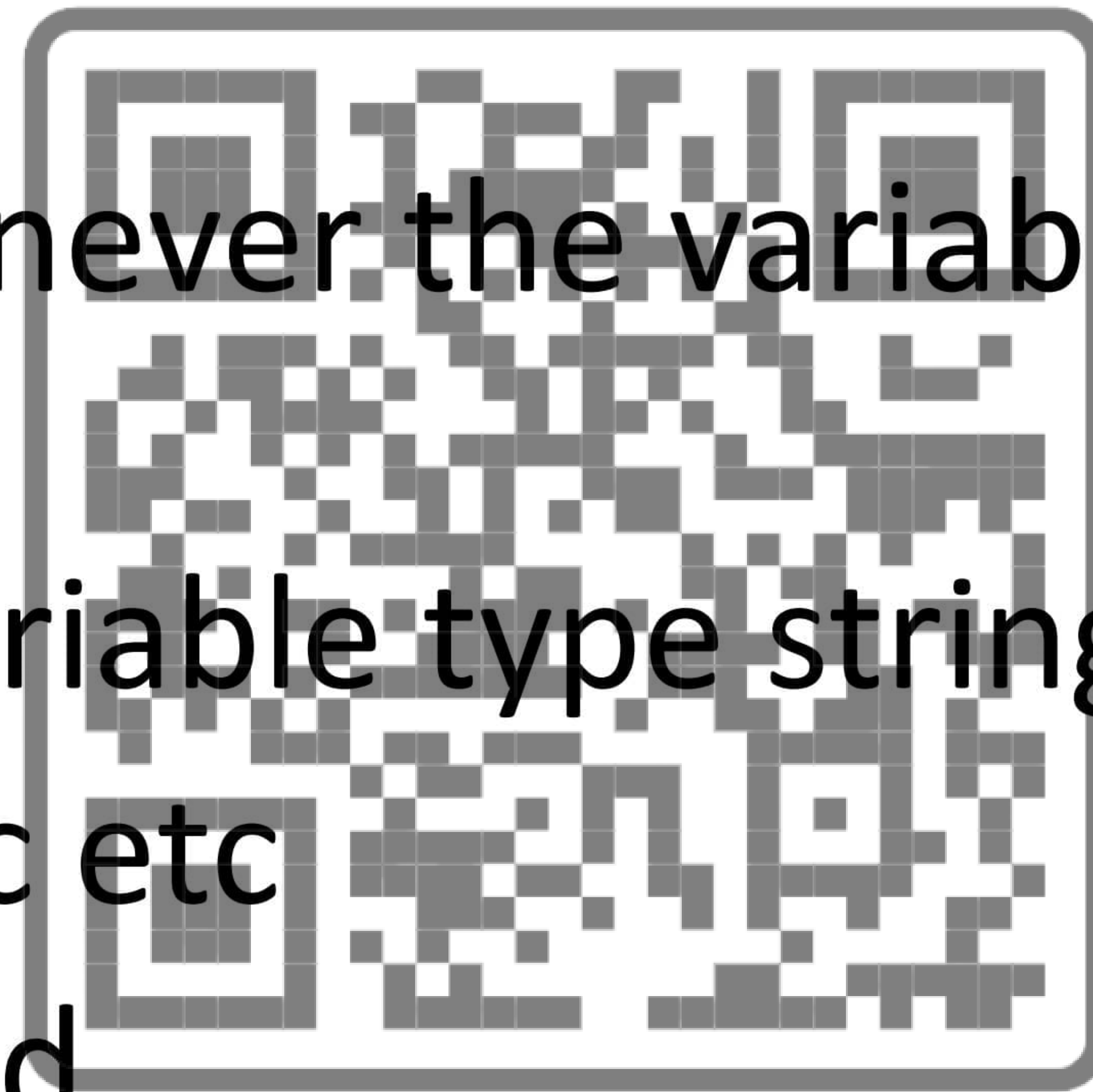
sum of 10 and 20 is: 30_



What is %d?

- %d is supposed to be used whenever the variable type int.
- Likewise %c is used whenever the variable type char
- And %s is used when, variable type string.
- Literal text: sum of a, b, c etc
- conversions specifier :%d
- escape sequences. \n etc

JOIN
FOR
MORE!!!



For demo purpose I changed float instead of int but I didn't change format specifier lets see what happen

```
#include<stdio.h>
#include<conio.h>
int main()
{
float a,b;
float _c;
clrscr();
a=10;
b=20;
c=a+b;
printf(" sum of %d and %d is: %d",a,b,c);
getch();
}
```

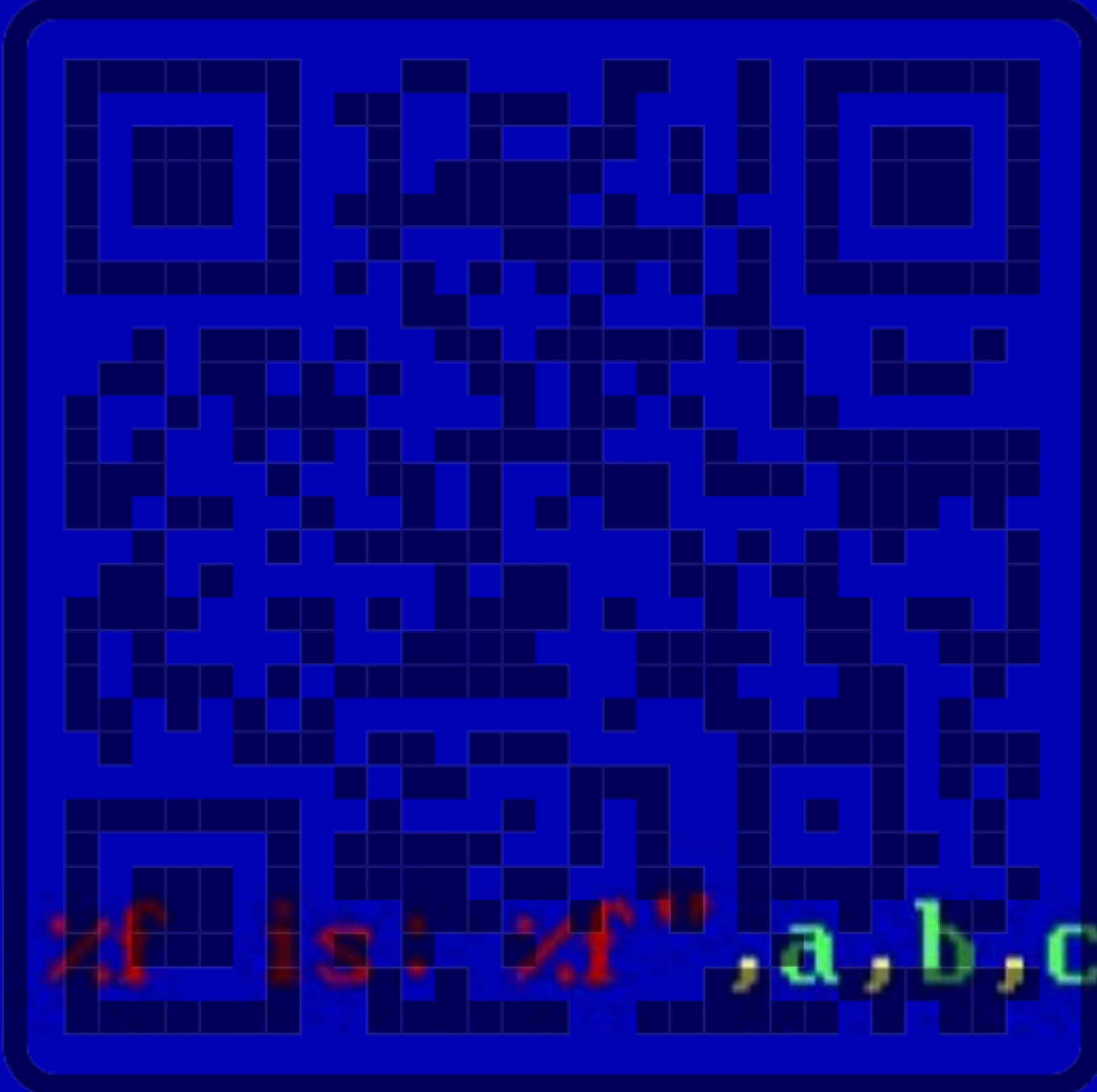
sum of 0 and 0 is: 0



Now I change format specifier %f lets see

```
#include<stdio.h>
#include<conio.h>
int main()
{
float a,b;
float c;
clrscr();
a=10;
b=20;
c=a+b;
printf(" sum of %f and %f is: %f",a,b,c);
getch();
}
```

JOIN FOR MORE!!!



```
sum of 10.000000 and 20.000000 is: 30.000000_
```



Advance level formating

- I can do here let us say

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int a,b;
6     float c;
7     a = 10;
8     b = 20;
9     c = a+b;
10    printf("Sum of %10d and %5d is %f\n", a, b, c);
11    printf("\n");
12 }
13
```

JOIN FOR MORE!!!



- What is %10d and %5d. See



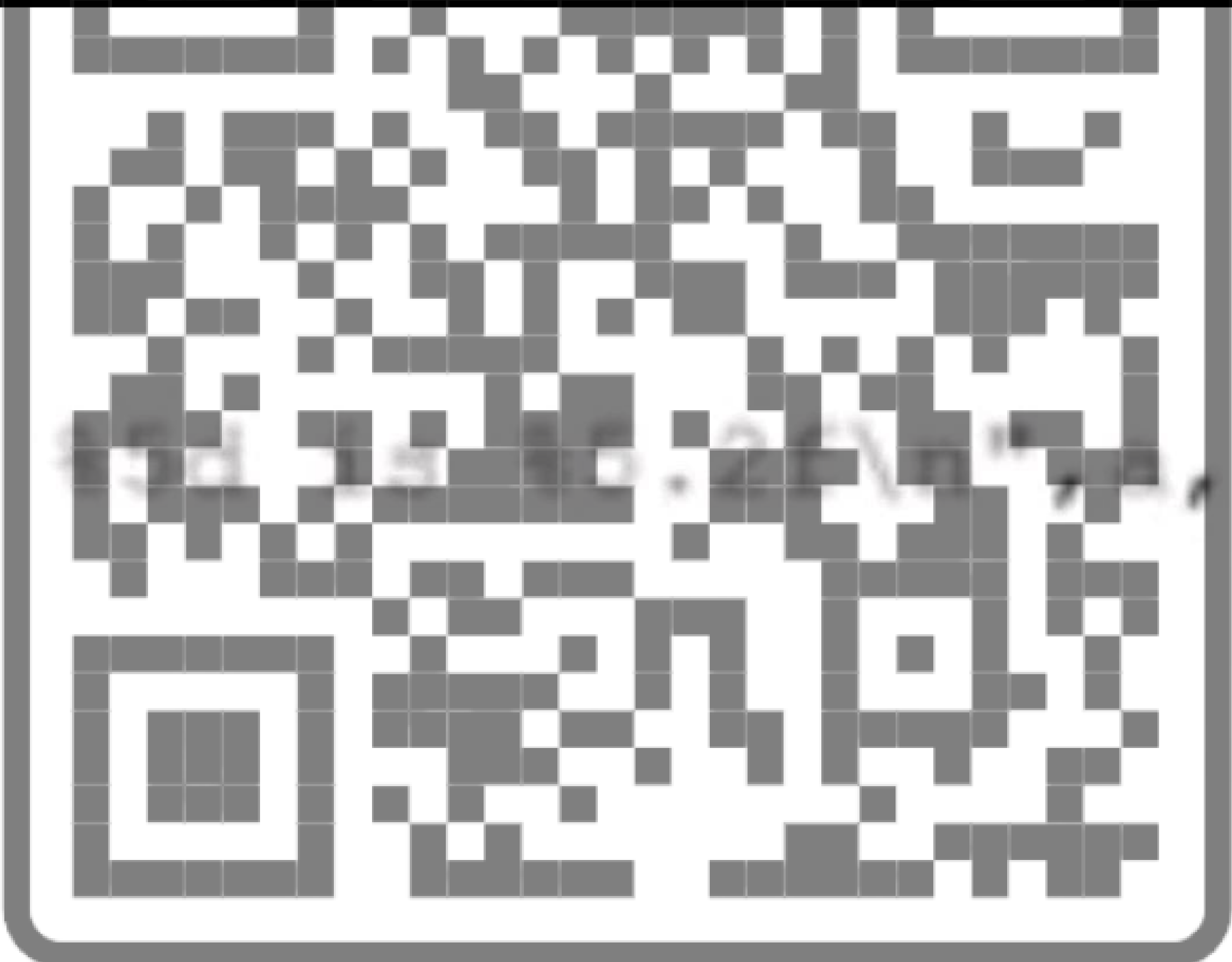
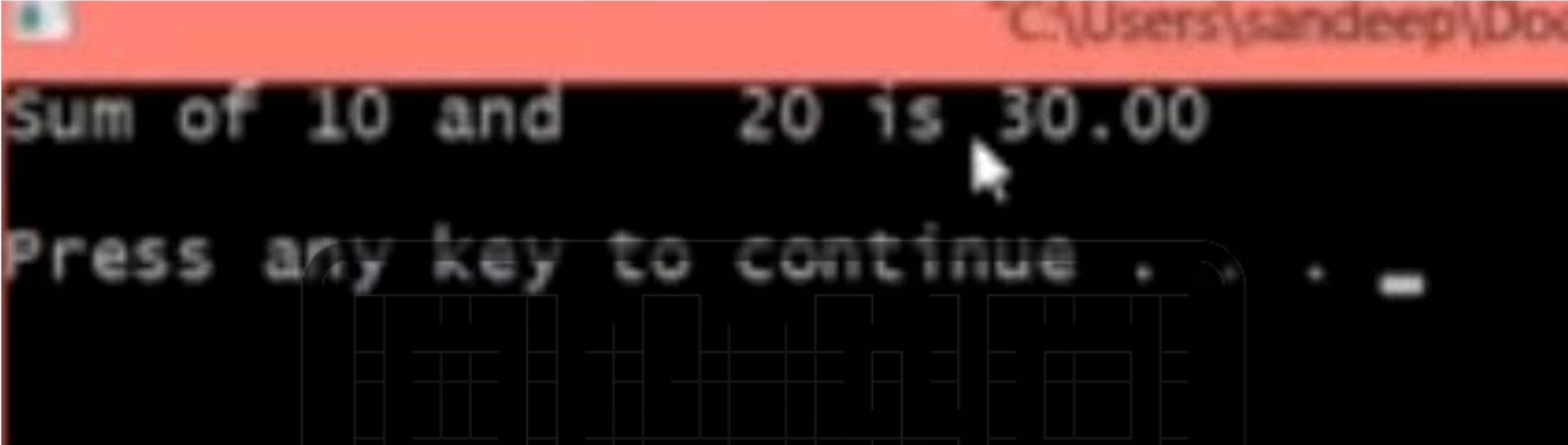
- 10 digit space create as well 5 digit space create



Similarly what happen if put % 5.2f

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int a,b;
6     float c;
7     a = 10;
8     b = 20;
9     c = a+b;
10    printf("Sum of %d and %d is %5.2f\n",a,b,c);
11    printf("\n");
12 }
13
```

JOIN FOR MORE!!!



What this mean? Total 5 digit and put decimal after two digit

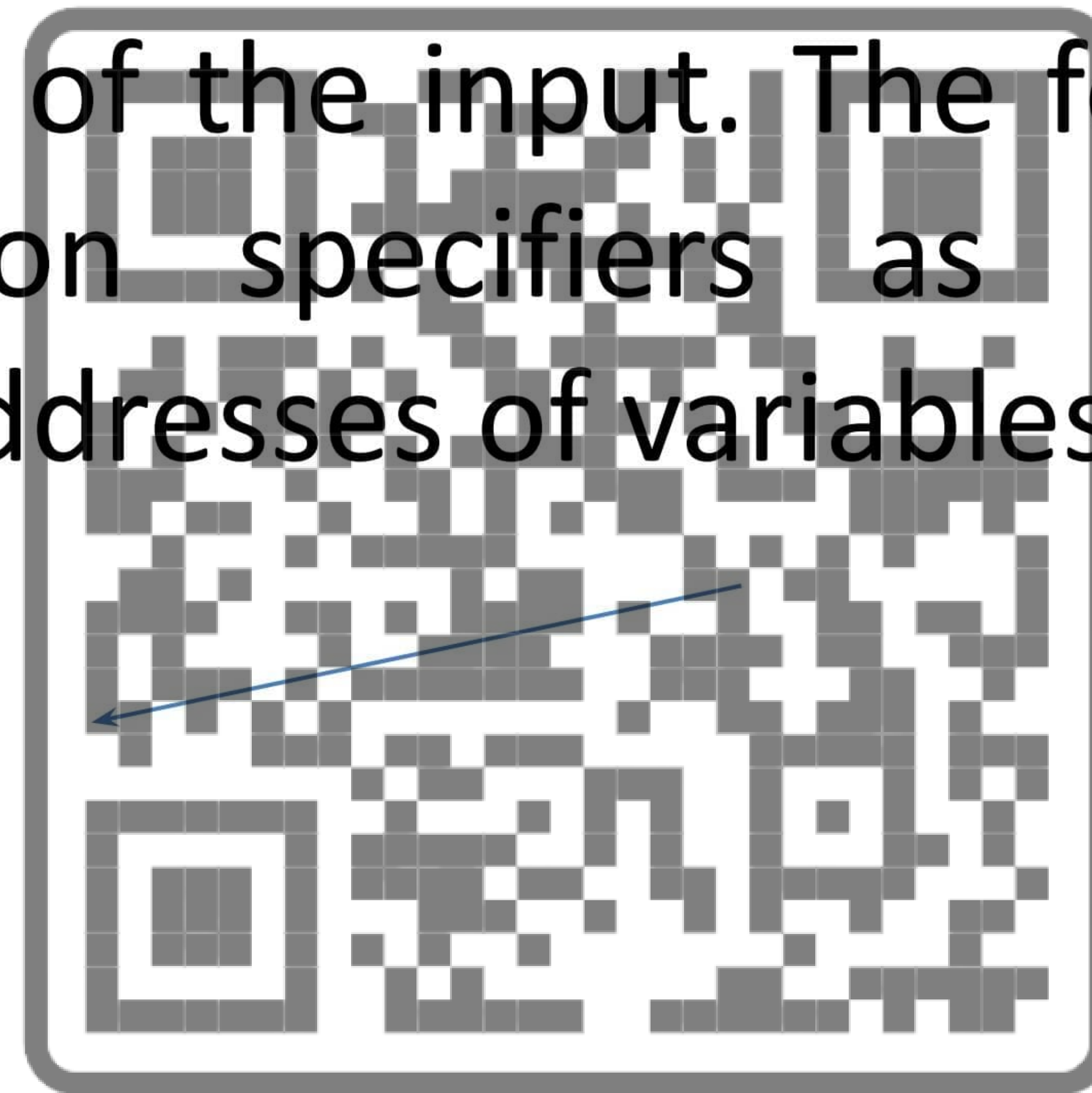


scanf

- The scanf() function reads data from the keyboard according to a specified format and assigns the input data to one or more program variables. Like printf(), scanf() uses a format string to describe the format of the input. The format string utilizes the same conversion specifiers as the printf() scanf("format string", list of addresses of variables);



&a




```
#include<stdio.h>
#include<conio.h>
```

```
int main()
```

```
{
```

```
int a,b;
```

```
int c;
```

```
clrscr();
```

```
printf("enter numbers which you want to add:");
```

```
scanf("%d",&a);
```

```
scanf("%d",&b);
```

```
c=a+b;
```

```
printf("sum of %d and %d is: %d",a,b,c);
```

```
getch();
```

```
}
```

■ JOIN
FOR
MORE!!!



```
enter numbers which you want to add:
```

```
20
```

```
30
```

```
sum of 20 and 30 is: 50_
```



The reading will be terminated, when scanf encounters an invalid mismatch of data or character that is not valid for value being read.

```
Enter the two numbers:10 A
Sum of 10.000000 and 0.000000 is 10.000000
Press any key to continue . . .
```

MORE!!!

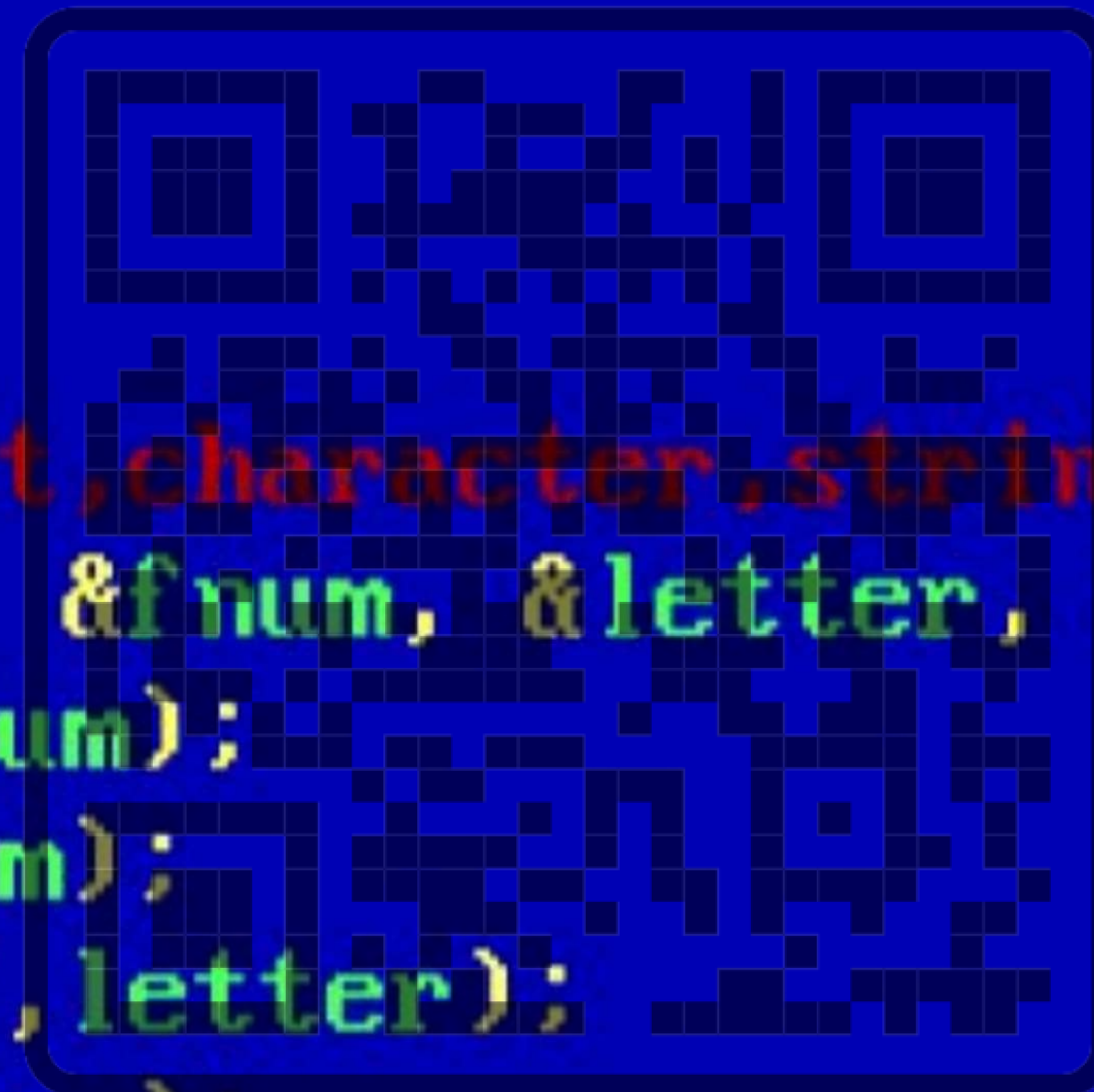
function. A single scanf() can input more than one value if you include multiple conversion specifiers in the format string and variable names.




```
void main()
{
    int num;
    float fnum;
    char letter, name[15];
    clrscr();
    printf("type Integer, float, character, string\n");
    scanf("%d %f %c %s", &num, &fnum, &letter, name);
    printf("\nInteger = %d", num);
    printf("\nfloat = %f", fnum);
    printf("\ncharacter = %c", letter);
    printf("\nstring = %s", name);
    getch();
}
```

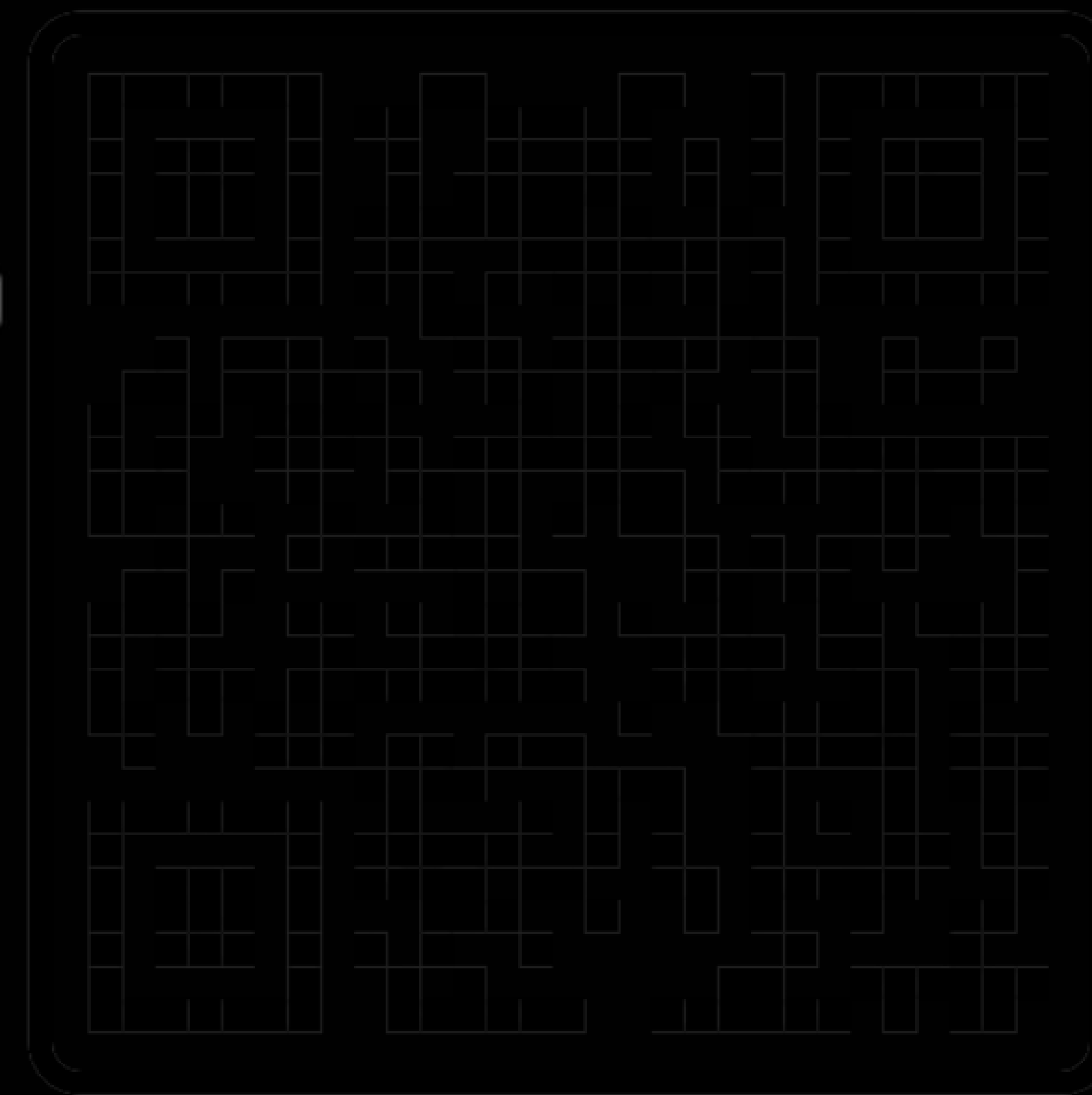
JOIN
FOR

MORE!!!




```
type Integer, float, character, string  
10  
4.5  
a  
sana mahmood
```

```
Integer = 10  
float = 4.500000  
character = a  
string = sana
```



Chapter # 07 Past Paper Questions with Answers 2nd Year

Define switch and break statements with an example (2018)

Switch Statements:

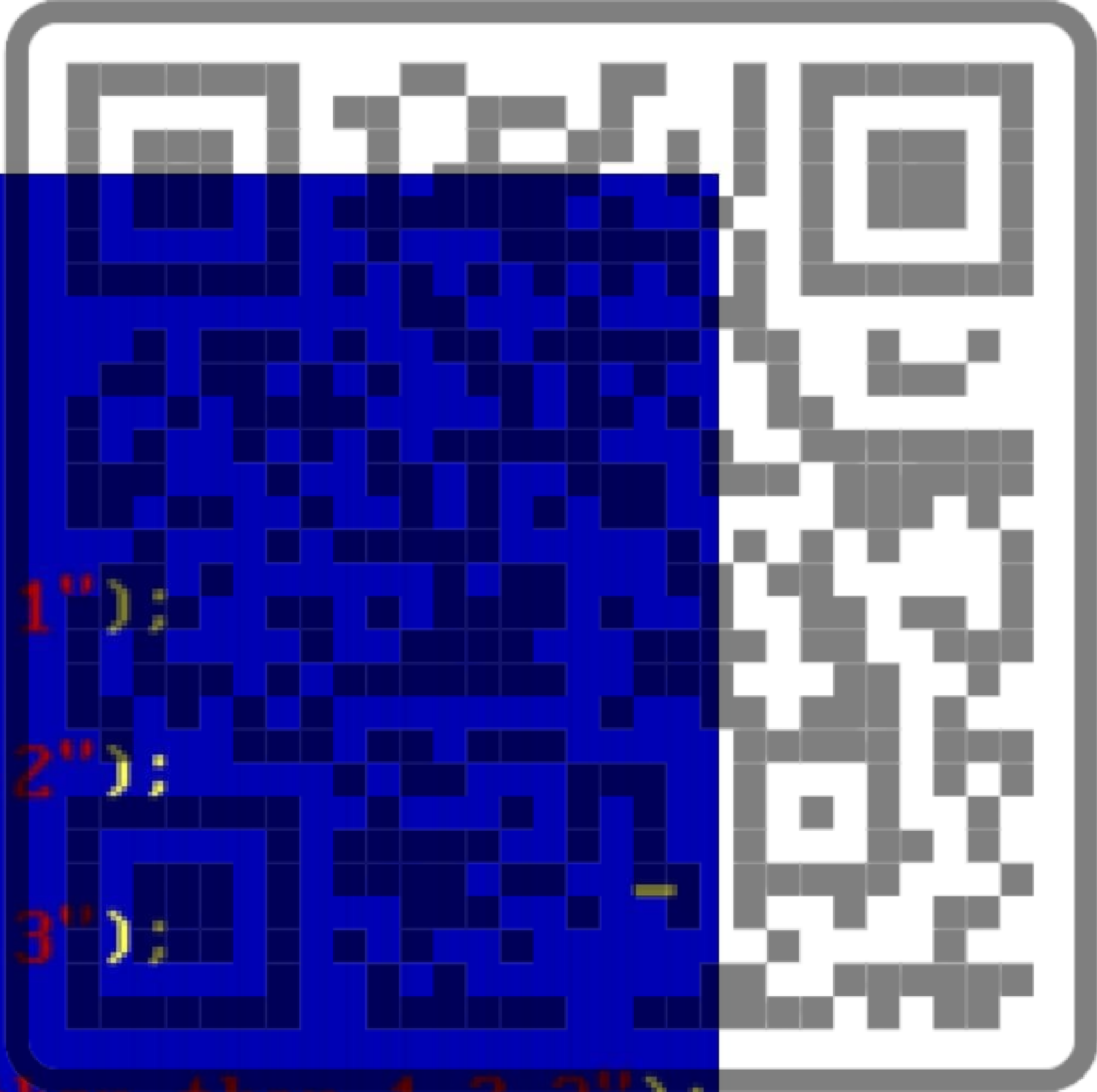
An alternative is the switch statement. The switch statement is used to select one of the several alternatives, such as menu options. The switch statement is especially useful when the selection is based on the value of a single variable (called controlling variable). The value of this variable may be of type int or char, but not of type float. A variable is tested against a list of integer or character constants. When the value matches a case condition, the statement(s) following the case is executed.

Break Statements:

The break statement at the end of each case statement sends control to the end of the switch, so once a case is performed, the remaining case statements are ignored and the switch ends.

JOIN FOR MORE!!!

```
{
int x=2;
clrscr();
switch (x)
{
case 1: printf("choice is 1");
break;
case 2: printf("choice is 2");
break;
case 3: printf("choice is 3");
break;
default: printf("choice other than 1,2,3");
}
getch();
}
```



How does else-if ladder differ from switch statements? (2017)

The switch statement also differs from else-if ladder, in that switch can only test for equality, whereas the if conditional expression can be of any type.

What is meant by control structure? Discuss selection control structure. (2017)

Control Structure:

A control structure is a block of programming that analyses variables and chooses a direction in which to go based on given parameters. The term *flow control* details the direction the program takes (which way program control "flows"). Hence it is the basic decision-making process in computing; flow control determines how a computer will respond when given certain conditions and parameters.



Selection Control Structure:

The selection control structures employ a number of conditions which lead to a selection of one out of several alternatives.

C provides three types of selection control structures.

- if statement
- if..else statements
- nested if statements
- switch statements

what is purpose of break and continue statement? (2017)

Difference Between break and continue

break	continue
A break can appear in both switch and loop (for, while, do) statements.	A continue can appear only in loop (for, while, do) statements.
A break causes the switch or loop statements to terminate the moment it is executed. Loop or switch ends abruptly when break is encountered.	A continue doesn't terminate the loop, it causes the loop to go to the next iteration. All iterations of the loop are executed even if continue is encountered. The continue statement is used to skip statements in the loop that appear after the continue.
The break statement can be used in both switch and loop statements.	The continue statement can appear only in loops. You will get an error if this appears in switch statement.
When a break statement is encountered, it terminates the block and gets the control out of the switch or loop.	When a continue statement is encountered, it gets the control to the next iteration of the loop.
A break causes the innermost enclosing loop or switch to be exited immediately.	A continue inside a loop nested within a switch causes the next loop iteration.

**JOIN
FOR
MORE!!!**

Similarities Between `break` and `continue`



COMPILED BY SANA MAHMOOD 2

Chapter # 07 Past Paper Questions with Answers 2nd Year

Both `break` and `continue` statements in C programming language have been provided to alter the normal flow of program.

Explain the uses of nested if-else statement. (2015)

Nested if-else statement

There comes situations in real life when we need to make some decisions and based on these decisions, we decide what should we do next. Similar situations arises in programming also where we need to make some decisions and based on these decision we will execute the next block of code.

Nested if-else statements test for multiple conditions by placing if-else statements inside



if-else statements. When a condition is evaluated as true, the corresponding statements are executed and the rest of the structure is skipped. i.e, we can place an if statement inside another if statement. This structure is referred to as nested if-else statement.

**JOIN
FOR
MORE!!!**



COMPILED BY SANA MAHMOOD 3

Chapter # 07 Past Paper Questions with Answers 2nd Year

Write a program that reads three numbers and prints the largest number. (2015)




```

void main()
{
int num1,num2,num3;
clrscr();
printf("enter number 1=");
scanf("%d",&num1);
printf("enter number 2=");
scanf("%d",&num2);
printf("enter number 3=");
scanf("%d",&num3);
if(num1>num2 && num1>num3)
{
printf("number 1 is greater than num1 and num2");
}
else if(num2>num1 && num2>num3)
{
printf("number 2 is greater than num1 and num3");
}
else if(num3>num1 && num3>num2)
{
printf("number 3 is greater than num1 and num3");
}
getch();
}

```

JOIN FOR MORE!!!




Write a program that uses switch statement. (2014)

```
{
int i;
clrscr();
printf("enter a number between 1 and 4:");
scanf("%d",&i);
switch (i)
{
case 1: printf("one");
break;
case 2: printf("two");
break;
case 3: printf("three");
break;
case 4: printf("four");
break;
default:
printf("unrecognised number");
}
getch();
}
```

COMPILED BY SANA MAHMOOD 5



- Decision making in programming:
- Decision making is the process of making choices.
- There comes situations in real life when we need to make some decisions and based on these decisions, we decide what should we do next. Similar situations arises in programming also where we need to make some decisions and based on these decision we will execute the next block of code.
- Once you have weighed all the evidence, you are ready to select the alternative that seems to be best one for you. You may even choose a combination of alternatives.
- Decision making structures require that the programmer specifies one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

JOIN

FOR

MORE!!!



- Selection control structures:

- The selection control structures employs a number of conditions which lead to a selection of one out of several alternatives.
- C provides three types of selection control structures.

1. If selection structures
2. If/else selection structures
3. The switch selection structures

**JOIN
FOR
MORE!!!**

- [if statement](#)
- [if..else statements](#)
- [nested if statements](#)
- [switch statements](#)



If selection structures:



if statement

if statement is the most simple decision making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e if a certain condition is true then a block of statement is executed otherwise not.

Syntax:

```
if(condition) { // Statements to execute if // condition is true }
```

Here, **condition** after evaluation will be either true or false. if statement accepts boolean values – if the value is true then it will execute the block of statements below it otherwise not. If we do not provide the curly braces '{' and '}' after if(condition) then by default if statement will consider the first immediately below statement to be inside its block.



Relational Operators

Operation	Description	Examples
<	Less than	If (hours<40)
<=	Less than or equal to	If(cost<=limit)
>	Greater than	If(hours>40)
>=	Greater than or equal to	If(cost>=limit)
==	Equal to	If(salary==10000)
!=	Not equal to	If(count!=1)

JOIN
FOR
MORE!!!



- Example: if hours greater than 40 ,then print overtime otherwise , nothing is displayed

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int hrs;
printf("enter hours");
scanf("%d",&hrs);
if (hrs>40)
printf("overtime\n",hrs);
getch();
}
```

JOIN
FOR
MORE!!!



Multiple if statements:

The basic if command performs one statement . If you want to perform more than one statements when condition is true, you must use a second level of braces.

Example:

If(condition)

{

Statement 1;

Statement 2;

Statement 3;

}

JOIN FOR MORE!!

```
int a,b,result;
clrscr();
printf("enter value a:\n");
scanf("%d",&a);
printf("enter value b:\n");
scanf("%d",&b);
if(a<b)
{
result=a+5;
printf("result=%d",result);
}
getch();
}
```



```
enter value a:
20
enter value b:
30
result=25_
```



- Nested if statements:
- An if condition can be written as deeply as you need within the body of another if statement. This is called nesting of if's. for example an action is performed if two condition are true. The condition 1 is tested and then condition 2 is tested.
- Example:
- If(income>10000)
- If(status=='S')
- Tax_rate=0.35;

```
void main()
{
    int dependents;
    float income;
    clrscr();
    printf("enter your income:");
    scanf("%f",&income);
    printf("enter number of dependents:");
    scanf("%d",&dependents);
    if (income<5000)
    if (dependents>5)
    printf("you owe no income tax");
    getch();
}
```



enter your income: 4000
enter number of dependents: 8
you owe no income tax

- Example:

```
void main()
{
int a,b, answer;
clrscr();
a=10;
b=30;
answer=a+b;
if (answer==40)
printf("correct");
if (answer!=40)
printf("wrong");
getch();
}
```

correct

JOIN
FOR
MORE!!!



```
void main()
{
int a,b, answer;
clrscr();
printf("enter the value of a & b:");
scanf("%d %d",&a,&b);_
answer=a+b;
if (answer==40)
printf("correct");
if (answer!=40)
printf("wrong");
getch();
}
```

enter the value of a & b: 20 30
wrong_

If-else statement

- The if-else statement structure performs an indicated action only when condition is true, otherwise action is skipped. If-else structure performs certain action when the condition is true and some different action when condition is false.
- If (condition)
- Statement1;
- Else
- Statement2;



- Example:

```
{
int num1,num2;
clrscr();
printf("enter num1");
scanf("%d",&num1);
printf("enter num2");
scanf("%d",&num2);

if(num2==0)
printf("cannot divide by zero");
else
printf(" answer is: %d",num1/num2);
getch();
}
```

**JOIN
FOR
MORE!!!**



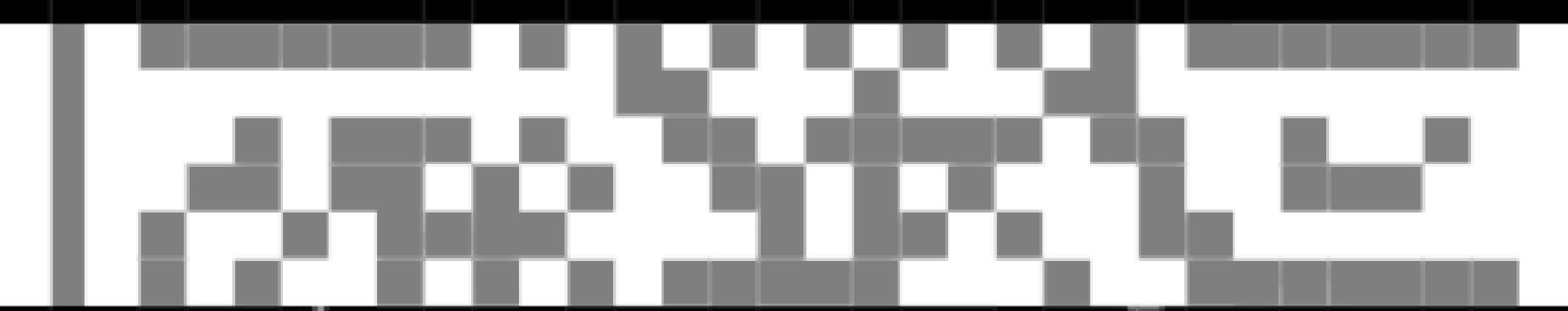

```
void main()
{
int num;
printf("enter a number");
scanf("%d",&num);_
if(num>0)
printf("number is positive");
else
printf("number is negative");

getch();
}
```

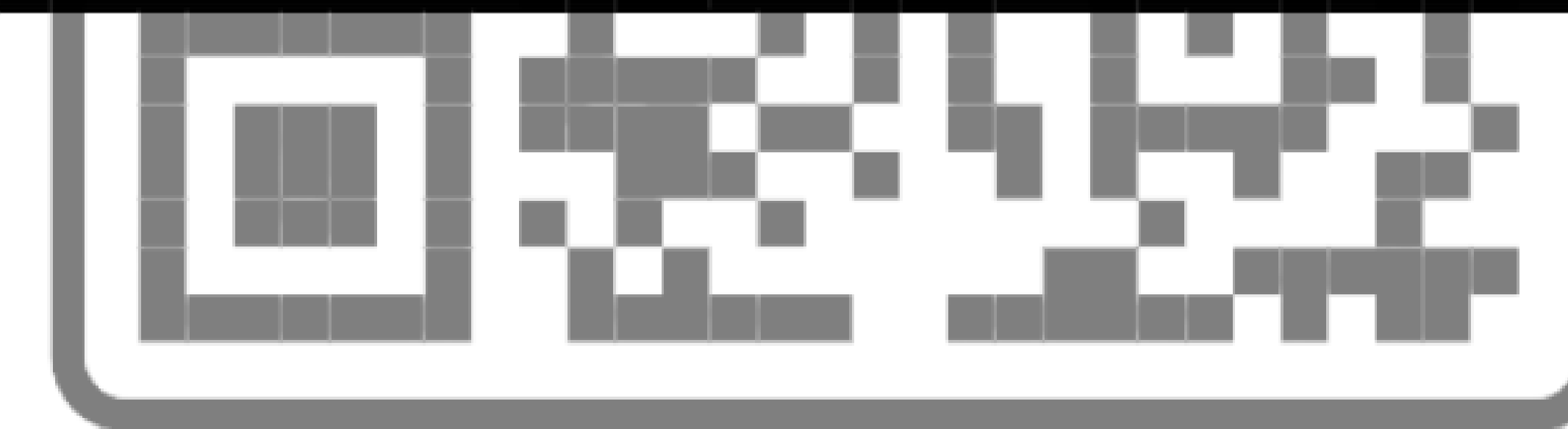
JOIN
FOR
MORE!!!



enter a number 8
number is positive



enter a number -2
number is negative_

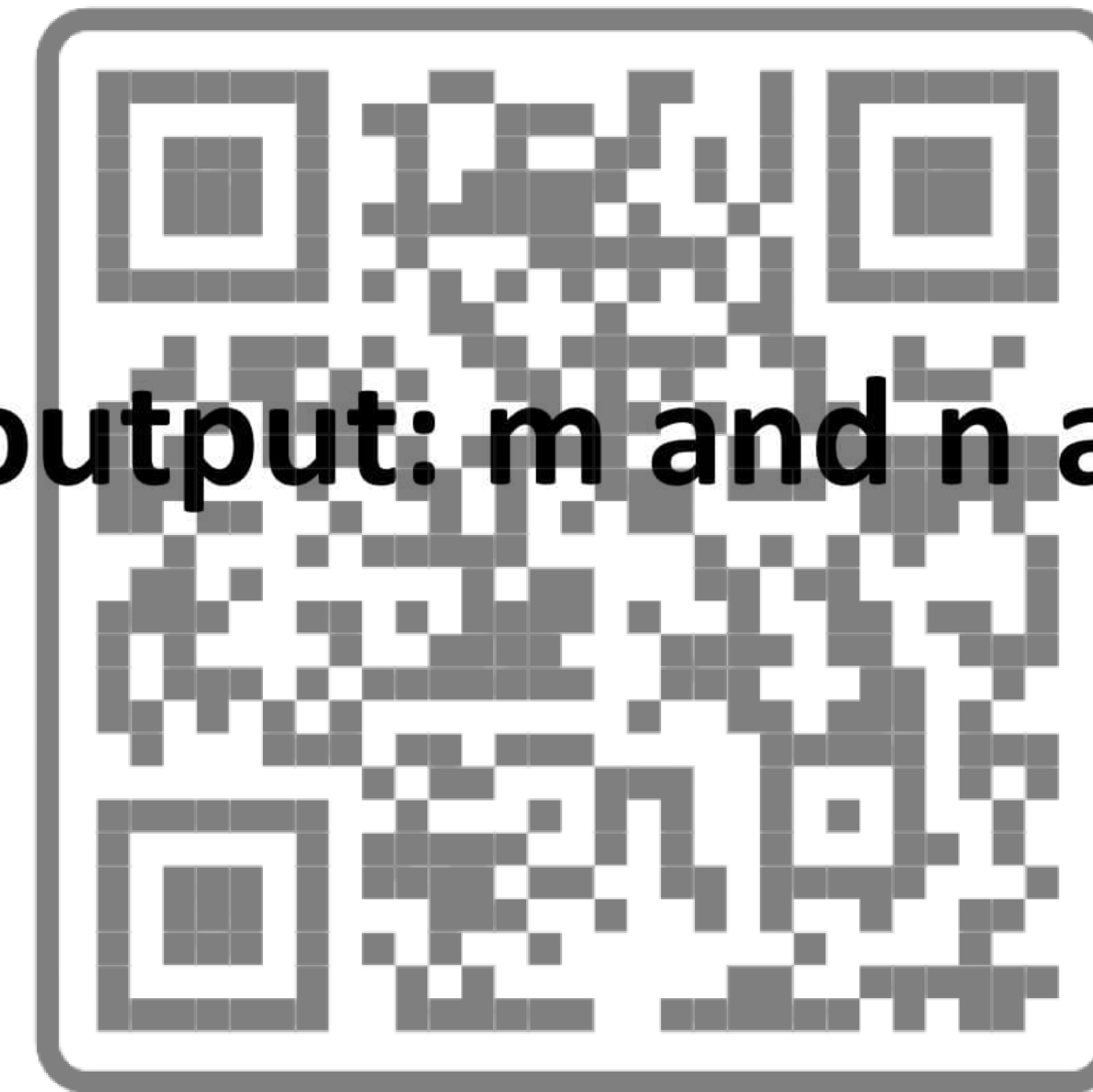


- Example:
- `#include <stdio.h>`
- `int main()`
- `{`
- `int m=40,n=20;`
- `if (m == n)`
- `{`
- `printf("m and n are equal");`
- `}`
- `else`
- `{`
- `printf("m and n are not equal");`
- `}`
- `}`

JOIN
FOR
MORE!!!

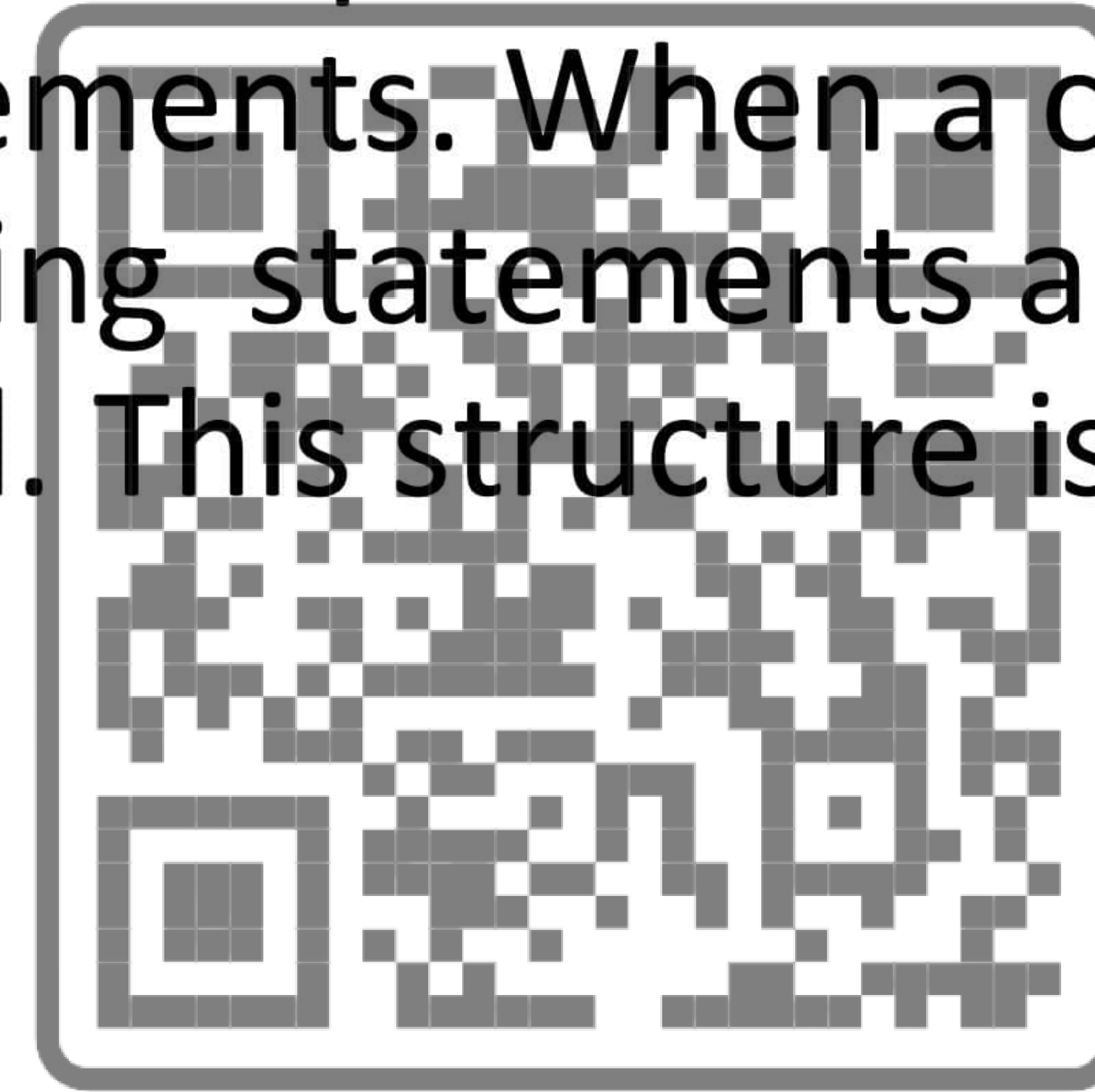


output: m and n are not equal



Nested if-else statement

- Nested if-else statements test for multiple conditions by placing if-else statements inside if-else statements. When a condition is evaluated as true, the corresponding statements are executed and the rest of the structure is skipped. This structure is referred to as nested if-else statement.



examples

```
clrscr();  
printf("input month number:");  
scanf("%d",&month);  
if(month==1)  
printf("January");  
else if(month==2)  
printf("february");  
else if(month==3)  
printf("March");  
else if(month==4)  
printf("April");  
else  
printf("error in month number");  
getch();
```

**JOIN
FOR
MORE!!!**



input month number: 6
error in month number

input month number: 4
April_




```
float percent;  
clrscr();  
printf("input percentage of marks:");  
scanf("%f",&percent);  
if(percent>=80)  
printf("Grade A-1\n");  
else if(percent>=70)  
printf("Grade A\n");  
else if(percent>=60)  
printf("Grade B\n");  
else if(percent>=50)  
printf("Grade C\n");  
else  
printf("You failed");
```

**JOIN
FOR
MORE!!!**



```
input percentage of marks: 75  
Grade A
```

```
input percentage of marks: 63.6  
Grade B
```

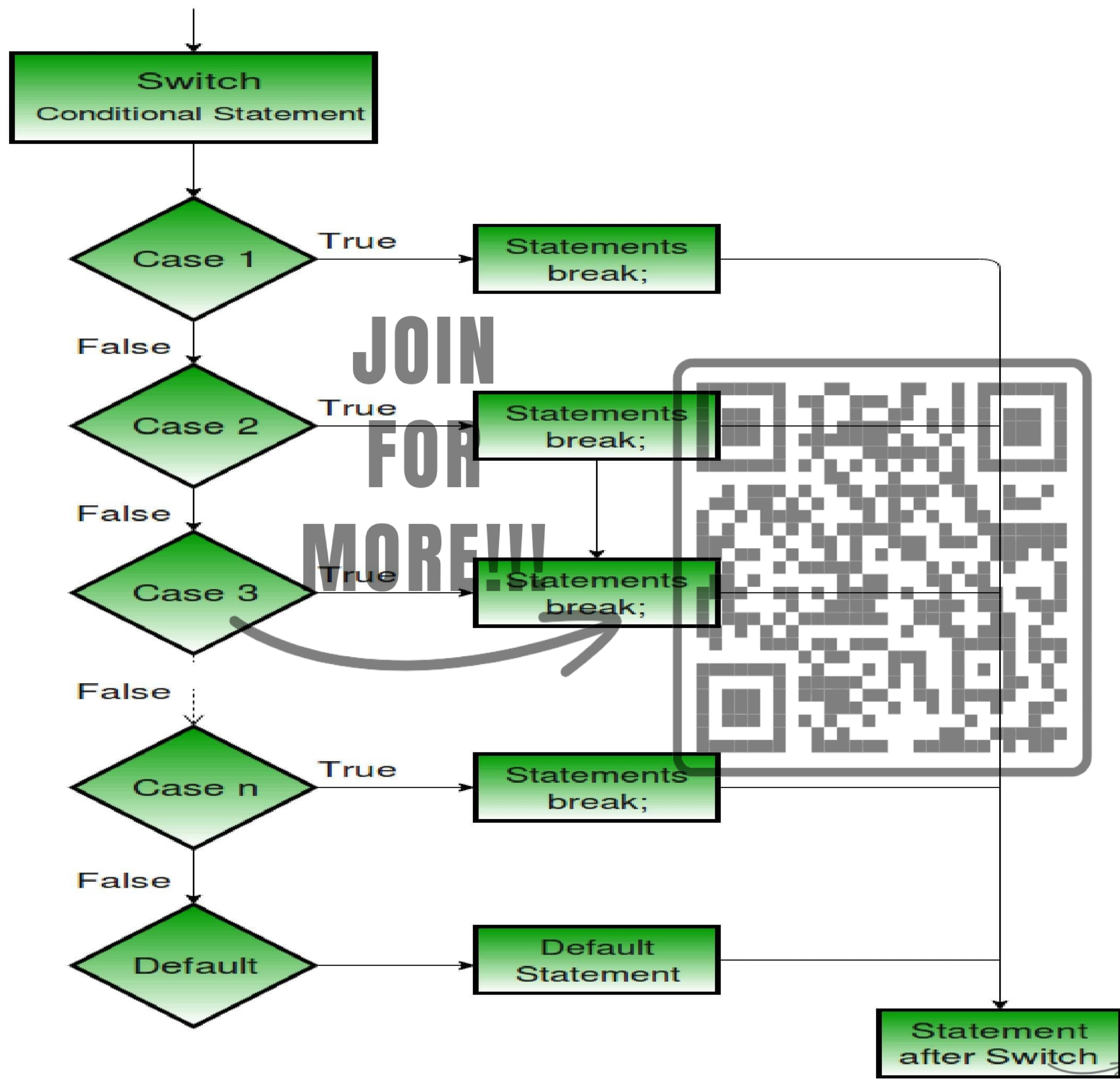
```
input percentage of marks: 45.5  
You failed_
```



The Switch Statements

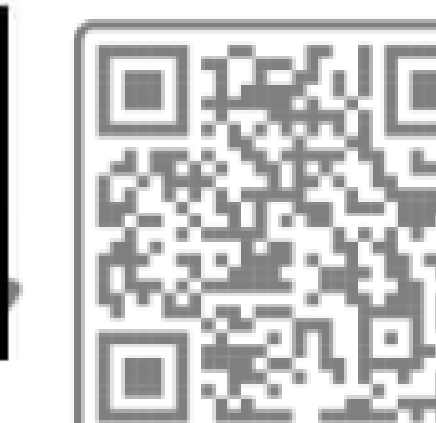
- The if-else statement is good for choosing among few(two to three) alternatives. It quickly become confusing when several alternatives are needed. An alternatives is the switch statement. The switch statement is used to select one of the several alternatives, such as menu options. The switch statement is especially useful when the selection is based on the value of a single variable (called controlling variable). The value of this variable may be of type int or char, but not of type float.
- A variable is tested against a list of integer or character constants. When the value matches a case condition , the statement(s)following the case is executed.
- The break statement at the end of each case statement sends control to the end of the switch , so once a case is performed , the remaining case statements are ignored and the switch ends.
- The default statement(s) is performed if no matches are found . The default is optional. If all matches fail and default is absent . No action takes place.





```
SWITCH(VARIABLE)
{
CASE (CONDITION)
//STATEMENT
BREAK;
CASE (CONDITION)
//STATEMENT
BREAK;
CASE(CONDITION):
//STATEMENT
BREAK;

DEFAULT:
//STATEMENT
}
```



POINTS TO REMEMBER

**JOIN
FOR
MORE!!!**

CASE DOESNT ALWAYS NEED TO HAVE ORDER 1 2 3.

IT CAN HAVE ANY INTEGER VALUE AFTER CASE KEYWORD.

CASE DOESNT NEED TO BE IN AN ASCENDING ORDER ALWAYS.

YOU CAN SPECIFY THEM IN ANY ORDER AS PER THE NEED OF THE PROGRAM.

CASE DOES NOT ACCEPT OTHER TYPES SUCH AS FLOAT OR STRING.



// Following is a simple program to demonstrate // syntax of switch.

```
{  
int x=2;  
clrscr();  
switch (x)  
{  
case 1: printf("choice is 1");  
break;  
case 2: printf("choice is 2");  
break;  
case 3: printf("choice is 3");  
break;  
default: printf("choice other than 1,2,3");  
}  
getch();  
}
```

**JOIN
FOR
MORE!!!**



choice is 2_




```
{
int i;
clrscr();
printf("enter a number between 1 and 4:");
scanf("%d",&i);
switch (i)
{
case 1: printf("one");
break;
case 2: printf("two");
break;
case 3: printf("three");
break;
case 4: printf("four");
break;
default:
printf("unrecognised number");
}
getch();
}
```

**JOIN
FOR
MORE!!!**



```
enter a number between 1 and 4: 1
one
```

```
enter a number between 1 and 4: 2
two_
```

```
enter a number between 1 and 4: 5
unrecognised number
```



Chap# 08 Past papers 2nd Year

Write the following in while () equivalent. (2018)

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a=3;
    clrscr();
    do
    {
        printf("%d\n",a);
        a=a+3;
    }
    while(a<=10);
    getch();
}
```

```
3
6
9
```

while () equivalent.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a=3;
    clrscr();
    while(a<=10)
    {
        printf("%d\n",a);
        a=a+3;
    }
    getch();
}
```

```
3
6
9
```



write a program that prints factorial of any inputted number.(2018)

EXAMPLE

$$4! = 4 \times 3 \times 2 \times 1 = 24$$

$$6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720$$

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i,num;
    int fac=1;
    printf("Enter a number:");
    scanf("%d",&num);
    for(i=1;i<=num;i++)
        fac=fac*i;
    printf("factorial of %d is: %d",num,fac);
    getch();
}
```



```
Enter a number: 4
factorial of 4 is: 24
```

What is the main difference between while and do-while loop?
(2017)



Chapter # 08 Past Paper with answers 2nd year

Comparison between while and do while statements.

while	do...while
This is the top tested loop	This is the bottom tested loop
The condition is first tested, if the condition is true then the block is executed until the condition becomes false.	It executes the body once, after it checks the condition, if it is true the body is executed until the condition
Loop will not be executed if the condition is false.	Loop will be executed atleast once even though the condition is false.

JOIN
FOR
MORE!!!

What is output of the following program? (2017)

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
for(int a=1;a<5;a++)
{
for(int b=1; b<5;b++)
printf("%d",b);
printf("\n");
}
getch();
}
```



Output :

```
1234
1234
1234
1234
```

Write a program that swap two imputed numbers. (2017)

```
#include<conio.h>
void main()
{
    int a,b,c;
    clrscr();
    printf("enter values of a & b:");
    scanf("%d%d",&a,&b);
    c=a;
    a=b;
    b=c;
    printf("\n value of a is :%d",a);
    printf("\n value of b is :%d",b);
    getch();
}
```



Chapter # 08 Past Paper with answers 2nd year

enter values of a & b: 20 10
value of a is :10
value of b is :20_



Write the equivalent code of the following statement in while loop format. (2017)

```
#include<conio.h>
void main()
{
    int a;
    clrscr();
    for( a=1;a<=100;a++)
        printf("%d\n",a*a);
    getch();
}
```



while () equivalent.

P REPAIED BY SANA MAHMOOD 5

Chapter # 08 Past Paper with answers 2nd year

JOIN
FOR

MORE!!!

```
#include<conio.h>
void main()
{
    int a=1;
    clrscr();
    while(a<=100)
    {
        printf("%d\n",a*a);
        a++;
    }
    getch();
}
```



define loop with an example:

In computer programming, a **loop** is a sequence of instructions that is continually repeated until a certain condition is reached. Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter

has reached a prescribed number

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int count;
    clrscr();
    for (count=1; count<=10; count++)
        printf("%3d",count);_
    getch();
}
```

P

REPAIRED BY SANA MAHMOOD 6

Chapter # 08 Past Paper with answers 2nd year .

JOIN
FOR
MORE!!!



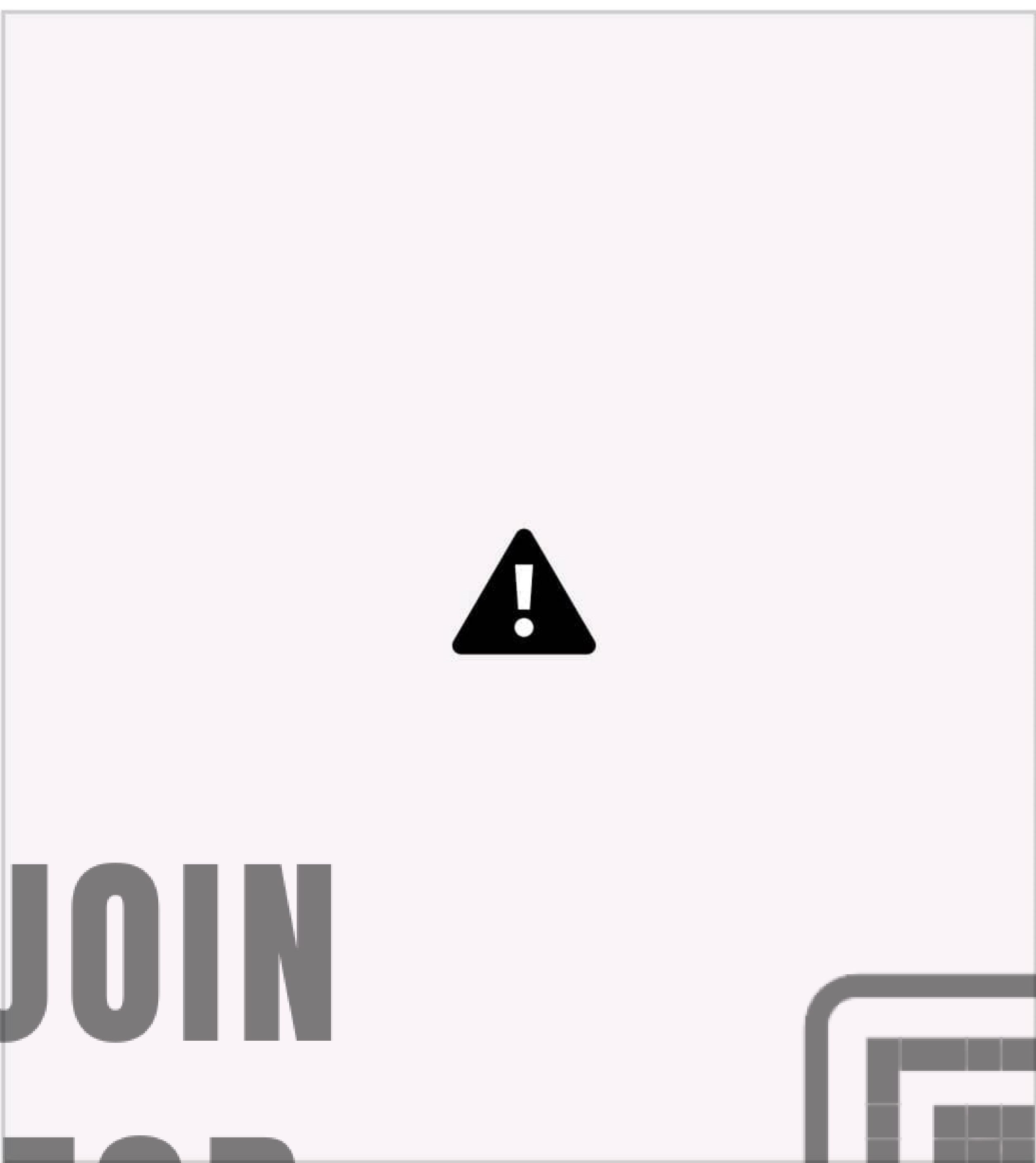
write a program in c language to print numbers from 4 to 9 and their squares using while loop. (2016)



P

Chapter # 08 Past Paper with answers 2nd year

write the output of the following program. (2016)




JOIN FOR MORE!!!



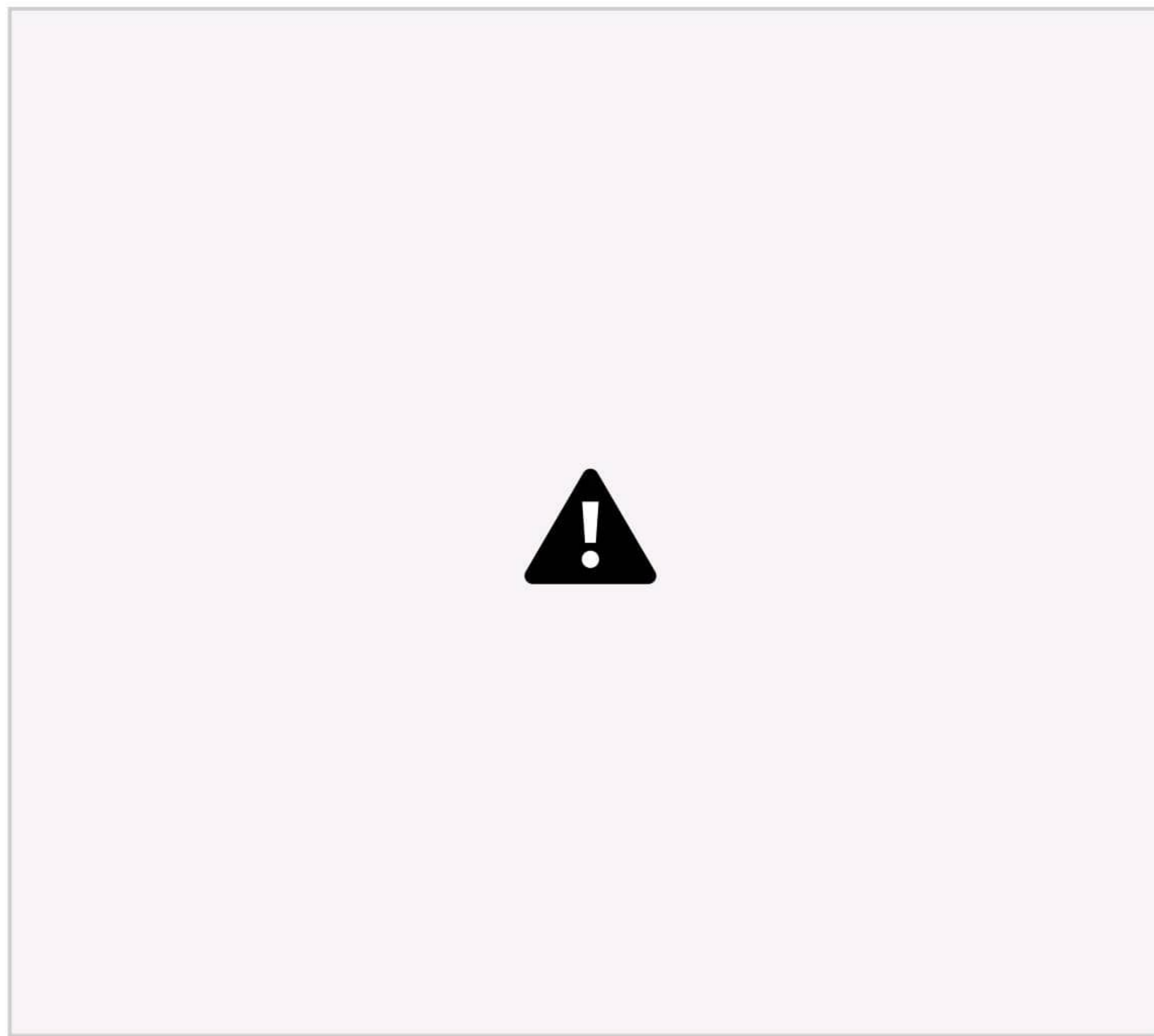
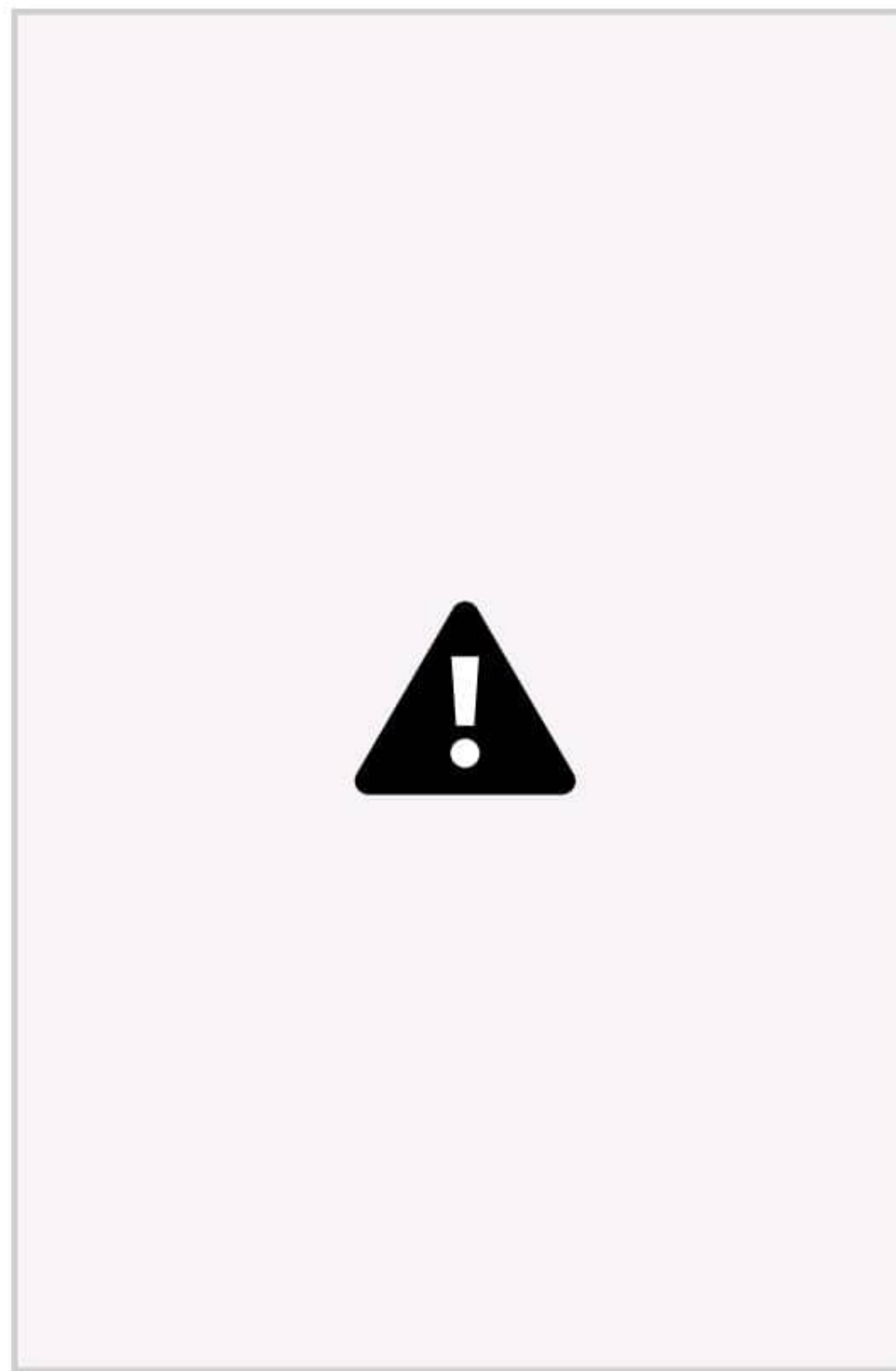
What is loop? Explain nested for with an example. (2016)

In computer programming, a **loop** is a sequence of instructions that is continually repeated until a certain condition is reached. Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number

Nested for loop:

When one for command is performed within another, they are said to be nested. The inside loop is completely repeated for each repetition of the outside loop





P REPAIED BY

SANA MAHMOOD 8

Chapter # 08 Past Paper with answers 2nd year

List all iterative control structures. Explain any one of them. (2014)

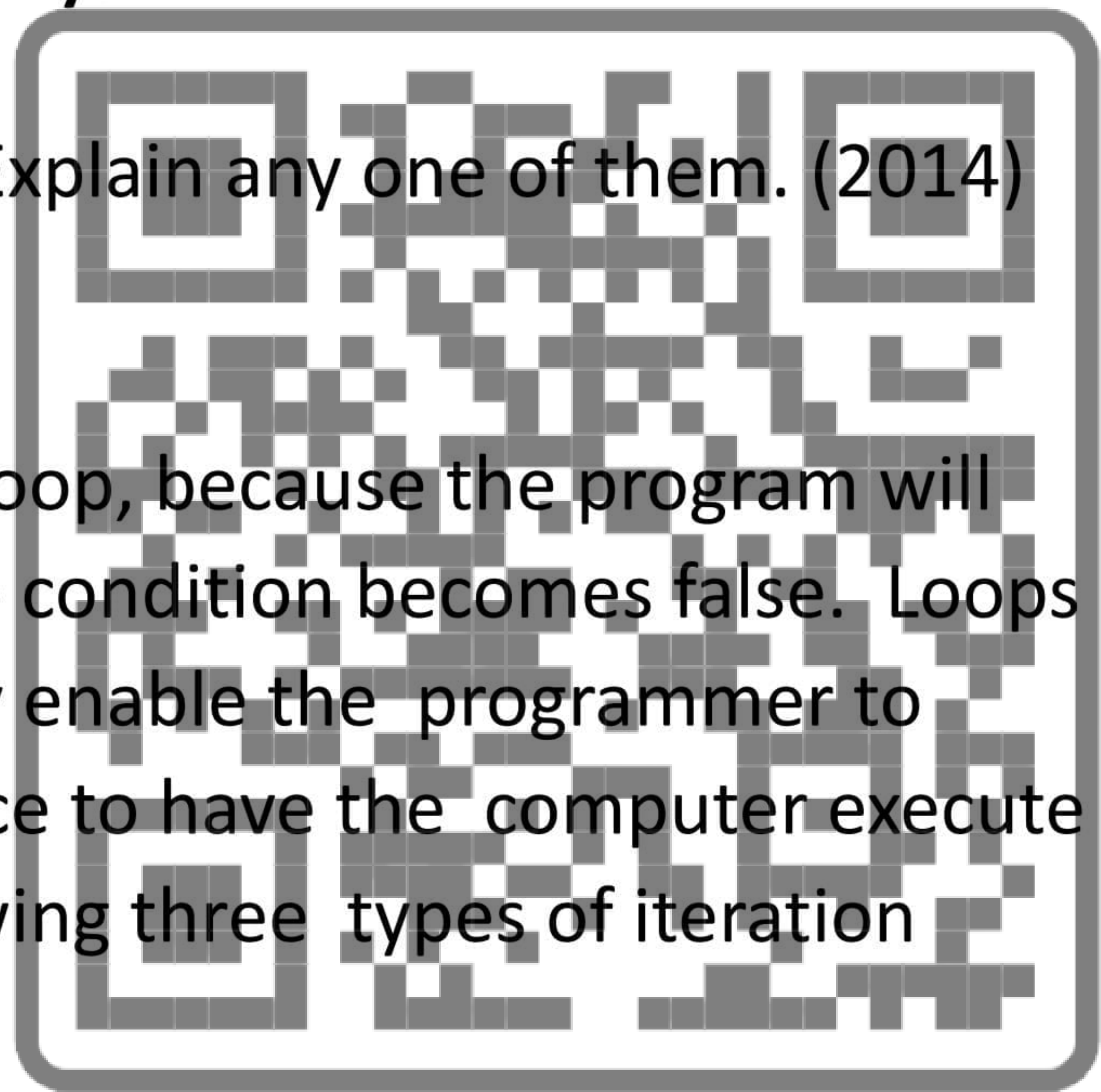
Iteration Control Structures:

• Iteration is often referred to as a loop, because the program will keep repeating the activity until the condition becomes false. Loops simplify programming because they enable the programmer to specify certain instructions only once to have the computer execute them many times. C provides following three types of iteration structures:

- 1: for loop
- 2: while loop
- 3: do-while loop

For loop: When one for command is performed within another, they are said to be nested. The inside loop is completely repeated for each repetition of the outside loop

Write a program in C language that to generate prime number up to 11 (like 2,3,5,7,11). (2013)





P

PREPARED BY SANA MAHMOOD 9

Chapter # 08 Past Paper with answers 2nd year

What is loop? Explain different types of loops in c. (2013)

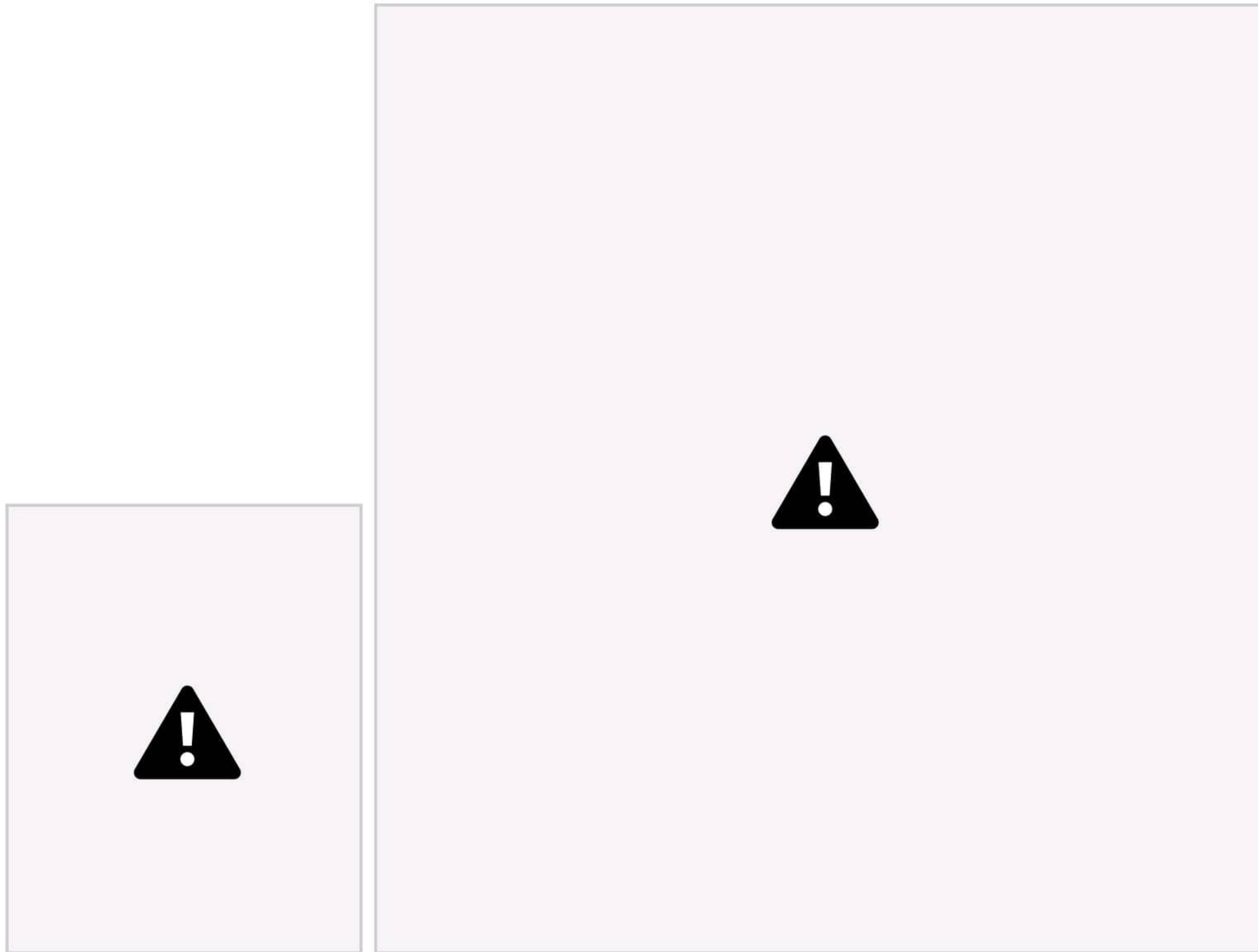
In computer programming, a **loop** is a sequence of instructions that is continually repeated until a certain condition is reached. Typically, a certain process is done, such as getting an item of data and changing it, and then some condition is checked such as whether a counter has reached a prescribed number.

Loop Type

- while loop: Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.
- for loop: Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.
- do...while loop : It is more like a while statement, except that it tests the condition at the end of the loop body.
- nested loops :You can. use one or more loops inside any other while, for, or do -while loop

write a program that prints the following.





JOIN

FOR

MORE!!!

PREPARED BY SANA MAHMOOD 10

Chapter # 08 Past Paper with answers 2nd year

Name the control structures used in C language and discuss all control structures. (2013)

Control structures:

In a program control structures control the flow of execution in program or function. Structured programming uses a limited number of control structures to minimise the complexity of programs and thus cut down on errors. There are three basic control structures in structured programming.

1: Sequence or sequential flow

2: Selection or conditional flow

3: Iteration or repetitive flow.



1: Sequence or sequential flow

The sequence control structures is the most straightforward: the statements are executed in the same order in which they appear

within the program. At a time only one step is to be carried out. Every step is to be carried out once none is repeated. Termination of the last step indicates the end of the program.

2: Selection or conditional flow

The selection control structures employ a number of conditions which lead to a selection of one out of several alternatives.

C provides three types of selection control structures.

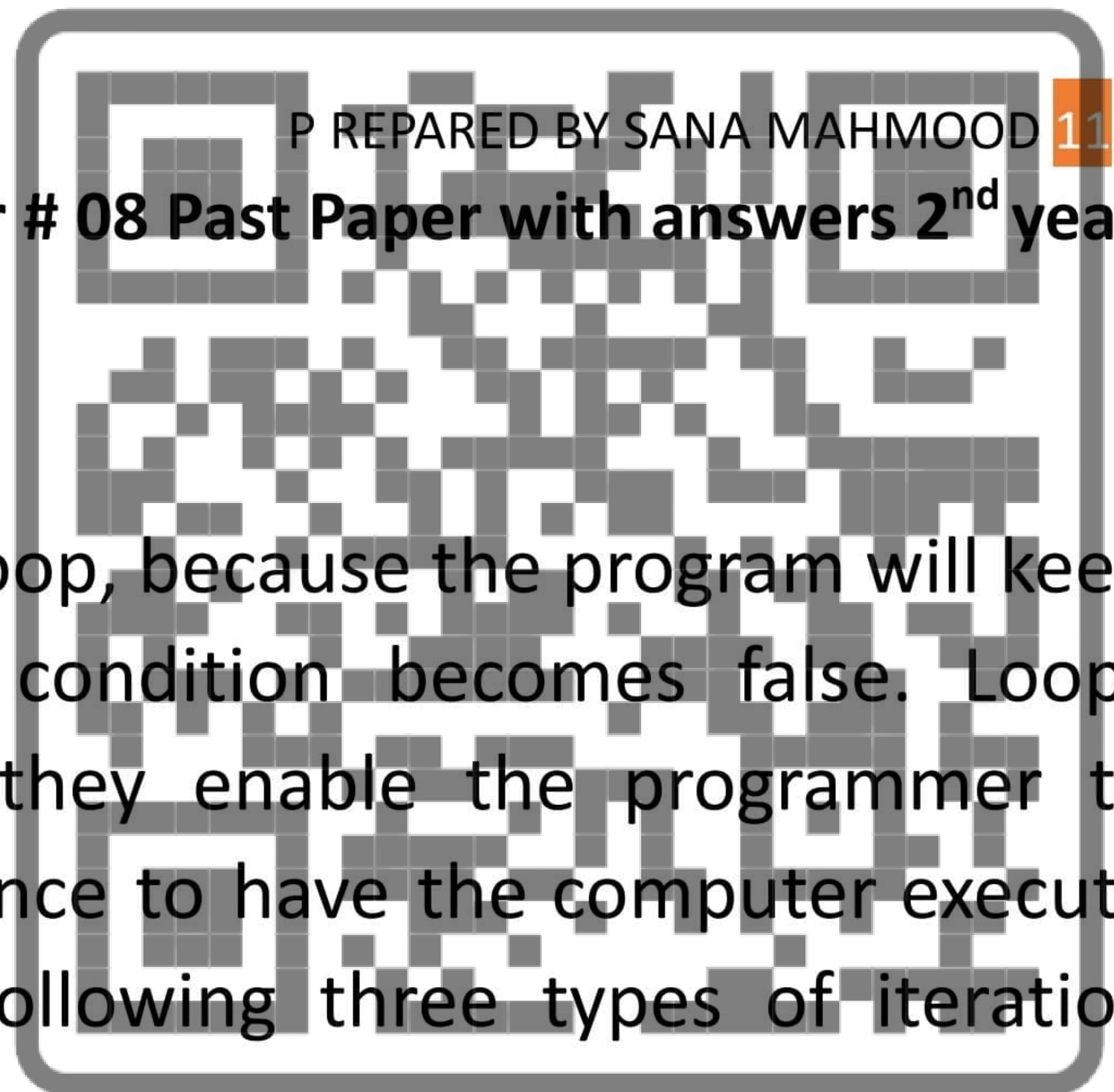
- [if statement](#)
- [if..else statements](#)
- [nested if statements](#)
- [switch statements](#)

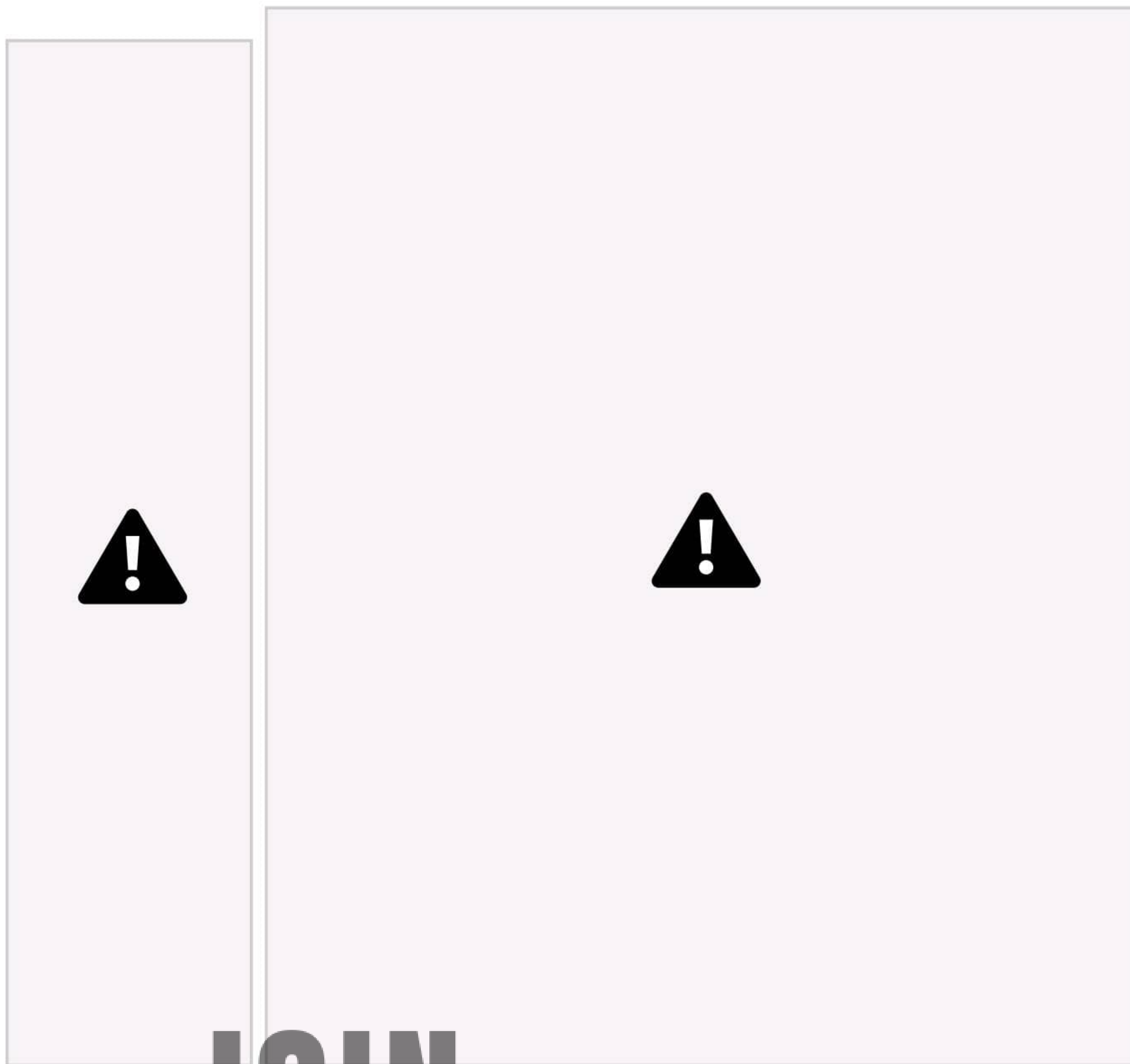
3: Iteration or repetitive flow.

Iteration is often referred to as a loop, because the program will keep repeating the activity until the condition becomes false. Loops simplify programming because they enable the programmer to specify certain instructions only once to have the computer execute them many times. C provides following three types of iteration structures:

- 1: for loop
- 2: while loop
- 3: do-while loop

Write a program that prints the squares of all numbers from 1 to 20. (2012)





**JOIN
FOR
MORE!!!**

Chapter # 08 Past Paper with answers 2nd year

What is the output of the following program. (2011)

a

P REPAIED BY SANA MAHMOOD 12





JOIN

PREPARED BY SANA MAHMOOD 13

FOR

MORE!!!

Chapter # 08 Past Paper with answers 2nd year

Write a program that inputs your age and your sisters age and then display who is older of the two. (2010)





JOIN FOR MORE!!!







Chapter # 08: Iteration Control Structures

- Iteration is often referred to as a loop, because the program will keep repeating the activity until the condition becomes false. Loops simplify programming because they enable the programmer to specify certain instructions only once to have the computer execute them many times. C provides following three types of iteration structures:
- 1: for loop
- 2: while loop
- 3: do-while loop

**JOIN
FOR
MORE!!!**



What is looping?

simplest example=>

```
printf(" Hello ");
```

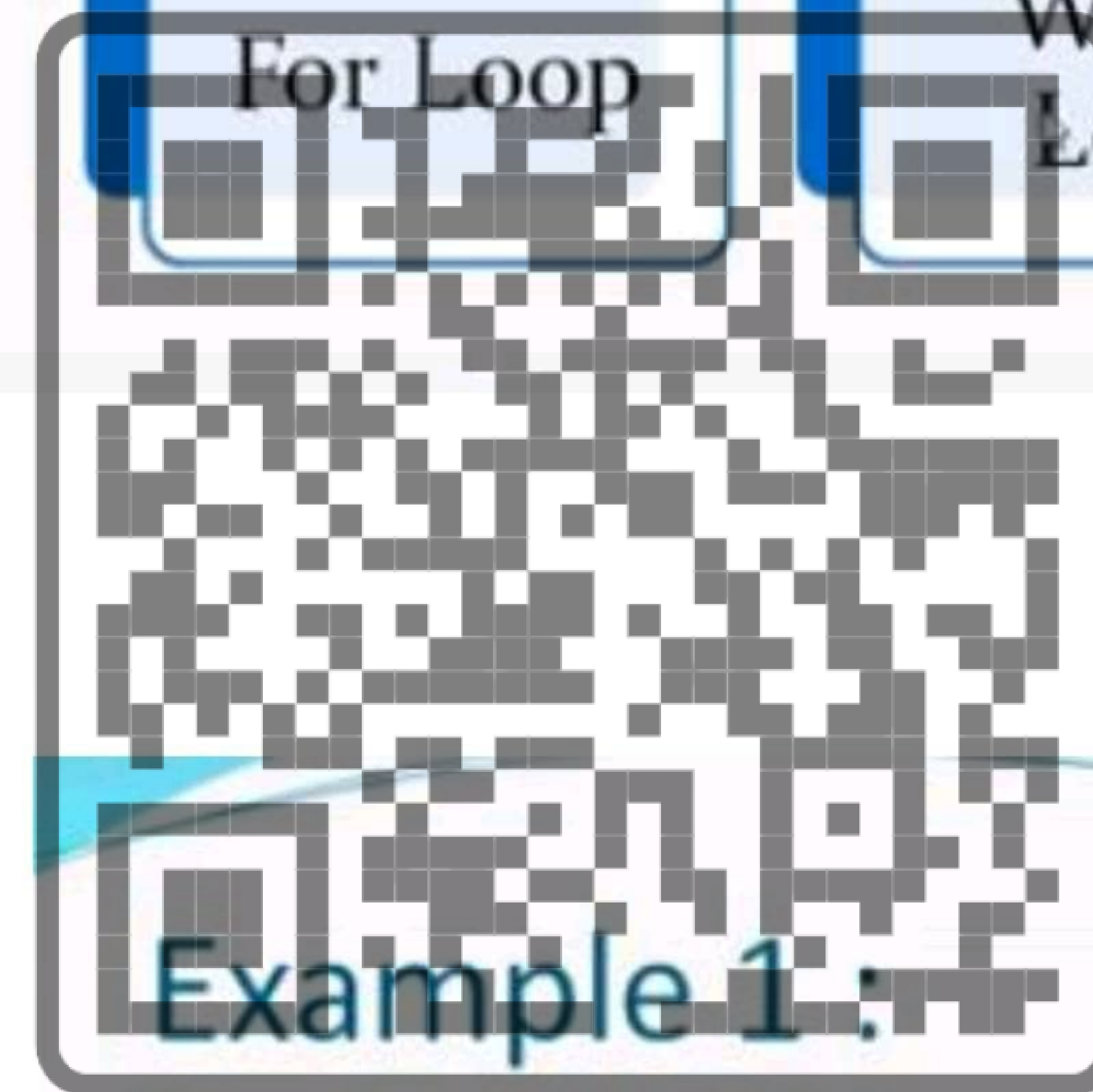
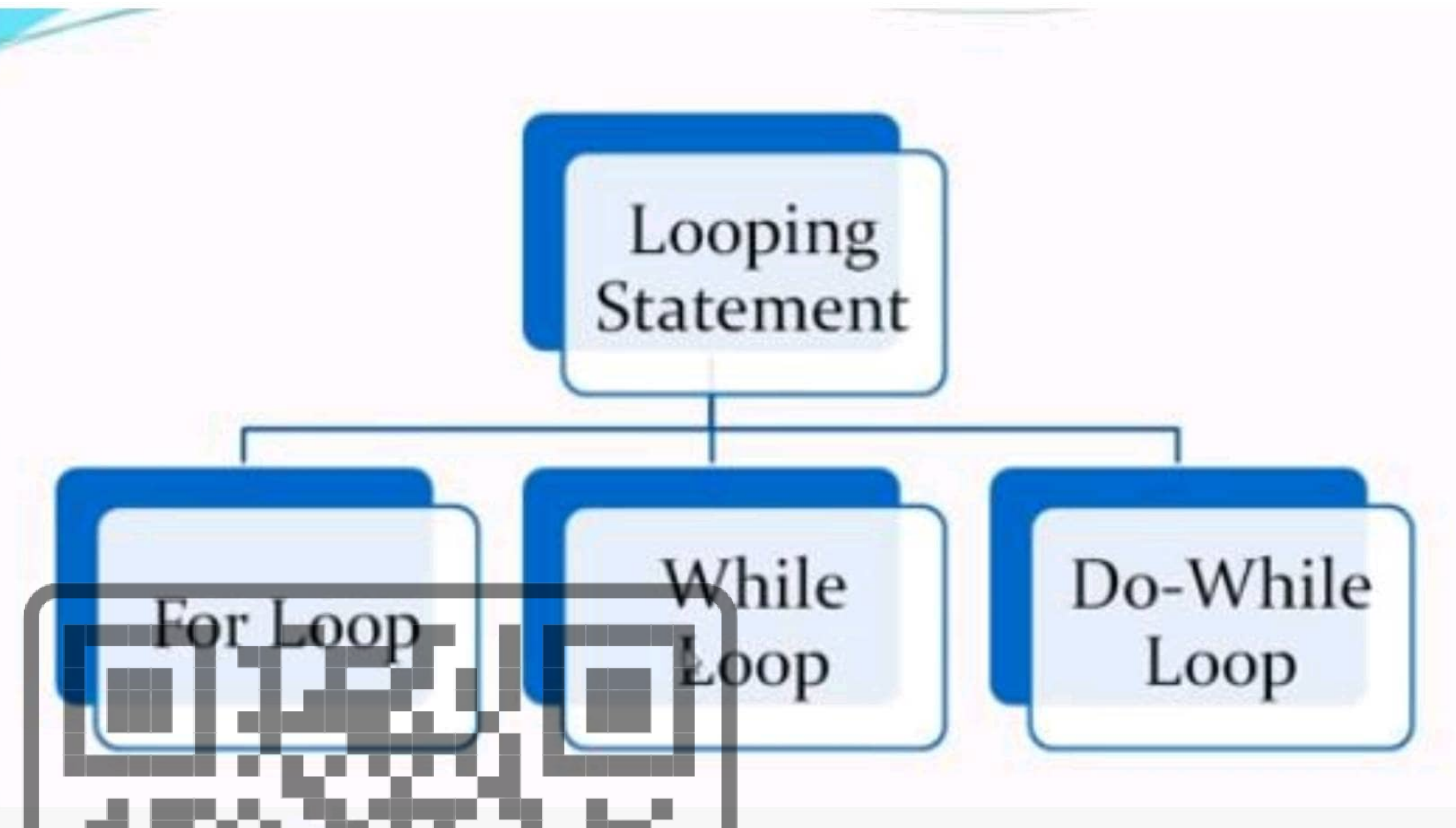
if I want to print hello **100** times then ???

**JOIN
FOR
MORE!!!**

For Loop:

For loop is used when we know these **3** things =

- Lower limit (Initial Value)
- Upper Limit (Maximum Value)
- Increment or Decrement



If we want to display **Hello** 100 times

- Lower limit = **1**
- Upper Limit = **100**
- Increment or Decrement = **Increment**



Example 2 :

If we want to display 20 to 100 digits

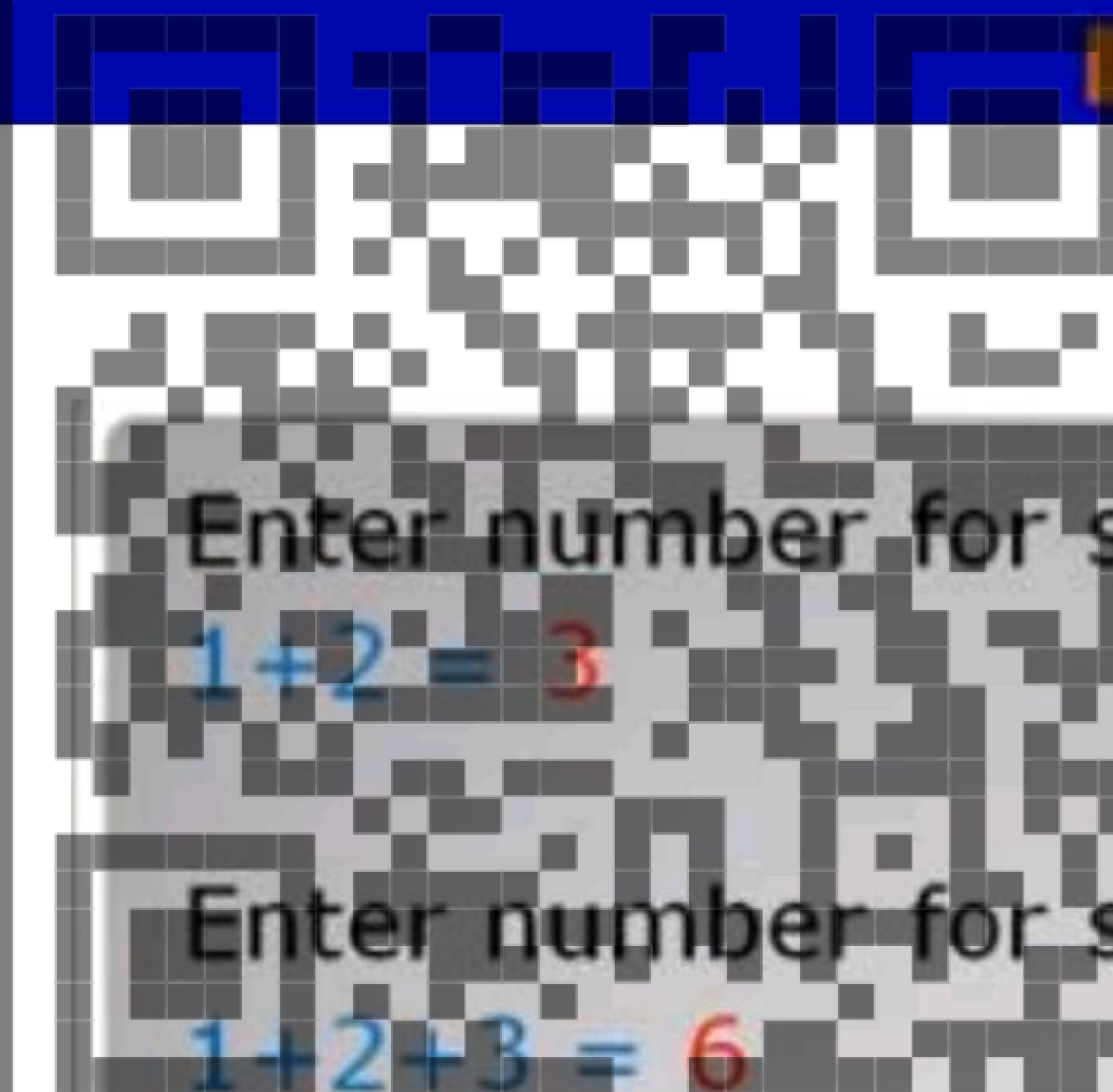
- Lower limit = 20
- Upper Limit = 100
- Increment or Decrement = Increment

JOIN
FOR
MORE!!!

```
void main()
{
    int i;
    clrscr();
    for( i=1; i<=100; i++ )
    {
        printf("%d \t", i);
    }
    getch();
}
```

Syntax of FOOR loop

```
int i;
for ( i = 1; i <= 100; i++ )
{
    printf(" HELLO ");
}
```



Enter number for summation : 2

1+2 = 3

Enter number for summation : 3

1+2+3 = 6

Enter number for summation : 4

1+2+3+4 = 10




```
#include<conio.h>
```

```
void main()
```

```
{
```

```
int i,n,sum=0;
```

```
clrscr();
```

```
printf("enter number for summation");
```

```
scanf("%d",&n);
```

```
for (i=1; i<=n; i++)
```

```
{
```

```
sum=sum+i;
```

```
}
```

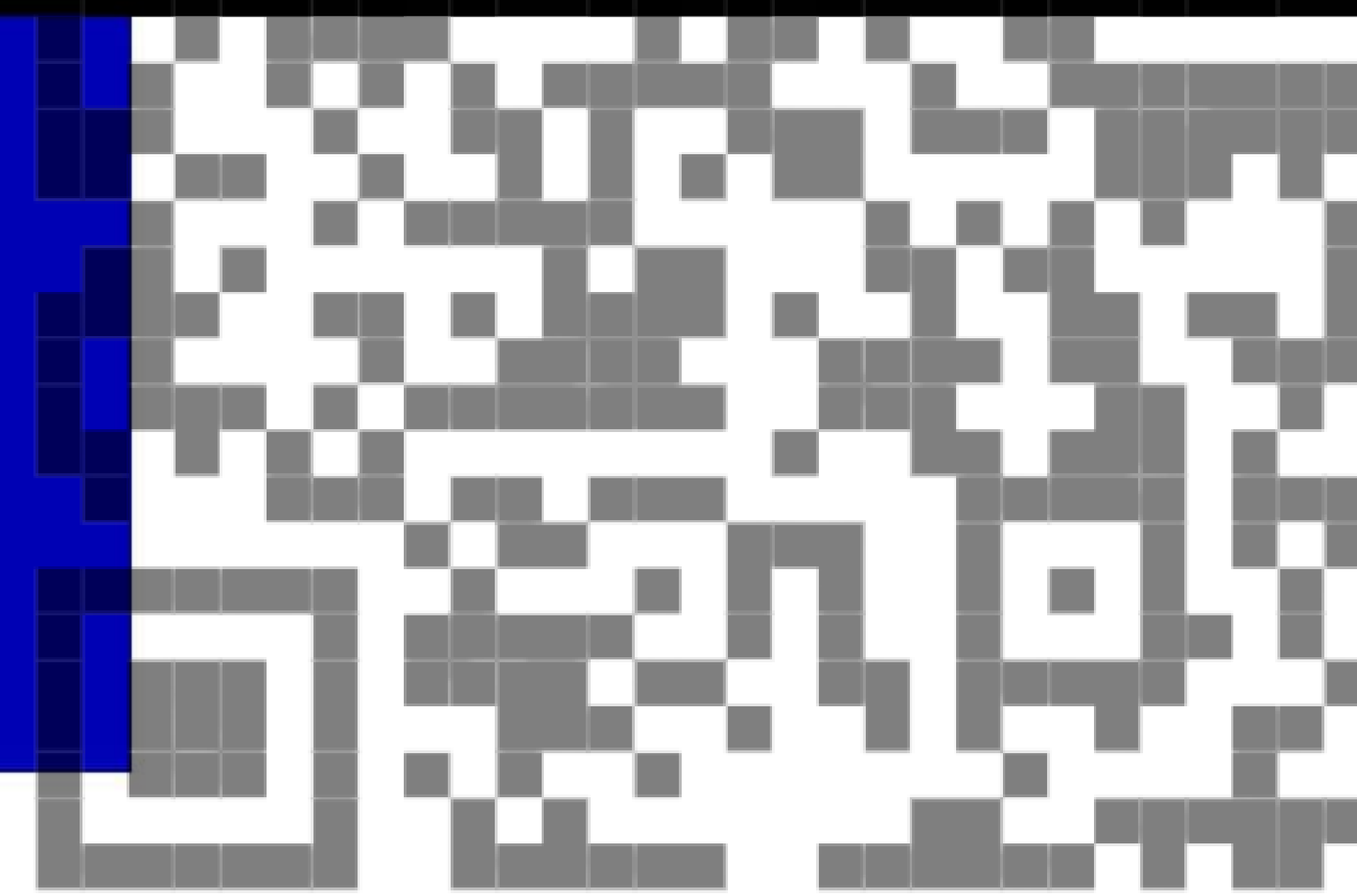
```
printf("summation is %d",sum);
```

```
getch();
```

```
}
```

JOIN
FOR
MORE!!!

enter number for summation 5
summation is 15_



- | Loop Type | Description |
|---------------------|--|
| • while loop: | Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body. |
| • for loop: | Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable. |
| • do...while loop : | It is more like a while statement, except that it tests the condition at the end of the loop body. |
| • nested loops : | You can. use one or more loops inside any other while, for, or do -while loop |

JOIN
FOR
MORE!!!



for loop:

- The for loop is used to repeat a statement or a block of statements for a specified number of times. The for loop statement includes the initialization of the counter, the condition, and the increment. The general form of the for statement is

```
for ( initialization; condition; increment )  
{  
    statement(s);  
}
```



Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int count;
    clrscr();
    for (count=1; count<=10; count++)
        printf("%3d",count);_
    getch();
}
```

**JOIN
FOR
MORE!!!**

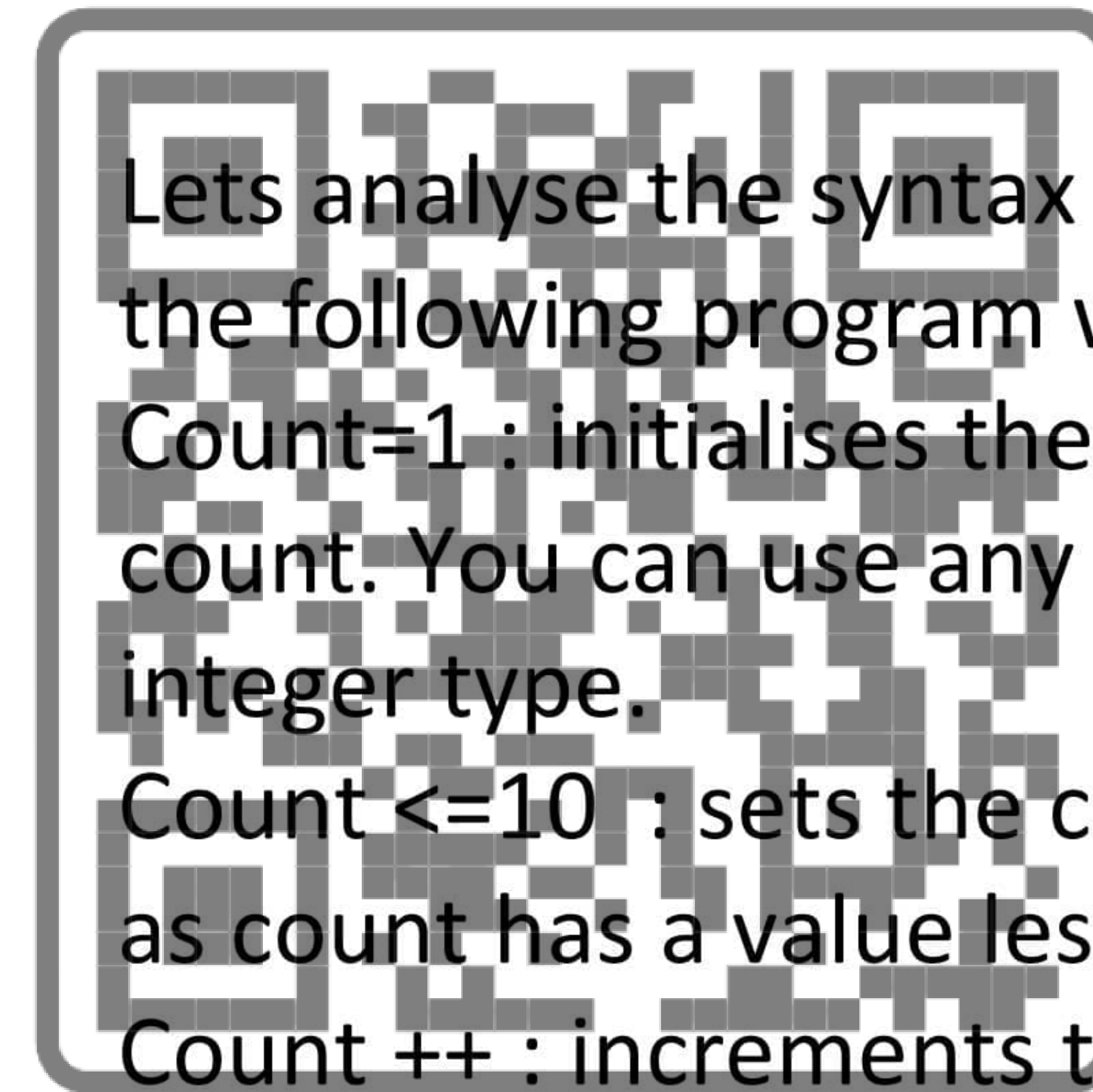


1 2 3 4 5 6 7 8 9 10

Lets analyse the syntax of for loop using output of the following program which is: 1 2 3 4 5 6 7 8 9 10
Count=1 : initialises the starting value of the variable count. You can use any name but must be an integer type.

Count <=10 : sets the condition to repeat the loop as count has a value less than or equal to 10.

Count ++ : increments the value of count after each repetition.



Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int count;
clrscr();
for (count=10; count>=1; count--)
printf("%3d",count);
getch();
}
```

10 9 8 7 6 5 4 3 2 1

Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int count;
clrscr();
for (count=1; count<=6; count++)
printf("take care\n");
getch();
}
```

take care
take care
take care
take care
take care
take care



- Example

```
#include <stdio.h>

void main ()
{
/* for loop execution */
for( int a = 10; a < 20; a = a + 1 )
    printf("value of a: %d\n", a);
    getch();
}
```

value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
value of a: 16
value of a: 17
value of a: 18
value of a: 19

**JOIN
FOR
MORE!!!**



Multiple statements

- To perform more than one statements in the loop, enclose the whole series in braces is called multiple statements.

- example:

- `#include <stdio.h>`

- `void main()`

- `{`

- `int j, sum = 0;`

- `printf("The first 10 natural number is :\n");`

`for (j = 1; j <= 10; j++)`

`{`

`sum = sum + j;`

`printf("%d ",j);`

`}`

`printf("\nThe Sum is : %d\n", sum);`

`getch();`

`}`

The first 10 natural number is : 1 2 3 4 5 6 7 8 9 10

The Sum is : 55

**JOIN
FOR
MORE!!!**

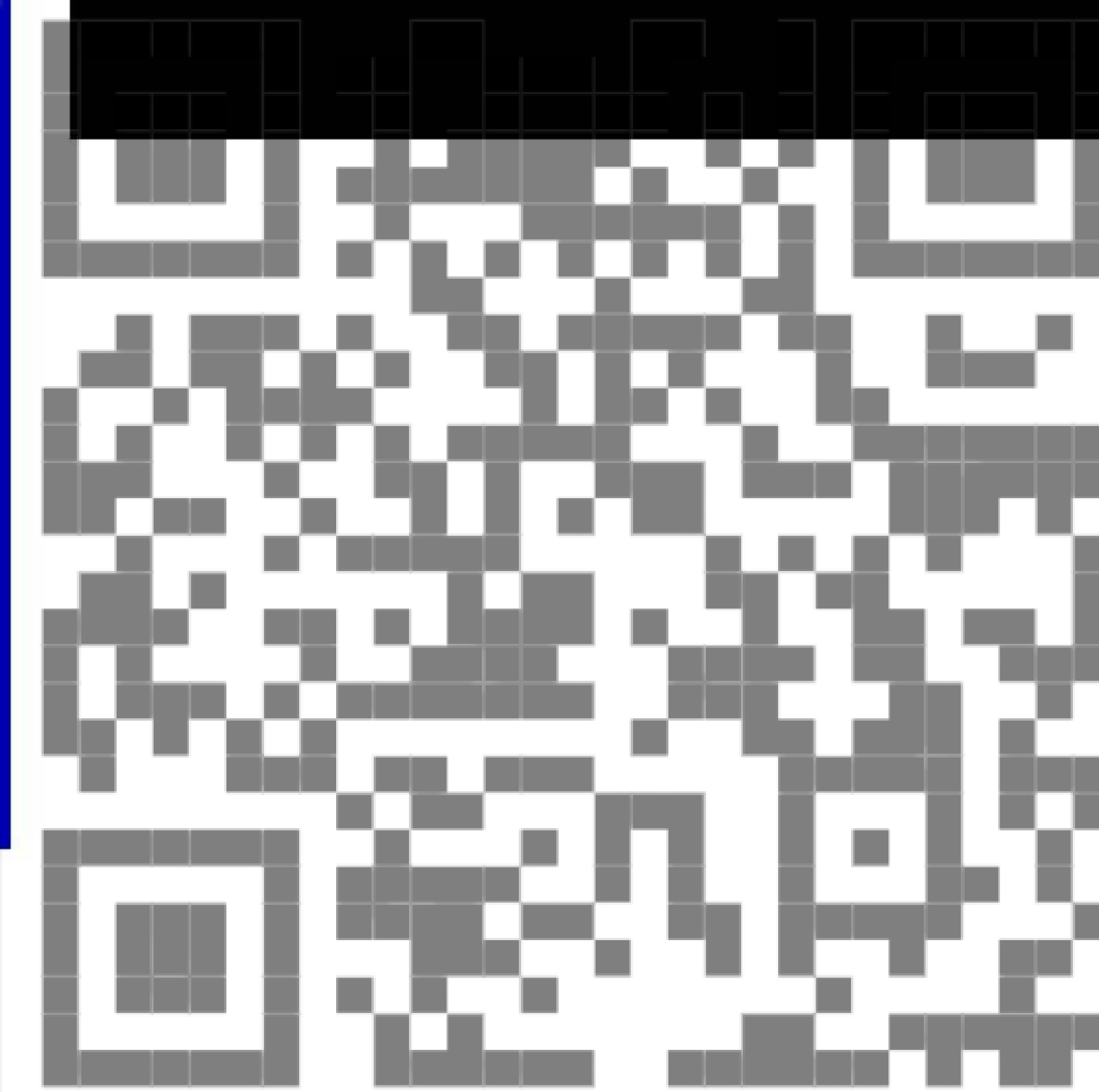



```
#include<conio.h>
void main()
{
int i,sum=0;
clrscr();
for(i=1; i<100; i=i+2)
sum=sum+i;
printf("\n\n\n\t the sum is =%d",sum);

getch();
}
```

**JOIN
FOR
MORE!!!**

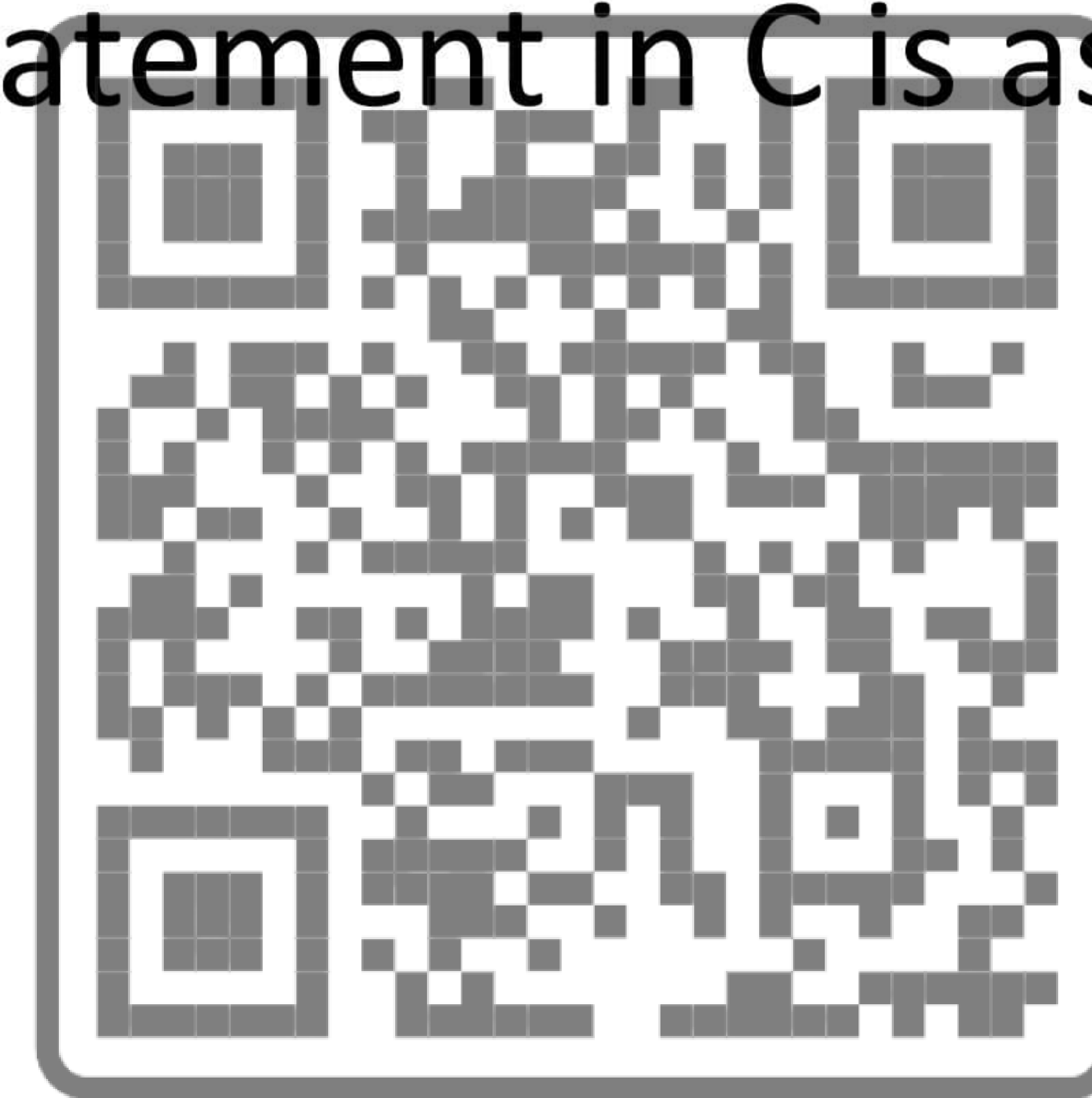
the sum is =2500_



Nested for loops

- When one for command is performed within another, they are said to be nested. The inside loop is completely repeated for each repetition of the outside loop.
- The syntax for a nested for loop statement in C is as follows:
- for (init; condition; increment)
- {
- for (init; condition; increment)
- {
- statement(s);
- }
- statement(s);
- }

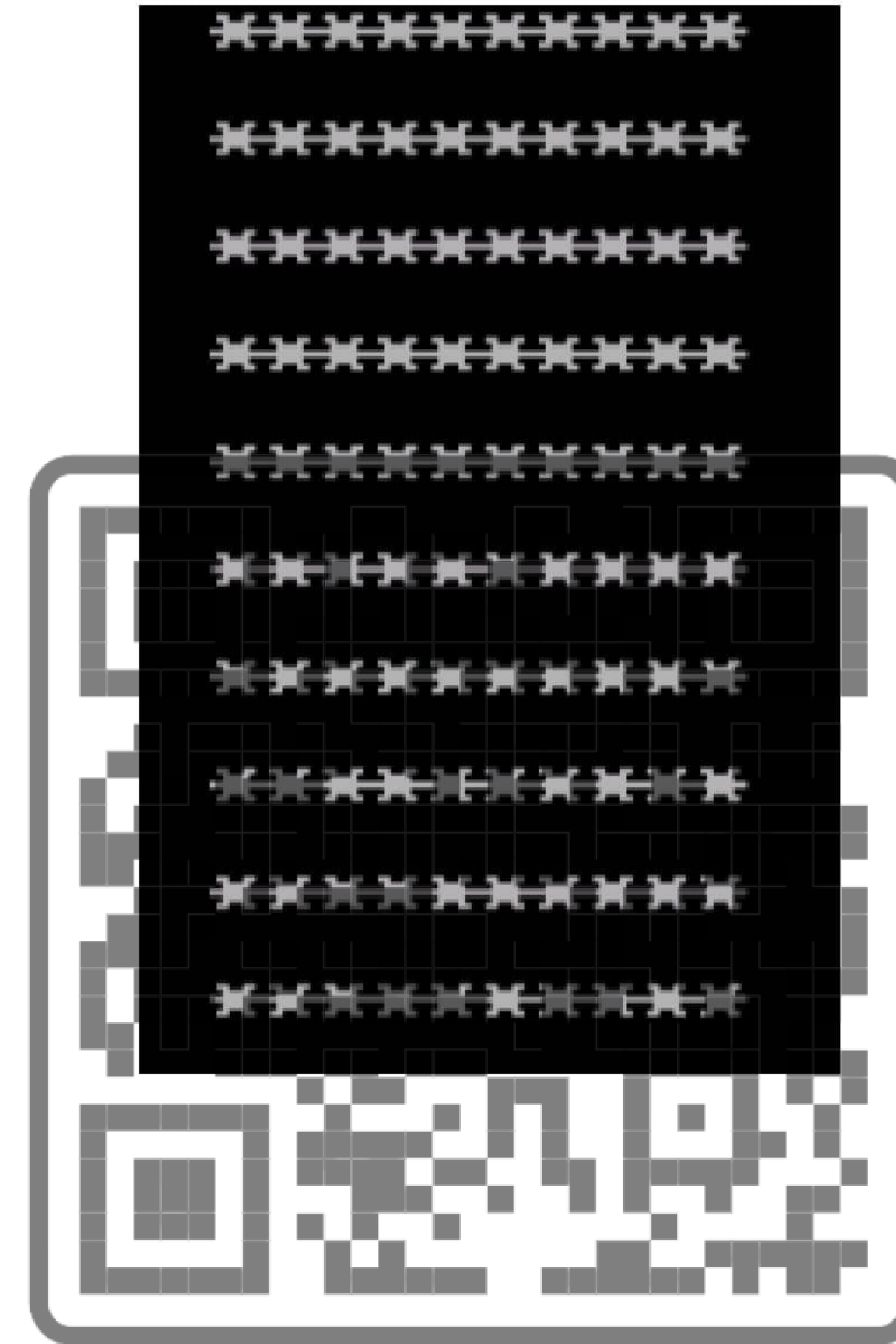
JOIN
FOR
MORE!!!



Example:

```
#include<conio.h>
void main()
{
int row,column;
clrscr();
for(row=1;row<=10;row++)
{
for(column=1;column<=10;column++)
printf("x");
printf("\n");
}
getch();
}
```

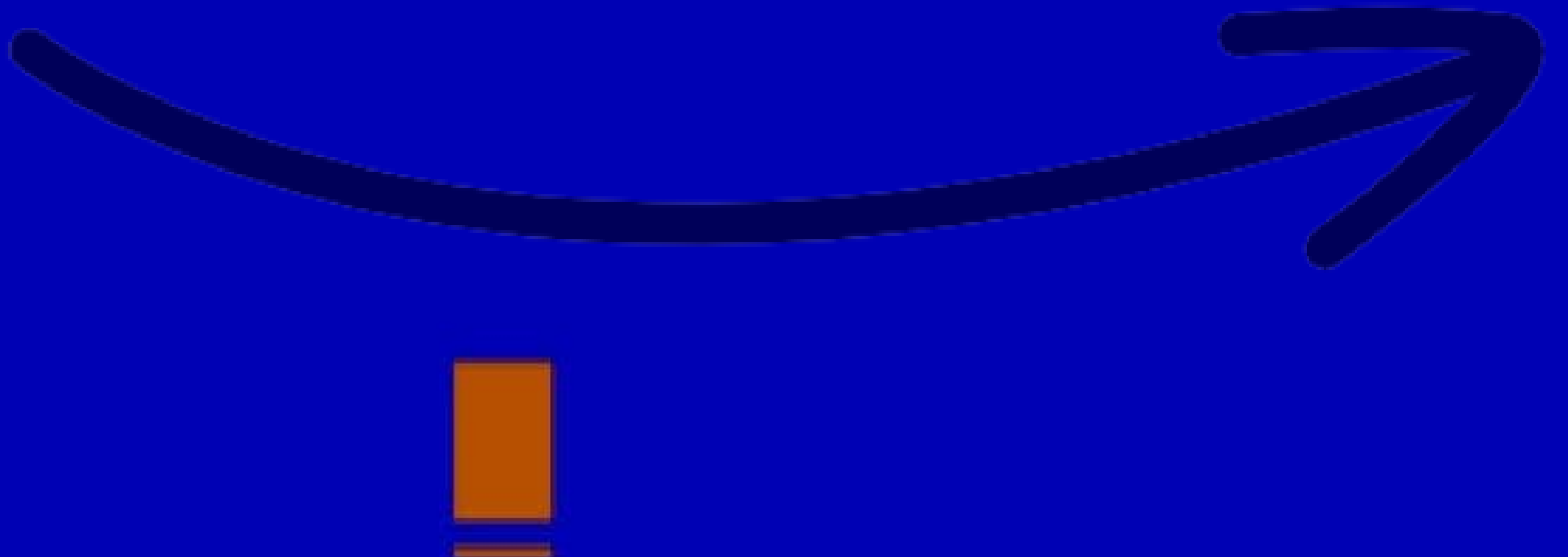
**JOIN
FOR
MORE!!!**



example

```
#include<conio.h>
void main()
{
int row,column;
clrscr();
for(row=1;row<=10;row++)
{
for(column=1;column<=row;column++)
printf("%*");
printf("\n");
}
getch();
}
```

JOIN FOR MORE!!!



example

```
NONAME01.CPP
#include<conio.h>
void main()
{
int row,column;
clrscr();
printf("\t\tMy handy multiplication table\n\n");
for(row=1;row<=10;row++)
{
for(column=1;column<=10;column++)
printf("%6d",row*column);
printf("\n");
}
getch();
}
```

JOIN
FOR
MORE!!!



My handy multiplication table

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100




```
#include<conio.h>
void main()
{
int i,sum=0;
clrscr();
for(i=1; i<100; i=i+2)
sum=sum+i;
printf("\n\n\n\t the sum is =%d",sum);

getch();
}
```

**JOIN
FOR
MORE!!!**

the sum is =2500_



While Loop:

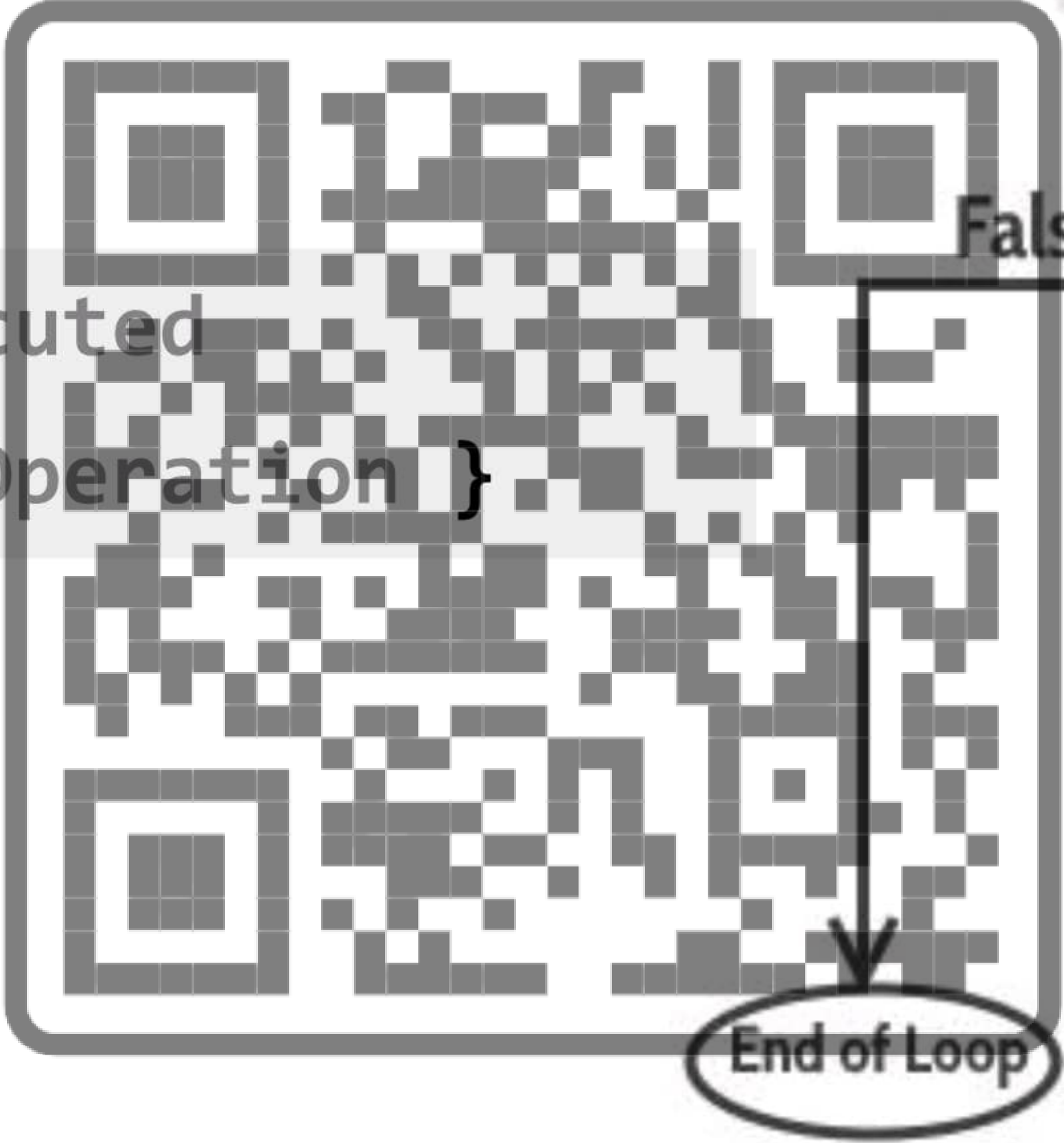
- In most computer programming languages, a **while loop** is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The *while* loop can be thought of as a repeating if statement.
- The *while* construct consists of a block of code and a condition/expression. The condition/expression is evaluated, and if the condition/expression is *true*, the code within the block is executed. This repeats until the condition/expression becomes false. Because the *while* loop checks the condition/expression before the block is executed, the control structure is often also known as a **pre-test loop**.

JOIN
FOR
MORE!!!

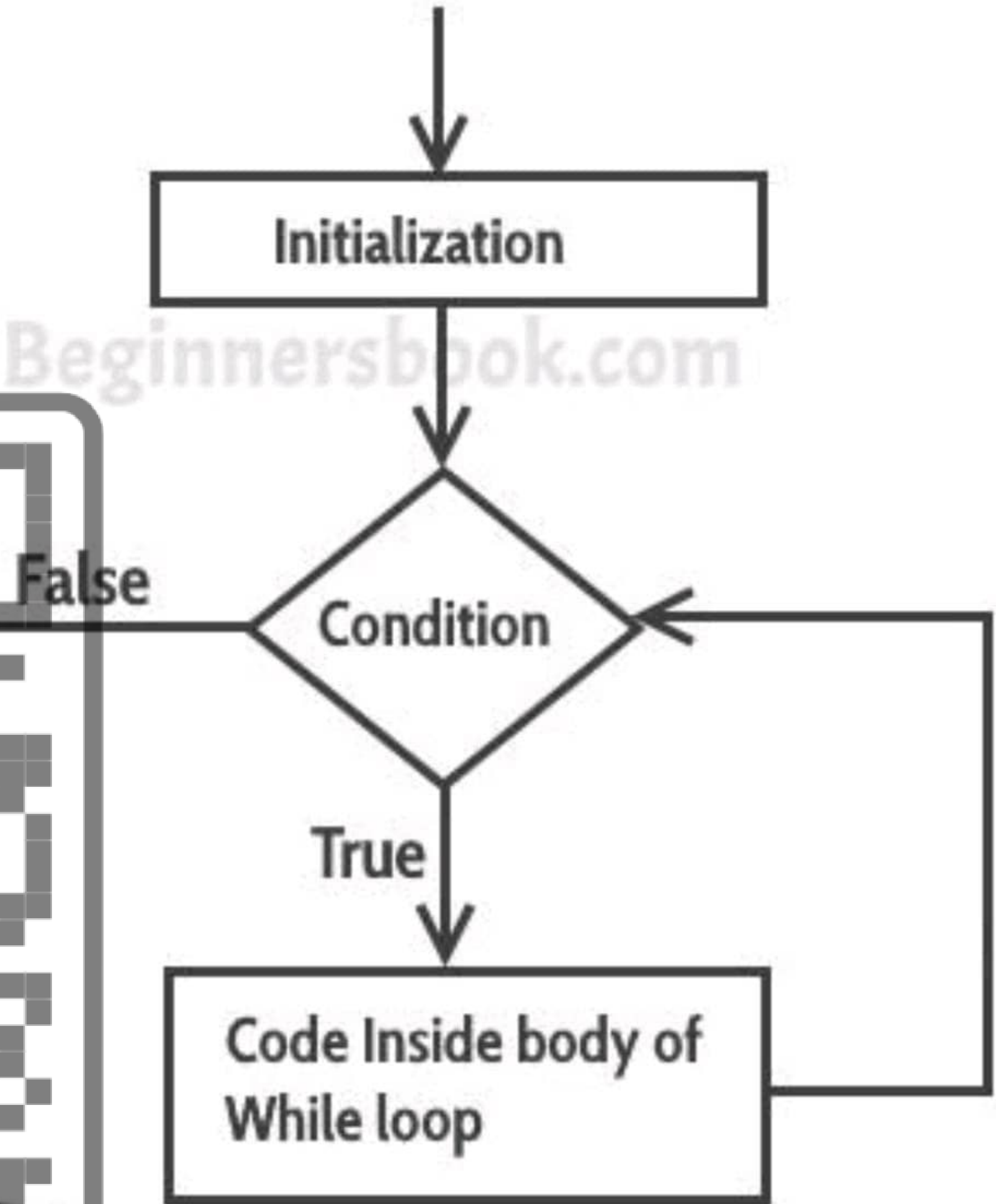


`while (condition test) { //Statements to be executed repeatedly // Increment (++) or Decrement (--) Operation }`

JOIN
FOR
MORE!!!



End of Loop



Beginnersbook.com



Example: this program uses a while loop to print number from 1 to 10.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int count=1;
clrscr();
while (count<=10)
{
printf("%3d",count);
++count;
}
getch();
}
```

1 2 3 4 5 6 7 8 9 10_

**JOIN
FOR
MORE!!!**



Example: write a program to get a user input ,how many times a name print. using while loop.

```
#include<conio.h>
void main()
{
    int num=1,n;
    clrscr();
    printf("\nEnter a number:");
    scanf("%d",&n);
    while(num<=n)
    {
        printf("sana\n");
        num++;
    }
    getch();
}
```

JOIN
FOR
MORE!!

Enter a number:5

sana

sana

sana

sana

sana



Example: write a program which uses while loop to print out a ASCII table codes from 32 to 127

```
#include<conio.h>
void main()
{
    int i;
    clrscr();
    i=31;
    while(i<128)
    {
        printf("%3d=%c\t", i, i);
        i++;
    }
    getch();
}
```

JOIN
FOR
MORE!!!



31=▼	32=	33=!	34="	35=#	36=\$	37=%	38=&	39='	40=(
41=)	42=*	43=+	44=,	45=-	46=.	47=/	48=0	49=1	50=2
51=3	52=4	53=5	54=6	55=7	56=8	57=9	58=:	59=;	60=<
61==	62=>	63=?	64=@	65=A	66=B	67=C	68=D	69=E	70=F
71=G	72=H	73=I	74=J	75=K	76=L	77=M	78=N	79=O	80=P
81=Q	82=R	83=S	84=T	85=U	86=V	87=W	88=X	89=Y	90=Z
91=[	92=\	93=]	94=^	95=_	96='	97=a	98=b	99=c	100=d
101=e	102=f	103=g	104=h	105=i	106=j	107=k	108=l	109=m	110=n
111=o	112=p	113=q	114=r	115=s	116=t	117=u	118=v	119=w	120=x
121=y	122=z	123={	124=	125=}	126=~	127=Δ	128=		



Nested while loop:

- Using While loop within while loops is said to be **nested while loop**. In nested while loop one or more statements are included in the body of the loop. In nested while loop, the number of iterations will be equal to the number of iterations in the outer loop multiplies by the number of iterations in the inner loop which is most same as nested for loop.

JOIN
FOR
MORE!!!



Syntax

```
while (test condition)
{
    while (test condition)
    {
        .// inner while loop
    }
    .// outer while loop
}
```



Example:

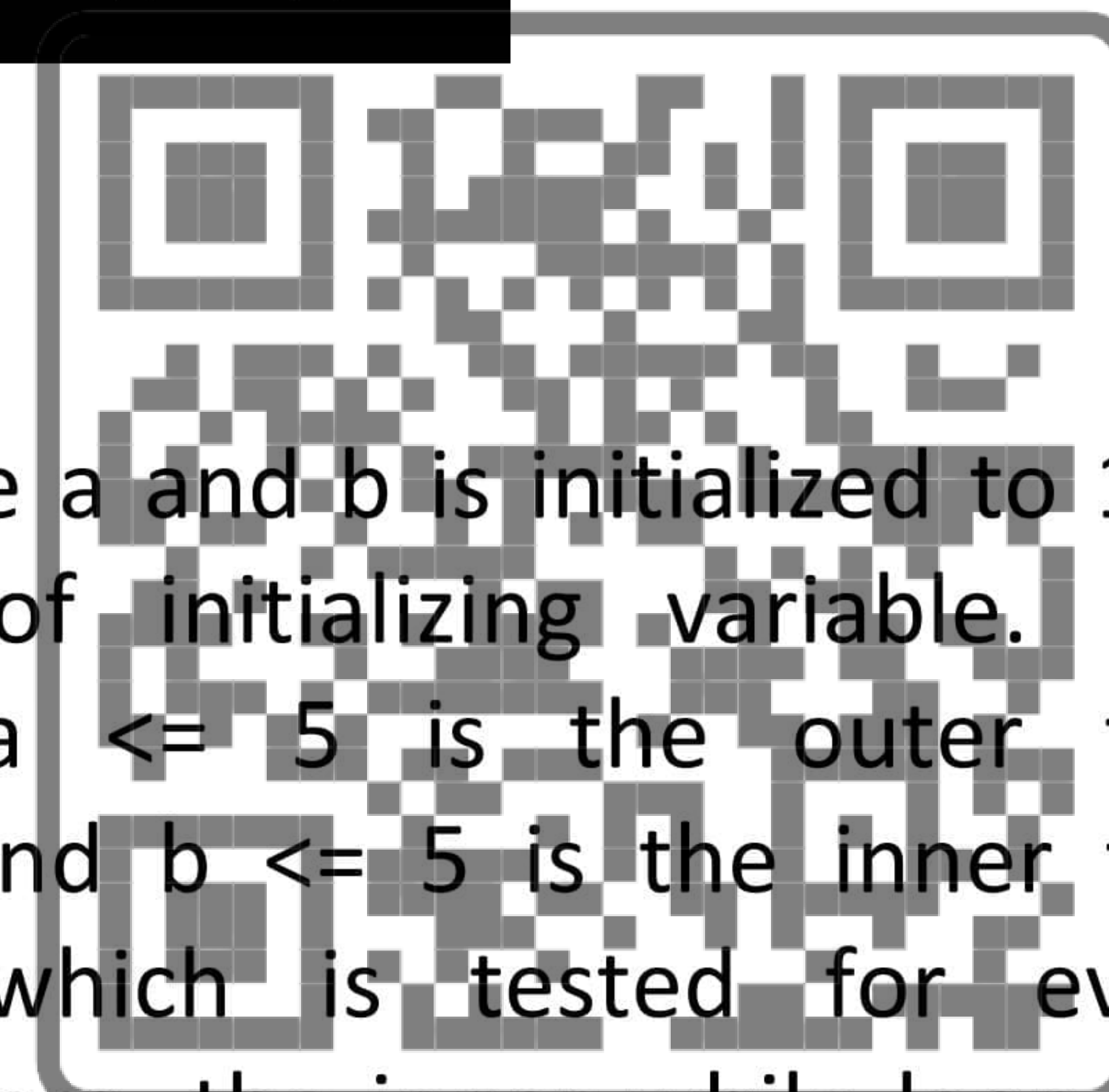
```
#include<conio.h>
void main()
{
int a=1;
int b;
clrscr();
while(a<=5)
{
b=1;
while(b<=5)
{
printf("%3d",b);
b++;
}
printf("\n");
a++;
}
getch();
}
```

1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5

JOIN
FOR
MORE!!!

Note:

The variable a and b is initialized to 1 at the time of initializing variable. The condition $a \leq 5$ is the outer test condition and $b \leq 5$ is the inner test condition which is tested for every iteration. Here, the inner while loop will be executed for 5 times for every time the test condition in the outer loop is true.




```
void main()
{
int a=1;
int b;
clrscr();
while(a<=5)
{
b=1;
while(b<=5)
{
printf("x");
b++;
}
printf("\n");
a++;
}
getch();
}
```

```
xxxxxx
xxxxxx
xxxxxx
xxxxxx
xxxxxx
```

**FOR
MORE!!!**




```
void main()
{
int a=1;
int b;
clrscr();
while(a<=5)
{
b=1;
while(b<=a)
{
printf("*");
b++;
}
printf("\n");
a++;
}
getch();
}
```

JOIN

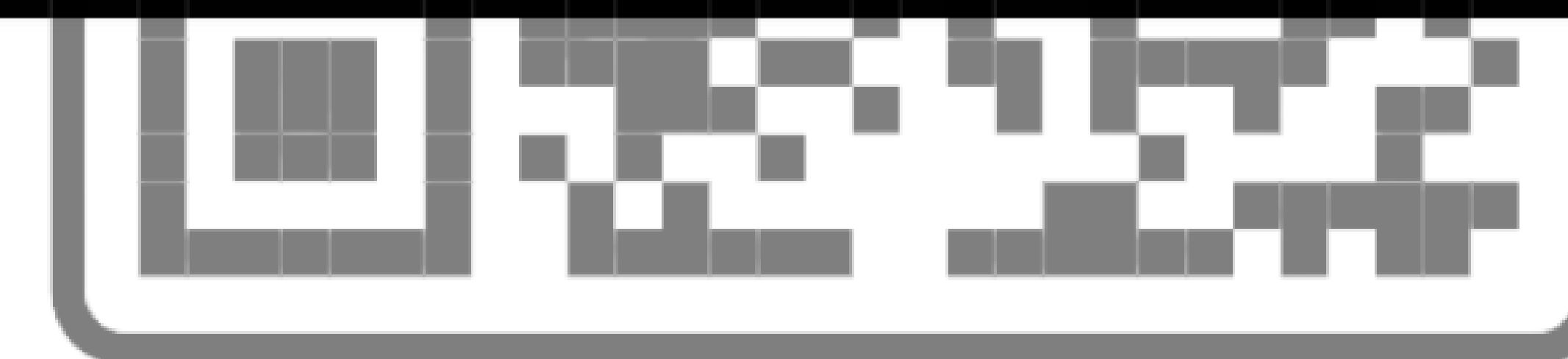
```
*
**
***
****
*****
```

!!




```
void main()
{
    int a=1;
    int b;
    clrscr();
    while(a<=10)
    {
        b=1;
        while(b<=10)
        {
            printf("%6d",a*b);
            b++;
        }
        printf("\n");
        a++;
    }
    getch();
}
```

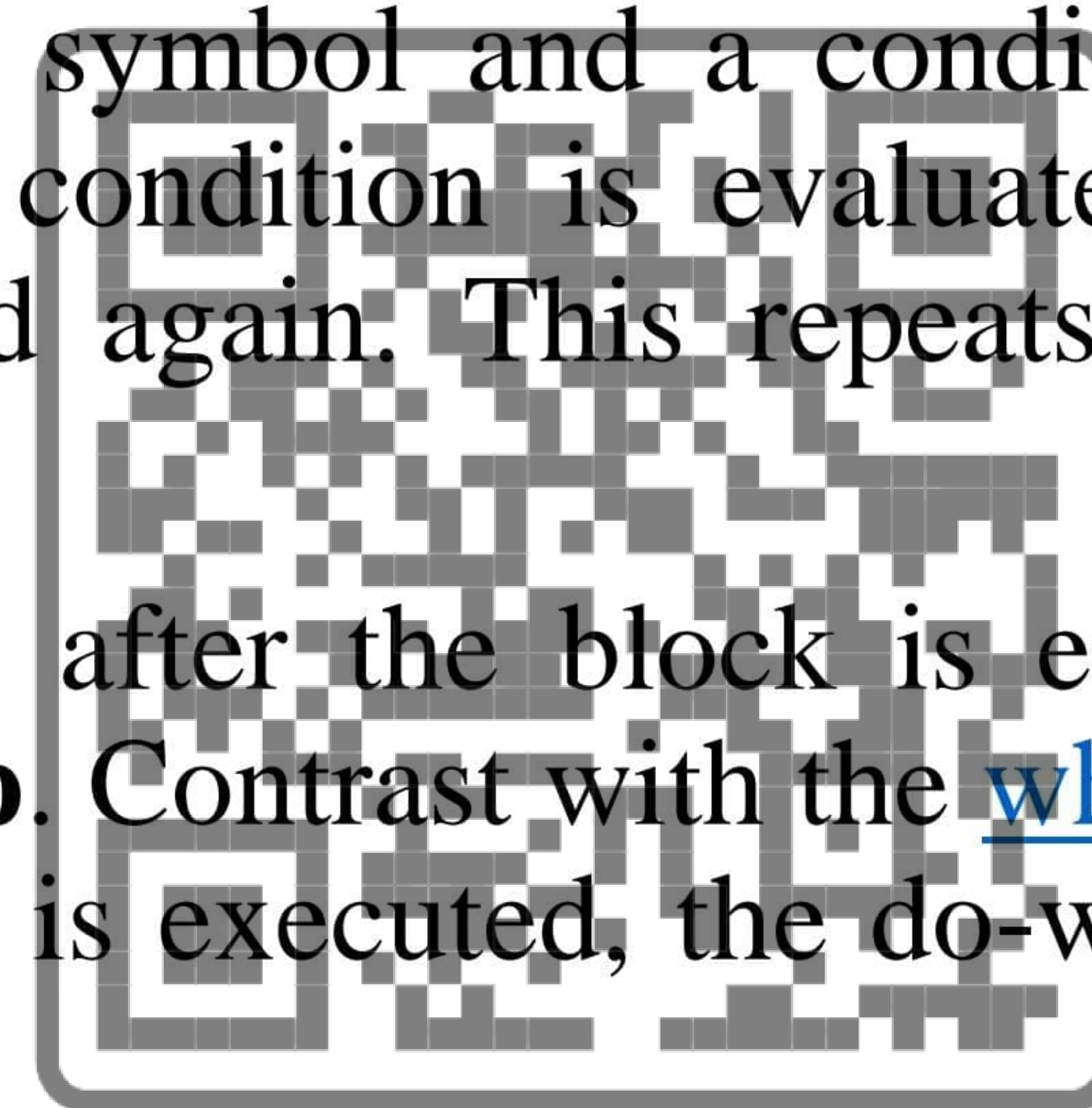
1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100



Do-while loop

- In most [computer programming](#) languages, a **do while loop** is a [control flow statement](#) that executes a block of code at least once, and then repeatedly executes the block, or not, depending on a given [boolean](#) condition at the end of the block.
- The *do while* construct consists of a process symbol and a condition. First, the code within the block is executed, and then the condition is evaluated. If the condition is [true](#) the code within the block is executed again. This repeats until the condition becomes [false](#).
- Because do while loops check the condition after the block is executed, the control structure is often also known as a **post-test loop**. Contrast with the [while loop](#), which tests the condition before the code within the block is executed, the do-while loop is an exit-condition loop.
- This means that the code must always be executed first and then the expression or test condition is evaluated. If it is true, the code executes the body of the loop again. This process is repeated as long as the expression evaluates to true. If the expression is false, the loop terminates and control transfers to the statement following the do-while loop.

JOIN
FOR
MORE!!!



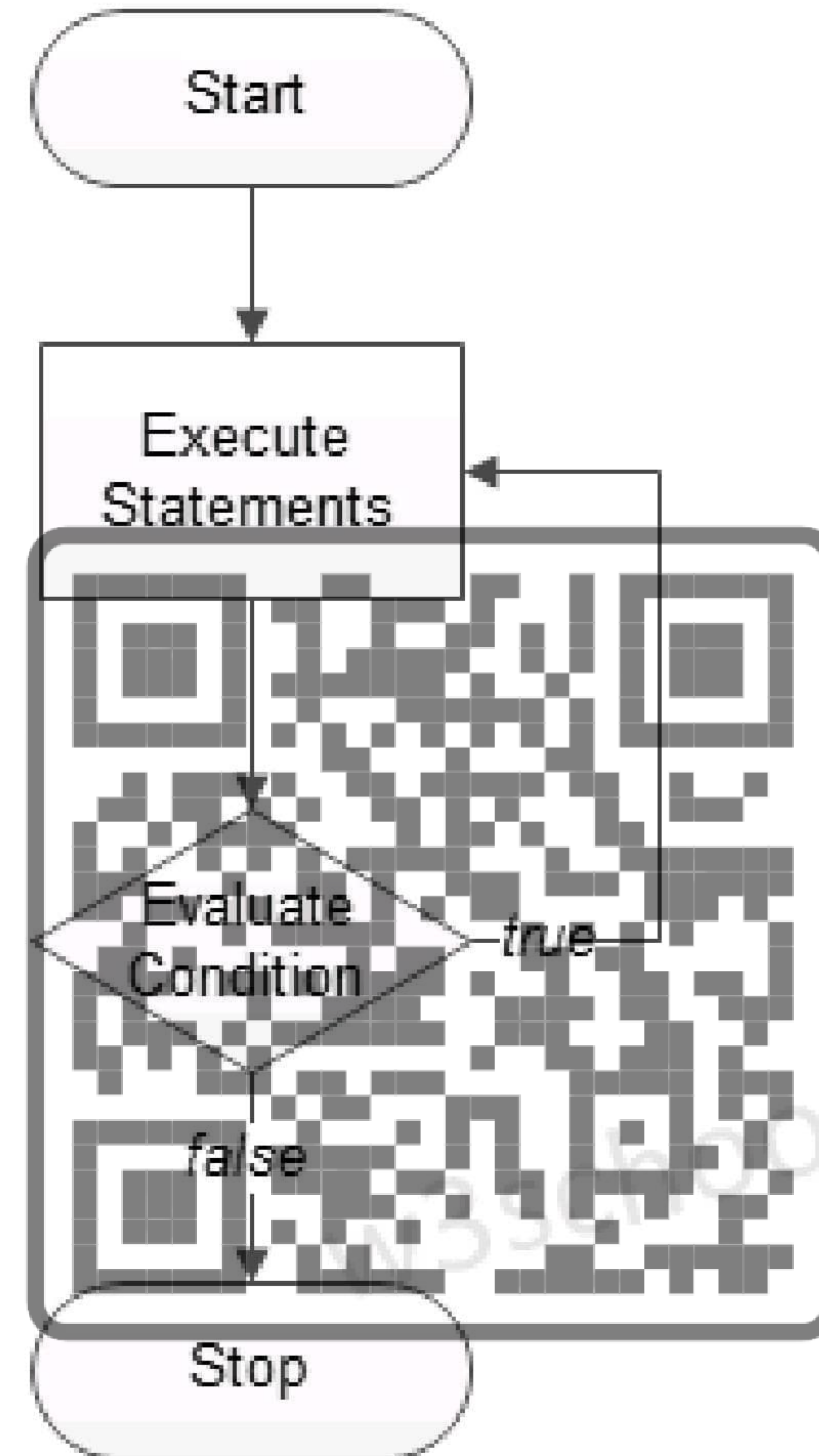
The syntax of
a **do...while** loop in C
programming language is –

Do

```
{  
statement(s);  
}
```

while(condition);

**JOIN
FOR
MORE!!!**

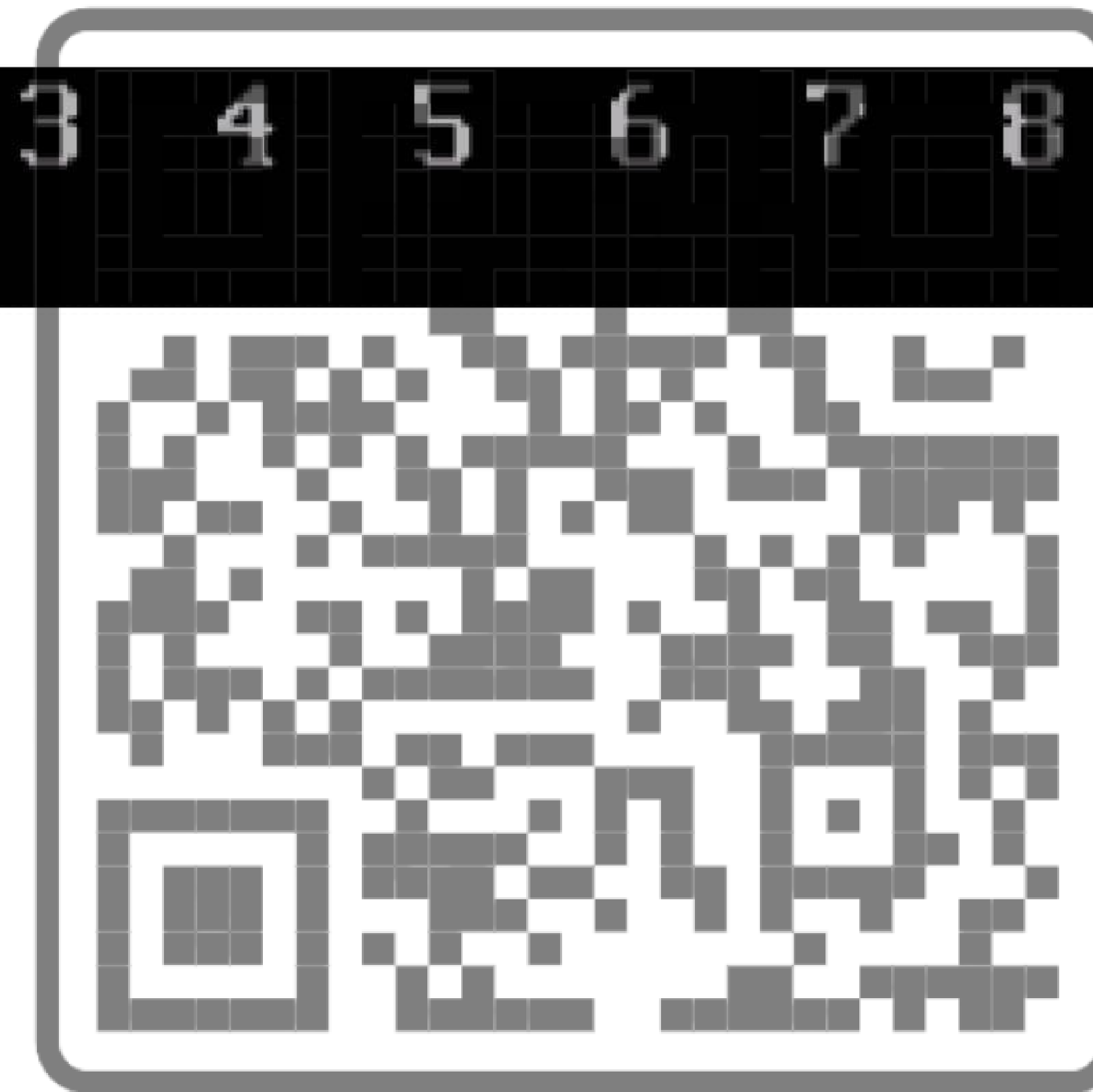


Example: : this program uses a while loop to print number from 1 to 10.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int count=1;
    do
    {
        printf("%3d",count);
        count++;
    }
    while(count<=10);
    getch();
}
```

JOIN
FOR
MORE!!!

1 2 3 4 5 6 7 8 9 10

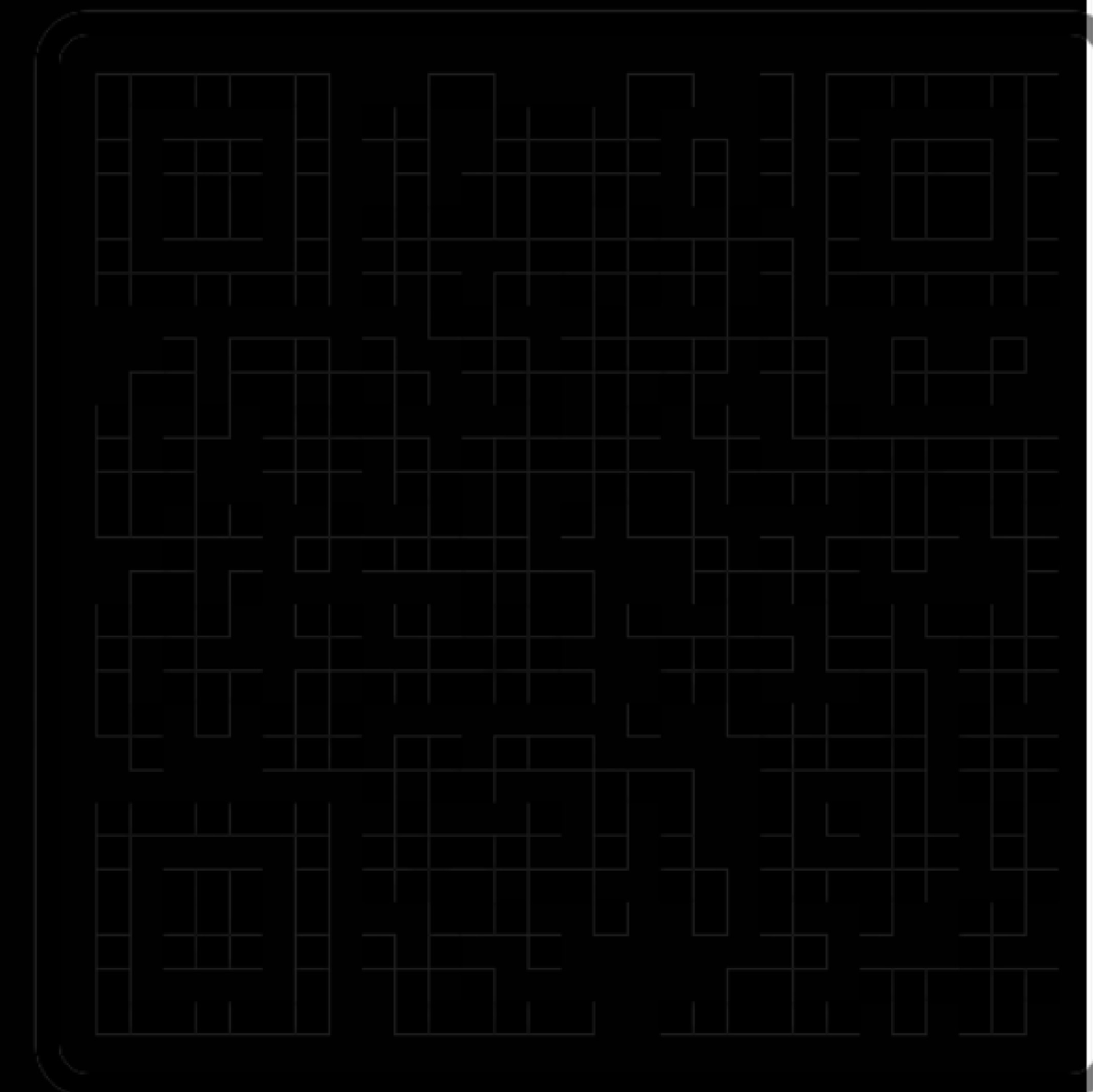


Example: user input to print name

```
void main()
{
    int count=1,n;
    clrscr();
    printf("enter a number");
    scanf("%d",&n);
    do
    {
        printf("Sana\n");
        count++;
    }
    while(count<=n);
    getch();
}
```

enter a number 10

Sana
Sana
Sana
Sana
Sana
Sana
Sana
Sana
Sana
Sana



Nested do while loop:

- **do-while loop** within **do-while loops** is said to be **nested do while loop**. The following example program will clearly explain the concept of **nested do while**.

**JOIN
FOR
MORE!!!**

```
Do
{
statement n;
do
{
statement n;
}
while(test condition);
}
while(test expression);
```



examples

```
void main()
{
  int count=1,n;
  clrscr();
  do
  { n=1;
  do
  {
    printf("%d",n);
    n++;
  }
  while (n<5);
  printf("\n");
  count++;
}
while(count<5);
getch();
}
```

1234
1234
1234
1234

FOR
MORE!!!




```
void main()
{
    int count=1,n;
    clrscr();
    do
    { n=1;
    do
    {
        printf("%6d",n*count);
        n++;
    }
    while (n<10);
    printf("\n");
    count++;
    }
    while(count<10);
    getch();
}
```

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81



BASIS FOR COMPARISON	WHILE	DO-WHILE
General Form	<pre>while (condition) { statements; //body of loop }</pre>	<pre>do{ . statements; // body of loop. } while(Condition);</pre>
Controlling Condition	In 'while' loop the controlling condition appears at the start of the loop.	In 'do-while' loop the controlling condition appears at the end of the loop.
Iterations	The iterations do not occur if, the condition at the first iteration, appears false.	The iteration occurs at least once even if the condition is false at the first iteration.



Comparison between while and do while statements.

while	do...while
This is the top tested loop	This is the bottom tested loop
The condition is first tested, if the condition is true then the block is executed until the condition becomes false.	It executes the body once, after it checks the condition, if it is true the body is executed until the condition
Loop will not be executed if the condition is false.	Loop will be executed atleast once even though the condition is false.



Past year questions with answers chap#09

Define the following functions. (2018)

Pow() sqrt() abs()

The **Abs** function returns the non-negative value of its argument. If a number is negative, **Abs** returns the positive equivalent.

The **Power** function returns a number raised to a power. It is equivalent to using the ^ [operator](#).

The **Sqrt** function returns the number that, when multiplied by itself, equals its argument.

Write a C program to calculate the power of numbers from 1 to 5.

JOIN FOR MORE!!!

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
int n=1;
while(n<6)
{
printf(" %3d is %3d\n",n,n*n);
n++;
}
getch();
}
```

(2018)



1	is	1
2	is	4
3	is	9
4	is	16
5	is	25

Define function. What are the main characteristics of a functions? (2018) and (2011)

Functions:

One of the most fundamental ideas in problems solving is dividing complex problem into small components makes program easy to understand and use. A function is a block of code that performs a specific task. Every C program has at least one function, main (). Suppose, a program related to graphics needs to create a circle and colour it is depending upon the radius and colour from the user. You can create two functions to solve this problem:

- create a circle function
- colour function

Characteristics of Functions/ Advantages of Functions:

- The length of a source program can be reduced by using functions at appropriate places.
- The program will be easier to understand, maintain and debug.
- Reusable codes that can be used in other programs
- A function may be used by many other programs. This means that a C programmer can build on what others have already done, instead of starting all over again from scratch.
- Functions can have inputs and outputs and can process information
- Functions help us to avoid unnecessary repetition of code.

What do you mean by function prototype? (2017)

Using functions in your program requires that you first declare the function and then use define the function. No function can be called from any other function that has not first been declared. The declaration of a function is called prototype. It is a line written before the main() function. The prototype tells the compiler in advance about some characteristics of a function used in the program.

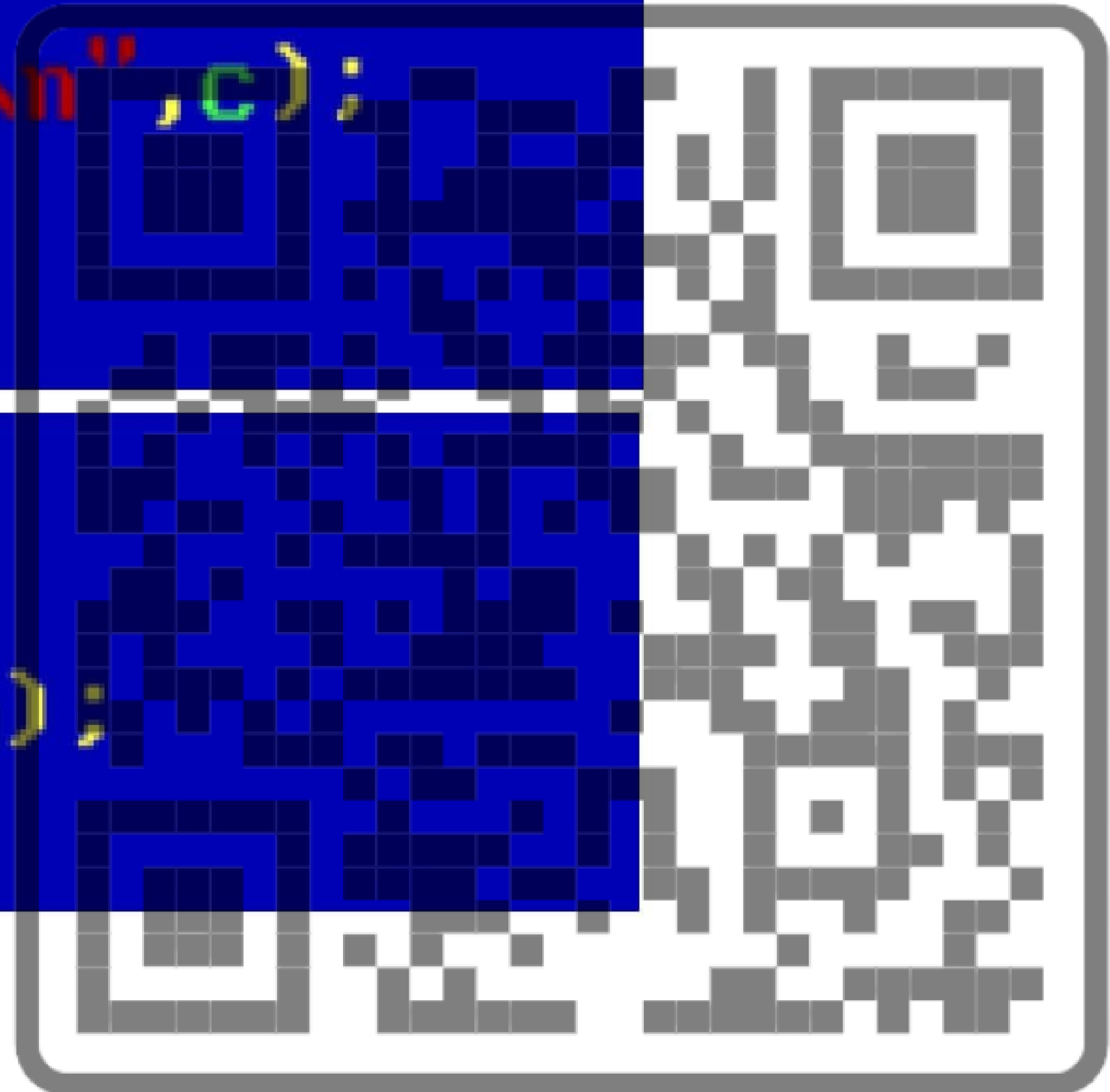


Write a C program to call a function which returns the cube of a given number. (2015)

```
#include<stdio.h>
#include<conio.h>
int cube(int num);
void main()
{
clrscr();
int num,c;
printf("enter number\n");
scanf("%d",&num);
c=cube(num);
printf("cube is %d:\n",c);
getch();
}

int cube(int num)
{
return(num*num*num);
}
```

```
enter number
3
cube is 27:
_
```



What are library functions? Discuss any three library functions. (2010)

Library functions:

Library functions in C language are inbuilt functions which are grouped together and placed



in a common place called library. Each library function in C performs specific operation. We can make use of these library functions to get the pre-defined **output** instead of writing our own code to get those outputs.

sqrt()

In C programming you can find the square root by just using `sqrt()` function which is defined under header file `<math.h>`

Abs():

The **Abs** function returns the non-negative value of its argument. If a number is negative, **Abs** returns the positive equivalent. This function which is defined under header file `<math.h>`

Pow()

The **Power** function returns a number raised to a power. It is equivalent to using the [^operator](#). This function which is defined under header file `<math.h>`

**JOIN
FOR
MORE!!!**



Chapter# 14 lecture Notes:

Data Files Introduction:

Many applications require that information be written to or read from an auxiliary storage device. Such information is stored on the storage device in the form of data file. Thus, data files allow us to store information permanently and to access later and alter that information whenever necessary. In C, many library functions are available for creating and processing data files.

Why Data Files Create In C:

This C Program creates a file & store information. We frequently use files for storing information which can be processed by our programs. In order to store information permanently and retrieve it we need to use files and this program demonstrate file creation and writing data in that.

Declaring a file:

A file can be treated as an external storage. It consists of a sequence of bytes residing on the disk. Group of related data can be stored in a single file. A program can create, read and write to a file. Unlike an array, the data in the file is retained even after the program finishes its execution. It's a permanent storage medium.

Declaring a file pointer

In C language, to declare a file, we use a file pointer. A file pointer is a pointer variable that specifies the next byte to be read or written to. Every time a file is opened, the file pointer points to the beginning of the file. A file is declared as follows:

```
FILE *fp; //fp is the name of the file pointer
```

OPENING AND CLOSING A DATA FILE

We must open the file before we can write information to a file on a disk or read it. Opening a file establishes a link between the program and the operating system. The link between our program and the operating system is a structure called FILE which has been defined in the header file "stdio.h". Therefore, it is always necessary to include this file when we do high level disk I/O. When we use a command to open a file, it will return a pointer to the structure FILE. Therefore, the following declaration will be there before opening the file, FILE *fp → each file will have its own FILE structure. The FILE structure contains information about the file being used, such as its current size, its location in memory etc. Let us consider the following statements,

FILE *fp;

fp=fopen("Sample.C," "r");

fp is a pointer variable, which contains the address of the structure FILE which has been defined in the header file "stdio.h". fopen() will open a file "sample.c" in 'read' mode, which tells the C compiler that we would be reading the contents of the file. Here, "r" is a string and not a character. When fopen() is used to open a file then, it searches on the disk the file to be opened. If the file is present, it loads the file from the disk into memory. But if the file is absent, fopen() returns a NULL.

'C' provides many different file opening modes which are as follows:

1. "r" Open the file for reading only.
2. "w" Open the file for writing only.
3. "a" Open the file for appending (or adding) data to it.
4. "r+" The existing file is opened to the beginning for both reading and writing.
5. "w+" Same as "w" except both for reading and writing.
6. "a+" Same as "a" except both for reading and writing.

File handling functions	Description
fopen ()	fopen () function creates a new file or opens an existing file.
fclose ()	fclose () function closes an opened file.
fprintf ()	fprintf () function writes formatted data to a file.
fscanf ()	fscanf () function reads formatted data from a file.

Error Checking:

The fopen() function returns NULL if there was an error while trying to open a file. Your program may not be able to open the file you specify in the fopen() command. There are

many possible reasons for the error. The disk drive door might be open, the disk could be full, the filename may be too long, the file may be in another directory, and so on.

If you attempt to use file that did not open, your program will stop and returns NULL. To prevent from occurring, use an if command that stops the program when the file cannot be opened.

Example of write file:



Example:

```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
FILE*sp;
int roll;
char name[25];
sp=fopen("studentt.txt","w");
printf("enter roll no");
scanf("%d",&roll);
printf("enter student name");
sp=fopen("studentt.txt","r");
fscanf(sp,"%d %s",roll,name);
printf("Student roll no: %d\n",roll);
printf("Student name : %s\n",name);
fclose(sp);
getch();
}
```




```
enter roll no 1234
enter student name madiha
Student roll no: 1234
Student name : madiha
```



Chapter# 14Data Files Past paper questions with answers2nd year

Define different basic modes opening modes used to open a data file in C language. 2018

Define data file. Describe any two types of data files used in C program. 2016

**JOIN
FOR
MORE!!!**



Chapter #15 lecture

Introduction to Database Management System

What is Data?

Data is nothing, but facts and statistics stored or free flowing over a network, generally it's raw and unprocessed. For example: When you visit any website, they might store you IP address, that is data, in return they might add a cookie in your browser, marking you that you visited the website, that is data, your name, it's data, your age, it's data. Data becomes **information** when it is processed, turning it into something meaningful

What is database?

A **Database** is a collection of related data organised in a way that data can be easily accessed, managed and updated. A database is one of the important components for many applications and is used for storing a series of data in a single set. In other words, it is a group/package of information that is put in order so that it can be easily accessed, manage and update. In a database, even the smallest portion of information becomes the data. Example, Student is a data, roll number is a data, and the address is a data, height, weight, marks everything is data. In brief, all the living and non-living objects in this world is a data.

What is DBMS?

A **DBMS** is a software that allows creation, definition and manipulation of database, allowing users to store, process and analyse data easily. DBMS provides us with an interface or a tool, to perform various operations like creating database, storing data in it, updating data, creating tables in the database and a lot more.

DBMS also provides protection and security to the databases. It also maintains data consistency in case of multiple users.

Here are some examples of popular DBMS used these days:

- MySql
- Oracle
- SQL Server





Users

A typical DBMS has users with different rights and permissions who use it for different purposes. Some users retrieve data and some back it up. The users of a DBMS can be broadly categorized as follows:

- **Administrators:** Administrators maintain the DBMS and are responsible for administering the database. They are responsible to look after its usage and by whom it should be used. They create access profiles for users and apply limitations to maintain isolation and force security. Administrators also look after DBMS resources like system license, required tools, and other software and hardware related maintenance.
- **Designers:** Designers are the group of people who actually work on the designing part of the database. They keep a close watch on what data should be kept and in what format. They identify and design the whole set of entities, relations, constraints, and views.
- **End Users:** End users are those who actually reap the benefits of having a DBMS. End users can range from simple viewers who pay attention to the logs or market rates to sophisticated users such as business analysts.



Character:

The basic building block of data is a character. A character sometimes called byte consist of letters (A, B, C.....Z or a, b, c,.....z), numeric digits (0,1,2,3,....9) or special characters (+,-,#,/,*etc).

Entity:

An **entity** is any object in the system that we want to model and store information about. **Entities** are usually recognizable concepts, either concrete or abstract, such as person, places, things, or events which have relevance to the **database**. Some specific examples of **entities** are Employee, Student, Lecturer.

Each table is about an object, person or thing.

- Customer
- Videos
- Appointments
- Patients
- Books
- Exam sessions
- Products
- Teachers

Field or /

Fields

These are known as 'entities'

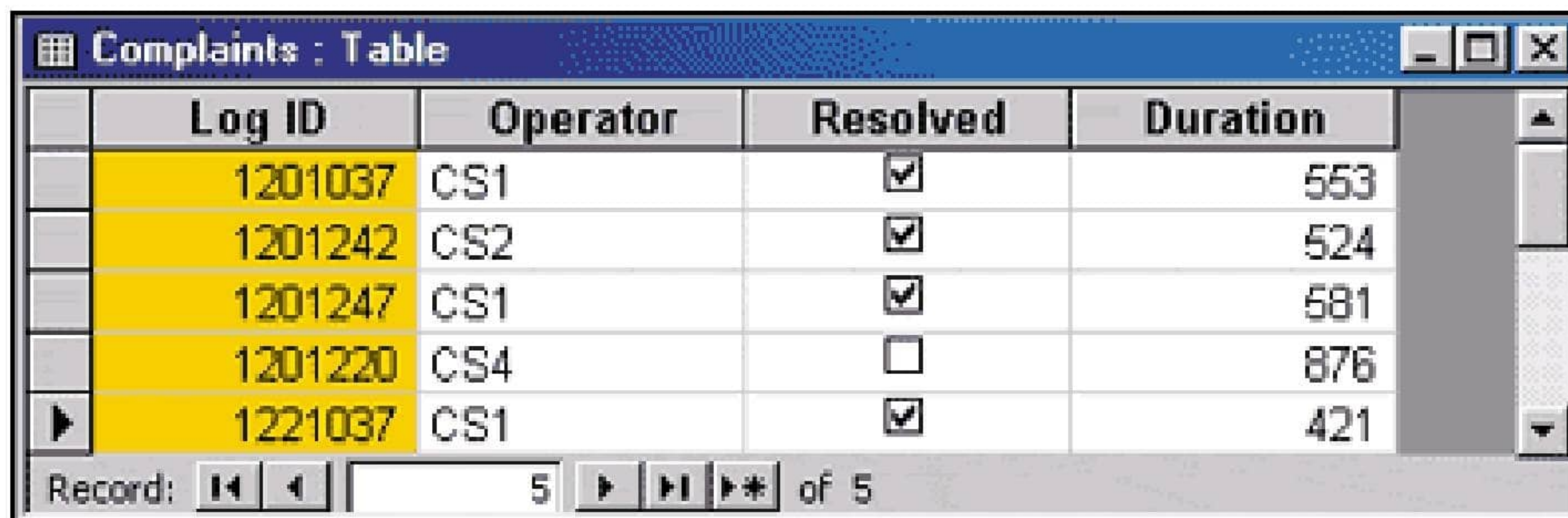


A field is part of a record and contains a single piece of data for the subject of the record. In the database table illustrated in Figure , each record contains four fields:

Log ID	A number assigned to this customer support incident for identification purposes
Operator	The code for the customer support operator who handled this incident
Resolved	A check box to indicate whether the incident was resolved
Duration	The time in seconds the operator spent on this incident

Fields appear as columns in a database table. Data from the Log ID field for five records is highlighted in the Figure



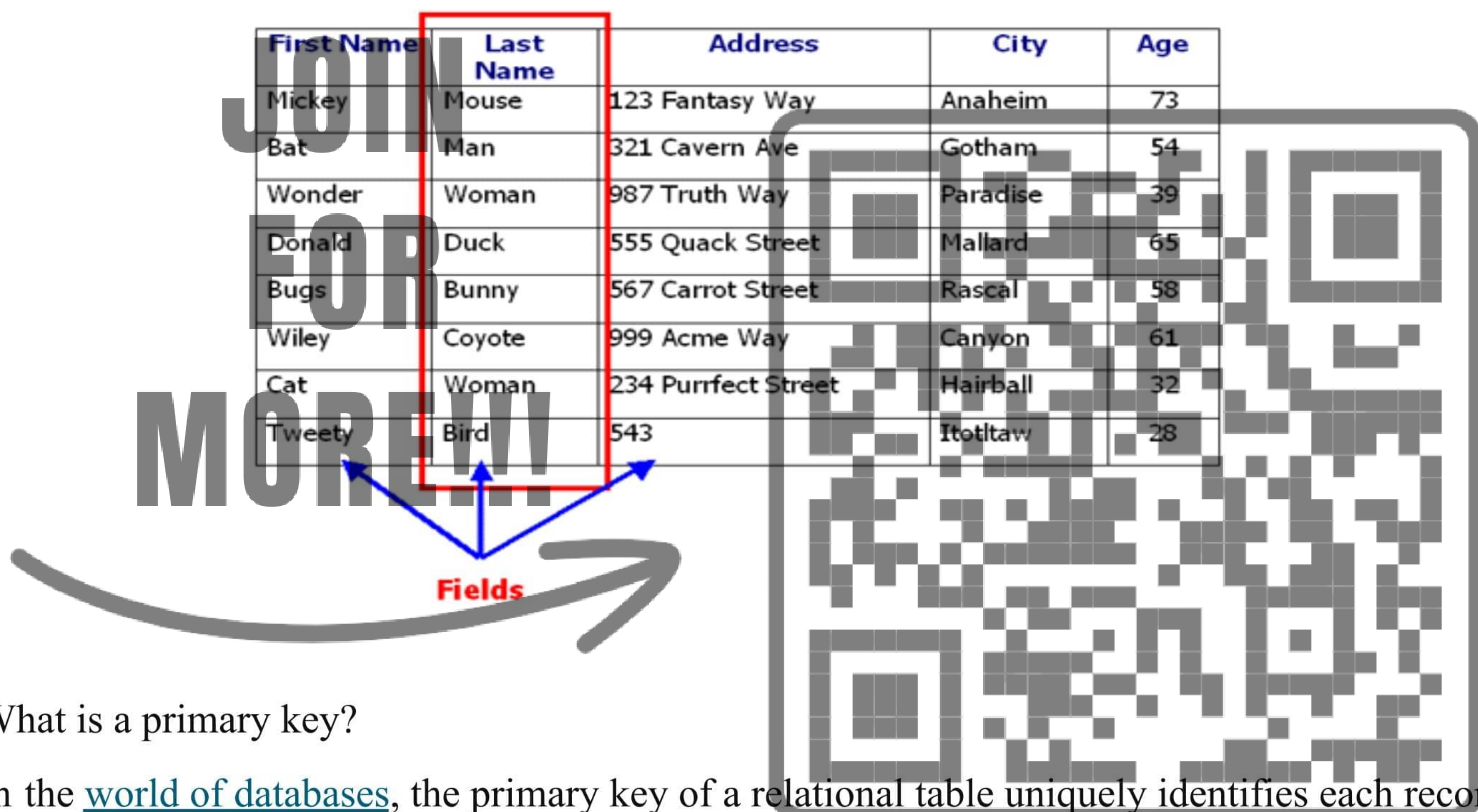


Log ID	Operator	Resolved	Duration
1201037	CS1	☒	553
1201242	CS2	☒	524
1201247	CS1	☒	581
1201220	CS4	☐	876
1221037	CS1	☒	421

Figure Fields appear as columns in a database table.

Key Field:

A **field** in a record that holds unique data which identifies that record from all the other records in the file or **database**. Account number, product code and customer name are typical **key fields**. As an identifier, each **key** value must be unique in each record.



First Name	Last Name	Address	City	Age
Mickey	Mouse	123 Fantasy Way	Anaheim	73
Bat	Man	321 Cavern Ave	Gotham	54
Wonder	Woman	987 Truth Way	Paradise	39
Donald	Duck	555 Quack Street	Mallard	65
Bugs	Bunny	567 Carrot Street	Rascal	58
Wiley	Coyote	999 Acme Way	Canyon	61
Cat	Woman	234 Purrfect Street	Hairball	32
Tweety	Bird	543	Itotitav	28

What is a primary key?

In the [world of databases](#), the primary key of a relational table uniquely identifies each record in the table. Databases use keys to compare, sort, and store records, and to create relationships between records. A **primary key** is a minimal set of attributes (columns) in a table that uniquely identifies tuples (rows) in that table.

- Choosing the primary key in a database is one of the most important steps in the process.
- The value of primary key should be unique for each row of the table. The column(s) that makes the key cannot contain duplicate values.
- The attribute(s) that is marked as primary key is not allowed to have null values.



- Primary keys are not necessarily to be a single attribute (column). It can be a set of more than one attributes (columns).

Primary Keys



<u>StudentId</u>	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042
L0023487	Peter	Murray	P301
L0018453	Anne	Norris	S042

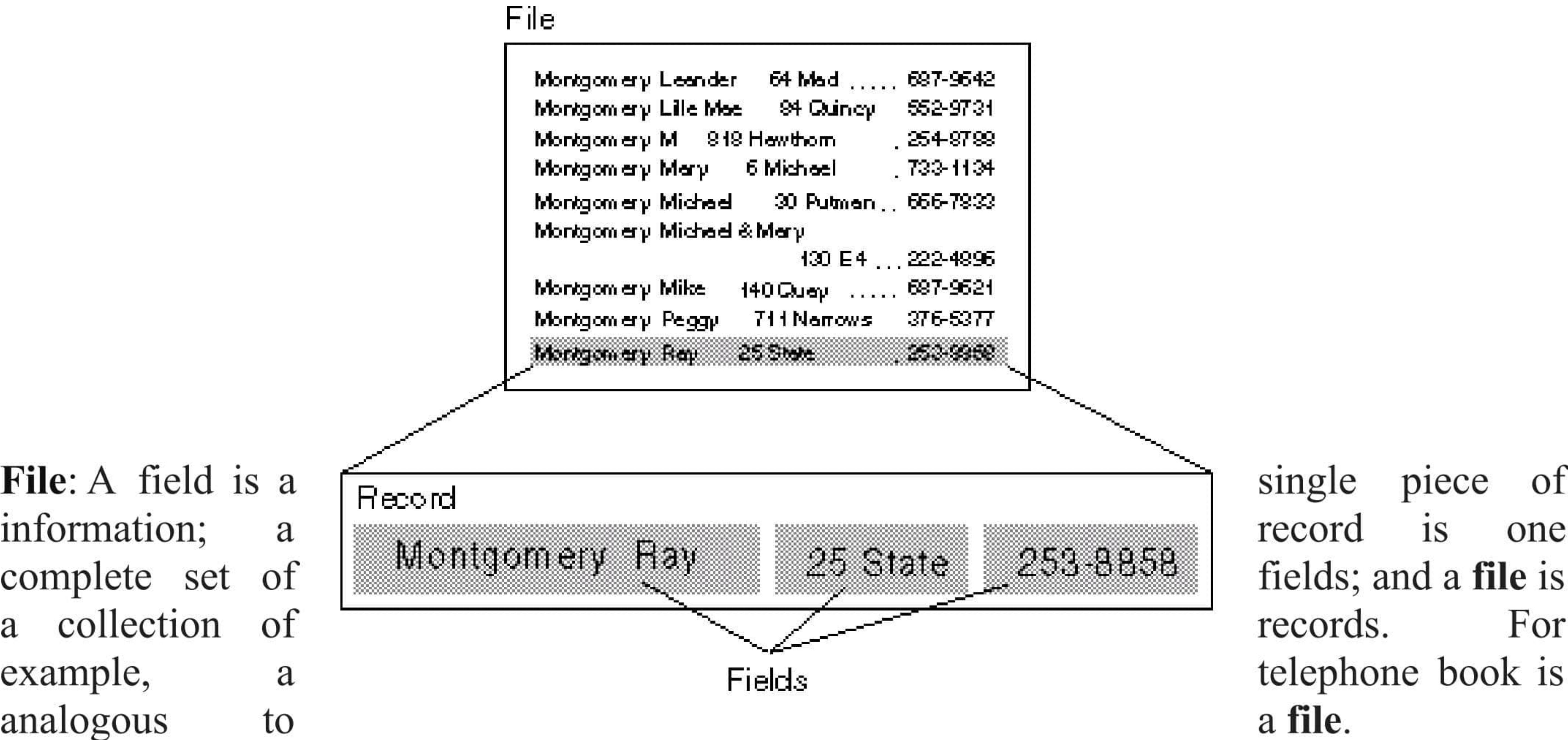
Secondary key:

An entity may have one or more choices for the primary key. Collectively these are known as candidate keys. One is selected as the primary key. Those not selected are known as **secondary keys**. For example, an employee has an employee number, a National Insurance (NI) number and an email address. If the employee number is chosen as the primary key, then the NI number and email address are secondary keys. However, it is important to note that if any employee does not have a NI number or email address (i.e: the attribute is not mandatory) then it cannot be chosen as a primary key.

Records

Records are composed of fields, each of which contains one item of information. A set of **records** constitutes a file. For example, a personnel file might contain **records** that have three fields: a name field, an address field, and a phone number field. In relational **database** management systems, **records** are called tuples.





Advantages of Database Management System (DBMS)

1. Improved data sharing

An advantage of the database management approach is, the DBMS helps to create an environment in which end users have better access to more and better-managed data. Such access makes it possible for end users to respond quickly to changes in their environment.

2. Improved data security

The more users access the data, the greater the risks of data security breaches. Corporations invest considerable amounts of time, effort, and money to ensure that corporate data are used properly. A DBMS provides a framework for better enforcement of data privacy and security policies.

3. Better data integration

Wider access to well-managed data promotes an integrated view of the organization's operations and a clearer view of the big picture. It becomes much easier to see how actions in one segment of the company affect other segments.



4. Minimized data inconsistency

Data inconsistency exists when different versions of the same data appear in different places. For example, data inconsistency exists when a company's sales department stores a sales representative's name as "Bill Brown" and the company's personnel department stores that same person's name as "William G. Brown," or when the company's regional sales office shows the price of a product as \$45.95 and its national sales office shows the same product's price as \$43.95. The probability of data inconsistency is greatly reduced in a properly designed database.

5. Improved data access

The DBMS makes it possible to produce quick answers to ad hoc queries. From a database perspective, a query is a specific request issued to the DBMS for data manipulation—for example, to read or update the data. Simply put, a query is a question, and an ad hoc query is a spur-of-the-moment question. The DBMS sends back an answer (called the query result set) to the application. For example, end users, when dealing with large amounts of sales data, might want quick answers to questions (ad hoc queries) such as:

- What was the dollar volume of sales by product during the past six months?
- What is the sales bonus figure for each of our salespeople during the past three months?
- How many of our customers have credit balances of 3,000 or more?

6. Improved decision making

Better-managed data and improved data access make it possible to generate better-quality information, on which better decisions are based. The quality of the information generated depends on the quality of the underlying data. Data quality is a comprehensive approach to promoting the accuracy, validity, and timeliness of the data. While the DBMS does not guarantee data quality, it provides a framework to facilitate data quality initiatives.

7. Increased end-user productivity

The availability of data, combined with the tools that transform data into usable information, empowers end users to make quick, informed decisions that can make the difference between success and failure in the global economy.



Disadvantages of Database Management System (DBMS):

Although the database system yields considerable advantages over previous data management approaches, database systems do carry significant disadvantages. For example:

1. Increased costs

one of the disadvantages of dbms is Database systems require sophisticated hardware and software and highly skilled personnel. The cost of maintaining the hardware, software, and personnel required to operate and manage a database system can be substantial. Training, licensing, and regulation compliance costs are often overlooked when database systems are implemented.

2. Management complexity

Database systems interface with many different technologies and have a significant impact on a company's resources and culture. The changes introduced by the adoption of a database system must be properly managed to ensure that they help advance the company's objectives. Given the fact that database systems hold crucial company data that are accessed from multiple sources, security issues must be assessed constantly.

3. Maintaining currency

To maximize the efficiency of the database system, you must keep your system current. Therefore, you must perform frequent updates and apply the latest patches and security measures to all components.

Because database technology advances rapidly, personnel training costs tend to be significant. Vendor dependence. Given the heavy investment in technology and personnel training, companies might be reluctant to change database vendors.

4. Frequent upgrade/replacement cycles

DBMS vendors frequently upgrade their products by adding new functionality. Such new features often come bundled in new upgrade versions of the software. Some of these versions require hardware upgrades. Not only do the upgrades themselves cost money, but it also costs money to train database users and administrators to properly use and manage the new features.



Objectives of Database

Mass Storage

DBMS can store a lot of data in it. So for all the big firms, DBMS is really ideal technology to use. It can store thousands of records in it and one can fetch all that data whenever it is needed.

Removes Duplicity

If you have lots of data, then data duplicity will occur at any instance. DBMS guarantee it that there will be no data duplicity among all the records. While storing new records, DBMS makes sure that same data was not inserted before.

Multiple Users Access

No one handles the whole database alone. There are lots of users who are able to access database. So, this situation may happen that two or more users are accessing database. They can change whatever they want, at that time DBMS makes it sure that they can work concurrently.

Data Protection

Information such as bank details, employee's salary details and sale purchase details should always be kept secured. Also, all the companies need their data secured from unauthorized use. DBMS gives a master level security to their data. No one can alter or modify the information without the privilege of using that data.

Data Backup and recovery

Sometimes database failure occurs so there is no option like one can say that all the data has been lost. There should be a backup of database so that on database failure it can be recovered. DBMS has the ability to backup and recover all the data in database.

Everyone can work on DBMS

There is no need to be a master of programming language if you want to work on DBMS. Any accountant who is having less technical knowledge can work on DBMS. All the definitions and descriptions are given in it so that even a non-technical background person can work on it.

Integrity

Integrity means your data is authentic and consistent. DBMS has various validity checks that make your data completely accurate and consistence.



Platform Independent

One can run dbms at any platform. No particular platform is required to work on database management system.

Components of DBMS

DBMS have several components, each performing very significant tasks in the database management system environment. Below is a list of components within the database and its environment.



Software

This is the set of programs used to control and manage the overall database. This includes the DBMS software itself, the Operating System, the network software being used to share the data among users, and the application programs used to access data in the DBMS.



Hardware

Consists of a set of physical electronic devices such as computers, I/O devices, storage devices, etc., this provides the interface between computers and the real-world systems.



Data

DBMS exists to collect, store, process and access data, the most important component. The database contains both the actual or operational data and the metadata.



Procedures

These are the instructions and rules that assist on how to use the DBMS, and in designing and running the database, using documented procedures, to guide the users that operate and manage it.



Database Access Language

This is used to access the data to and from the database, to enter new data, update existing data, or retrieve required data from databases. The user writes a set of appropriate commands

in a database access language, submits these to the DBMS, which then processes the data and generates and displays a set of results into a user readable form.



Query Processor

This transforms the user queries into a series of low level instructions. This reads the online user's query and translates it into an efficient series of operations in a form capable of being sent to the run time data manager for execution.



Run Time Database Manager

Sometimes referred to as the database control system, this is the central software component of the DBMS that interfaces with user-submitted application programs and queries and handles database access at run time. Its function is to convert operations in user's queries. It provides control to maintain the consistency, integrity and security of the data.



Data Manager

Also called the cache manger, this is responsible for handling of data in the database, providing a recovery to the system that allows it to recover the data after a failure.



Database Engine

The core service for storing, processing, and securing data, this provides controlled access and rapid transaction processing to address the requirements of the most demanding data consuming applications. It is often used to create relational databases for online transaction processing or online analytical processing data.



Data Dictionary

This is a reserved space within a database used to store information about the database itself. A data dictionary is a set of read-only table and views, containing the different information about the data used in the enterprise to ensure that database representation of the data follow one standard as defined in the dictionary.



Report Writer

Also referred to as the report generator, it is a program that extracts information from one or more files and presents the information in a specified format. Most report writers allow the user to select records that meet certain conditions and to display selected fields in rows and columns, or also format the data into different charts.

Types of database models

There are many kinds of data models. Some of the most common ones include:

- Hierarchical database model
- Relational model
- Network model

Relational model

The most common model, the relational model sorts data into tables, also known as relations, each of which consists of columns and rows. Each column lists an attribute of the entity in question, such as price, zip code, or birth date. Together, the attributes in a relation are called a domain. A particular attribute or combination of attributes is chosen as a primary key that can be referred to in other tables, when it's called a foreign key.

Each row, also called a tuple, includes data about a specific instance of the entity in question, such as a particular employee.

The model also accounts for the types of relationships between those tables, including one-to-one, one-to-many, and many-to-many relationships. Here's an example:



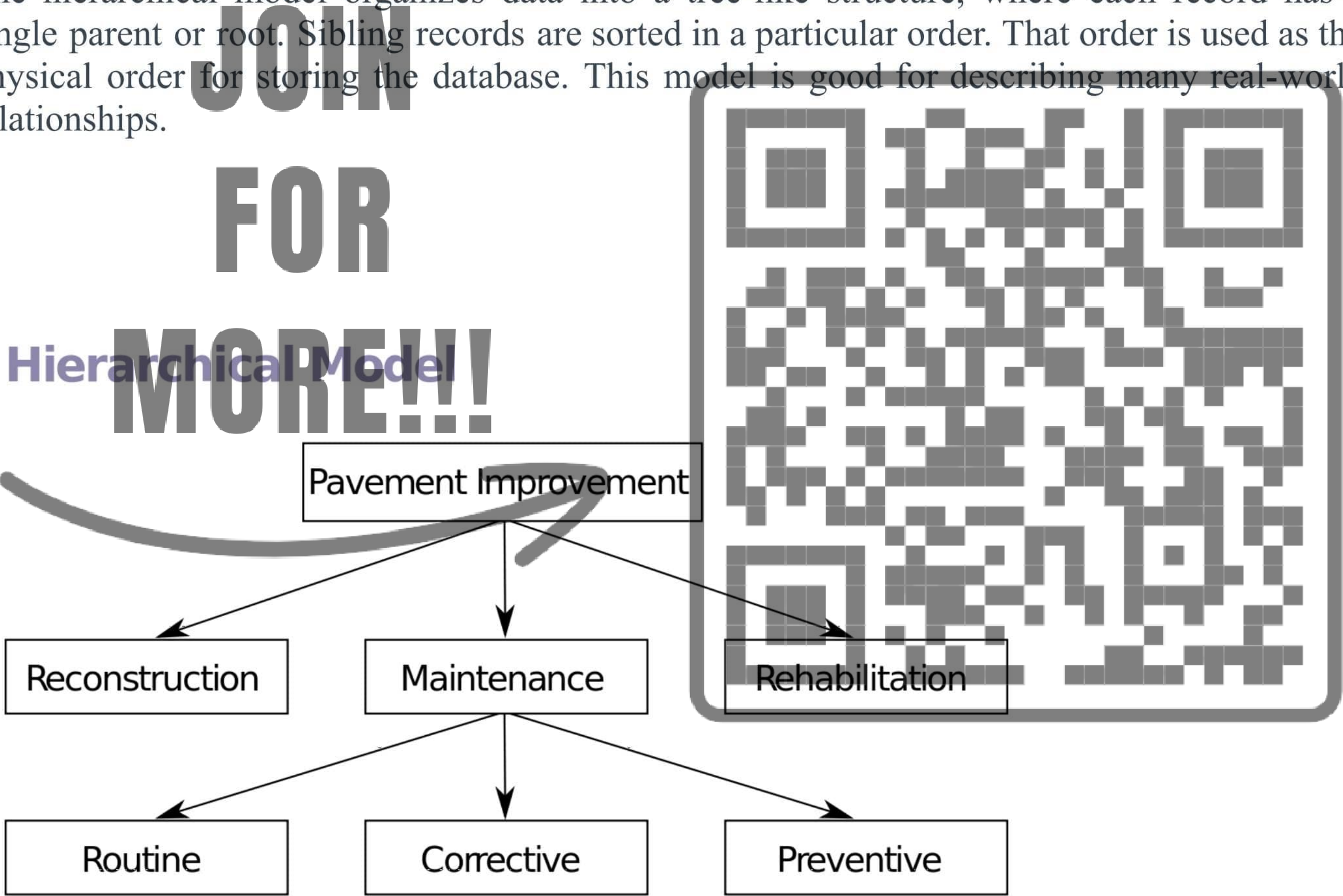
Student ID	First name	Last name
52-743965	Charles	Peters
48-209689	Anthony	Sondrup
14-204968	Rebecca	Phillips

ProviderID	Provider name
156-983	UnitedHealth
146-823	Blue Shield
447-784	Carolina Inc.

Student ID	ProviderID	Type of plan	Start date
52-743965	156-983	HSA	04/01/2016
48-209689	146-823	HMO	12/01/2015
14-204968	447-784	HSA	03/14/2016

Hierarchical model

The hierarchical model organizes data into a tree-like structure, where each record has a single parent or root. Sibling records are sorted in a particular order. That order is used as the physical order for storing the database. This model is good for describing many real-world relationships.



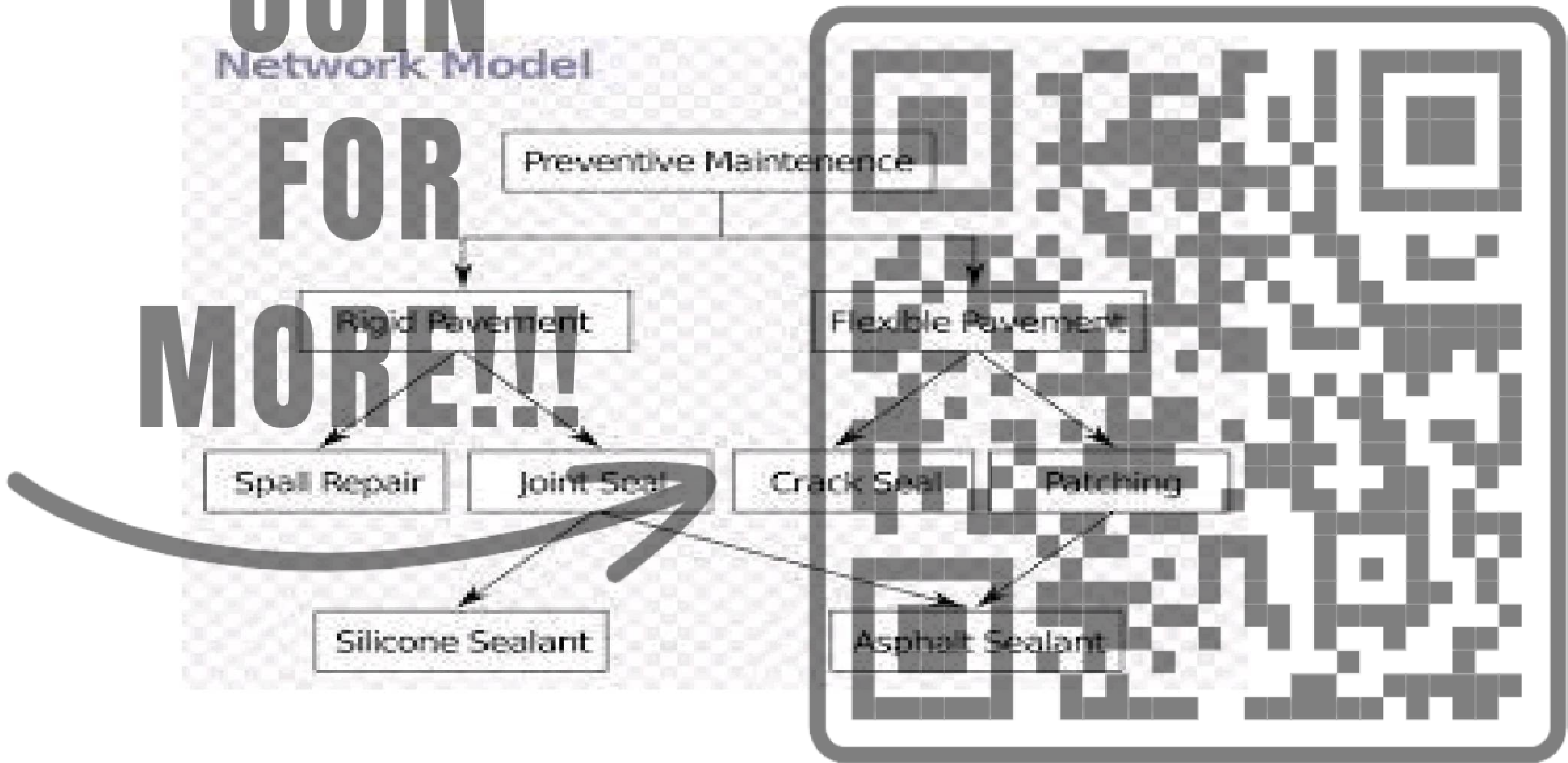
Network model

The network model builds on the hierarchical model by allowing many-to-many relationships between linked records, implying multiple parent records. Based on mathematical set theory, the model is constructed with sets of related records. Each set consists of one owner or parent record and one or more member or child records. A record can be a member or child in multiple sets, allowing this model to convey complex relationships.



WHAT IS A RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)?

A relational database management system (RDBMS) is a program that allows you to create, update, and administer a relational database. Most relational database management systems use the SQL language to access the database.



Chapter # 15 past papers with answers 2nd year

1. What is the difference between DBMS and RDBMS? 2018
2. Write the full form of DCI, IDE, DBMS. 2018
3. What is composite key? Where do we use? 2018
4. How can we define one to one relationship in database? 2018
5. Why DBMS necessary for any big organization? 2018
6. In which condition is one to many relationships is used? 2017
7. Define network Model. where do we use? 2017
8. Define any four objects of DBMS. 2017
9. What are the responsibilities of DBA? 2016
10. What is database? What are the advantages of relational database management system (RDBMS)? 2016
11. List some advantages of DBMS? 2015
12. Write full form of DML, DDL, SQL, IDE, RDBMS, DBMS. 2015
13. What is a key filed? Why is it important in DBMS? 2015
14. Explain various types of database Model? 2015
15. Define database Model with diagram. 2013
16. Elaborate the term primary key with its characteristics. 2011
17. Elucidate the term database model. Explain relational and database model. 2011
18. Explain how primary and foreign key are used in database management system? 2009
19. Define relational database model. 2018
20. What is data dictionary in DBMS. 2017

What is the difference between DBMS and RDBMS? 2018

Difference	
DBMS	RDBMS
1. In DBMS relationship between two table or file are maintained programmatically. 2. DBMS does not support client-server architecture. 3. DBMS does not support distributed. 4. No security.	1. In RDBMS relationship between two table or file can be specified at the time of table creation. 2. RDBMS supports client-server architecture. 3. RDBMS supports distributed. 4. Multiple level of security a) Log in at O.S. level b) Commend level c) Object level



Chapter # 15 past papers with answers2nd year

Write the full form of DCI, IDE, DBMS. 2018

IDE: integrated development environment

DBMS: database management system

DCI: data control language

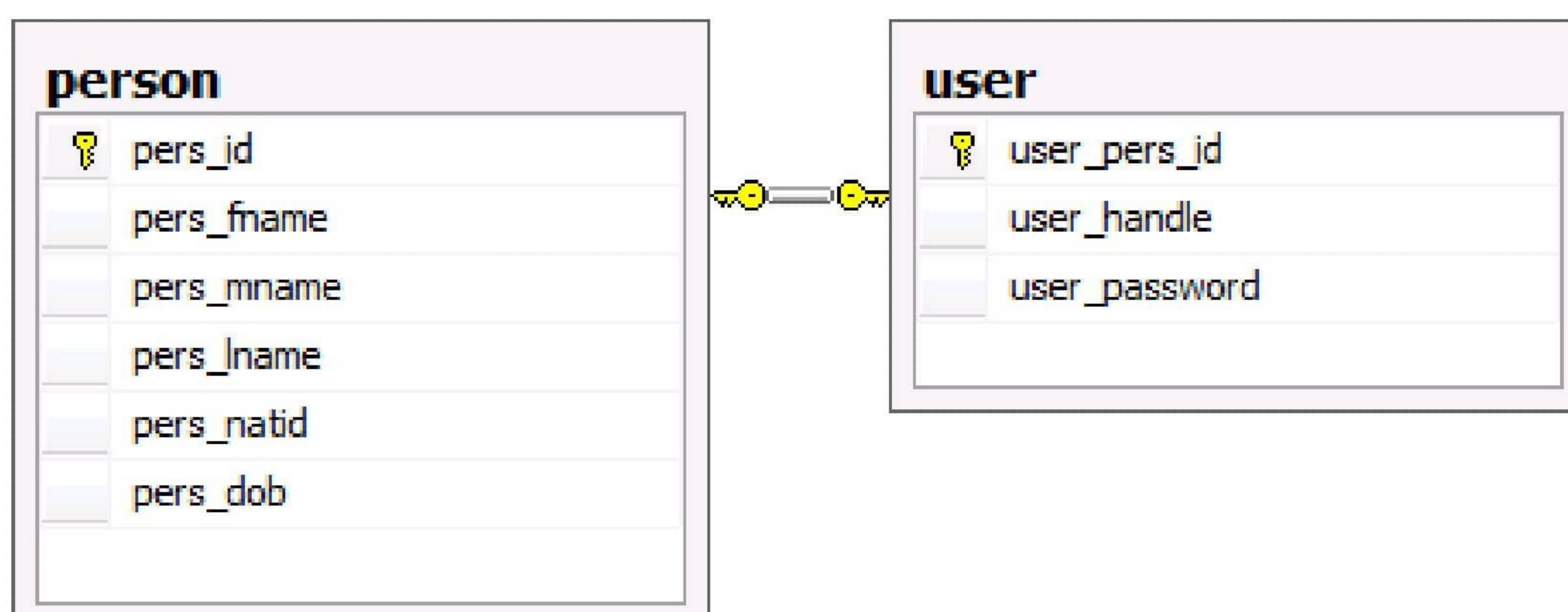
What is composite key? Where do we use?

Any column(s) that can guarantee uniqueness is called a candidate key; however, a composite key is a special type of candidate key that is only formed by a combination of two or more columns. Sometimes the candidate key is just a single column, and sometimes it is formed by joining multiple columns.

Consider an example of a certain table in the database of a commercial bank. This table is used to store records of individuals' bank accounts. Assuming that the table has separate columns for the account type (C for checking, S for savings and so on), followed by another column for the year and month of the account's creation, and another column for a sequential number within that month, it is obvious that any one of these columns by itself cannot identify an account – one can deduce that there would be several C's in the "Account Type" column, there would be several entries for May 2008 in the "Date of Creation" column, and so on. However, if all three columns are combined, then a unique record for each and every account is produced. A hypothetical account number in this example would be "C 200807 001" for the first account created in July 2008, which is a checking account. Another is "S 201003 004" for the fourth savings account created in March 2010. This is a composite key, that is, a candidate key that guarantees uniqueness only when two or more columns are joined together.

How can we define one to one relationship in database? 2018

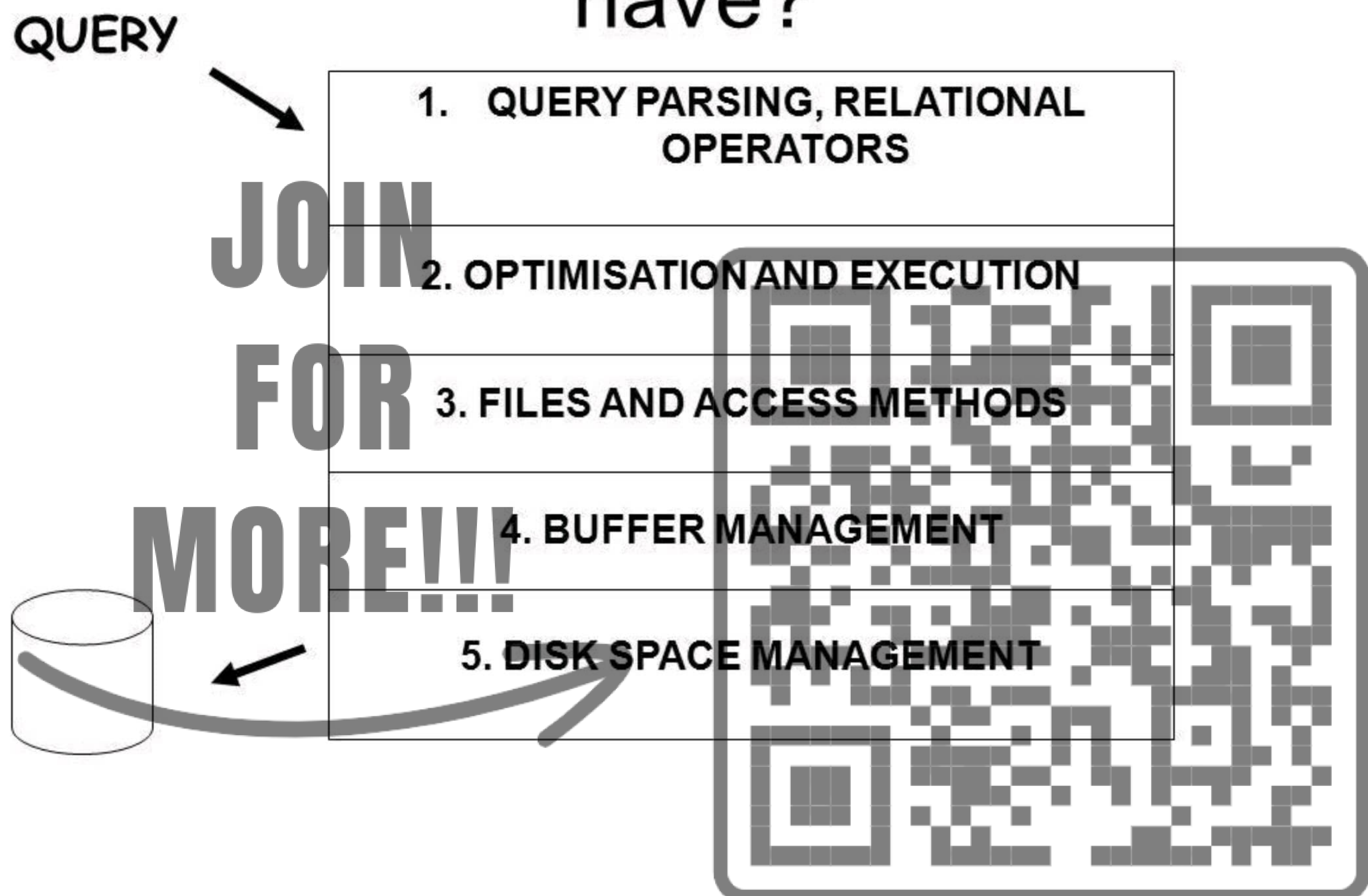
A **"One to one" relationship** between two tables means that a row in **one** table can only relate to **one** row in the table on the other side of their **relationship** and vice versa. This is the least common **database relationship**. **One to One relationships** are the least common **database relationship**.



Why DBMS necessary for any big organization?2018

Thousands of companies depend on the accurate recording, updating and tracking of their data on a minute-to-minute basis. Employees use this data to complete accounting reports, calculate sales estimates and invoice customers. The workers access this data through a computerized database. A proven method to manage the relationships between the various database elements is the use of a relational database management system.

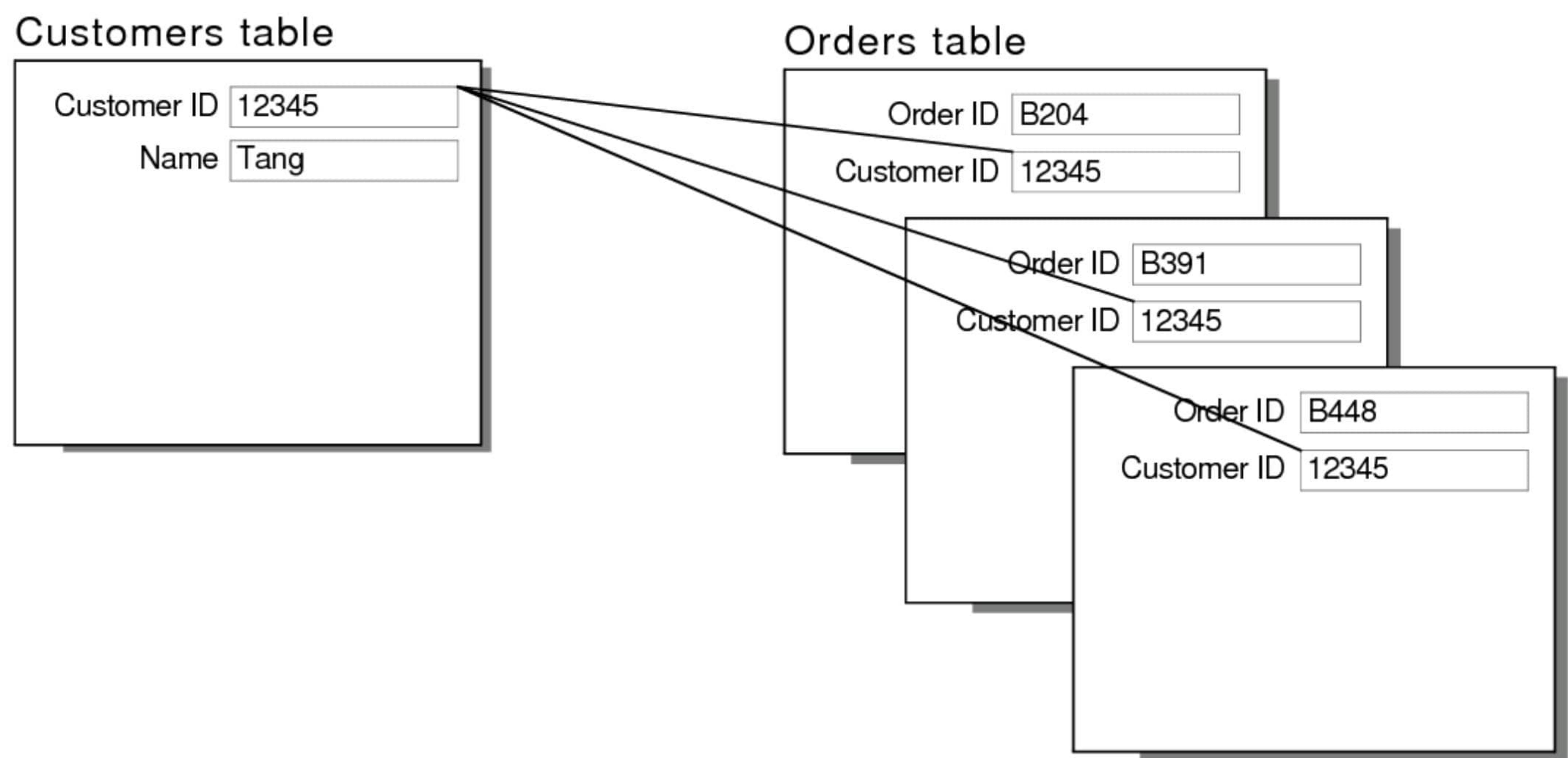
What role does the DBMS have?



In which condition is one to many relationships is used?

In relational databases, a one-to-many relationship occurs when a parent record in one table can potentially reference several child records in another table. In a one-to-many relationship, the parent is not required to have child records; therefore, the one-to-many relationship allows zero child records, a single child record or multiple child records. The important thing is that the child cannot have more than one parent record.





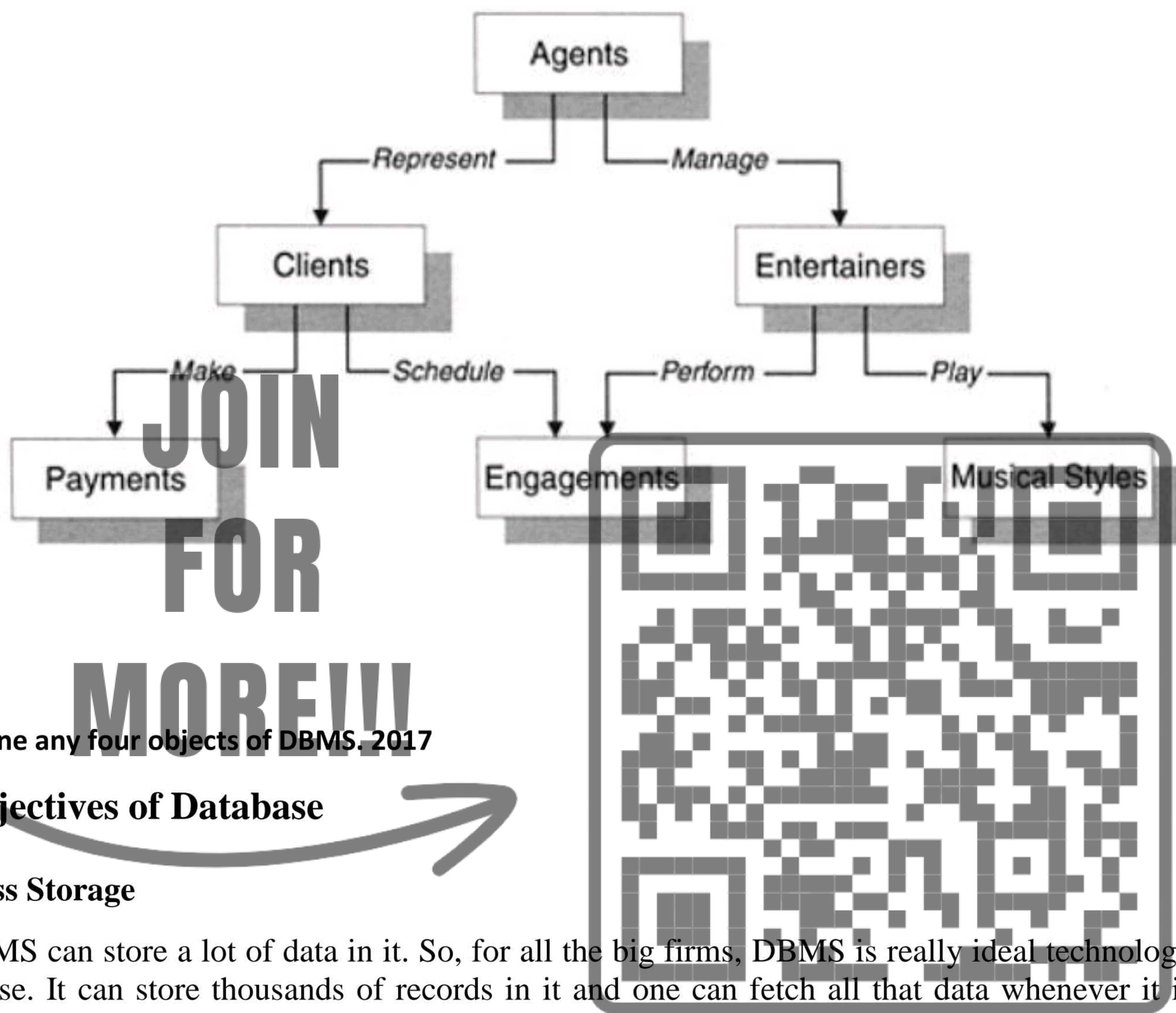
Define network Model. where do we use? 2017

Network model

The network model builds on the hierarchical model by allowing many-to-many relationships between linked records, implying multiple parent records. Based on mathematical set theory, the model is constructed with sets of related records. Each set consists of one owner or parent record and one or more member or child records. A record can be a member or child in multiple sets, allowing this model to convey complex relationships.



Network database model



Define any four objects of DBMS. 2017

Objectives of Database

Mass Storage

DBMS can store a lot of data in it. So, for all the big firms, DBMS is really ideal technology to use. It can store thousands of records in it and one can fetch all that data whenever it is needed.

Removes Duplicity

If you have lots of data, then data duplicity will occur at any instance. DBMS guarantee it that there will be no data duplicity among all the records. While storing new records, DBMS makes sure that same data was not inserted before.

Multiple Users Access

No one handles the whole database alone. There are lots of users who are able to access database. So, this situation may happen that two or more users are accessing database. They can change whatever they want, at that time DBMS makes it sure that they can work concurrently.



Chapter # 15 past papers with answers 2nd year

Data Protection

Information such as bank details, employee's salary details and sale purchase details should always be kept secured. Also, all the companies need their data secured from unauthorized use. DBMS gives a master level security to their data. No one can alter or modify the information without the privilege of using that data.

Data Backup and recovery

Sometimes database failure occurs so there is no option like one can say that all the data has been lost. There should be a backup of database so that on database failure it can be recovered. DBMS has the ability to backup and recover all the data in database.

Everyone can work on DBMS

There is no need to be a master of programming language if you want to work on DBMS. Any accountant who is having less technical knowledge can work on DBMS. All the definitions and descriptions are given in it so that even a non-technical background person can work on it.

Integrity

Integrity means your data is authentic and consistent. DBMS has various validity checks that make your data completely accurate and consistent.

Platform Independent

One can run dbms at any platform. No particular platform is required to work on database management system.

What are the responsibilities of DBA? 2016

DBA Responsibilities

- Maintaining all databases required for development, testing, training and production usage
- Maximizing uptime of databases
- Managing share resources used amongst applications
- Administrates all database objects, including tables, views, indexes, stored procedures, functions, packages, sequences and clusters
- Enforces and maintains database constraints to ensure integrity of the database
- Installation and configuration of DBMS server software and related products.
- Upgrading and patching/hot-fixing of DBMS server software and related products.
- Assists with impact analysis of any changes made to the database objects.
- Migrate database to another server.
- Evaluate DBMS features and DBMS related products.
- Ensure that the site is running the products that are most appropriate
- ensure that any new product usage or release upgrade takes place with minimal impact
- Establish and maintain sound backup and recovery policies and procedures.
- Take care of the Database design and implementation.
- Implement and maintain database security (create and maintain logins, users and roles, assign privileges).
- Performance tuning and health monitoring on DBMS, OS and application.
- Setting Up Server Level, Database Level and Operating System Alerts
- Implementation of robust maintenance plan (Index Defrag, Stats Update, DBCC, Reorg etc)



Chapter # 15 past papers with answers 2nd year

What is database? What are the advantages of relational database management system (RDBMS)?

A **Database** is a collection of related data organised in a way that data can be easily accessed, managed and updated. A database is one of the important components for many applications and is used for storing a series of data in a single set. In other words, it is a group/package of information that is put in order so that it can be easily accessed, managed and updated. In a database, even the smallest portion of information becomes the data. Example, Student is a data, roll number is a data, and the address is a data, height, weight, marks everything is data. In brief, all the living and non-living objects in this world is a data.

Features of RDBMS Systems:

- All data stored in the tables are provided by an RDBMS
- Ensures that all data stored are in the form of rows and columns
- Facilitates primary key, which helps in unique identification of the rows
- Index creation for retrieving data at a higher speed
- Facilitates a common column to be shared amid two or more tables
- Multi-user accessibility is facilitated to be controlled by individual users
- A virtual table creation is enabled to store sensitive data and simplify queries

These features are the reasons why RDBMS is widely used across the world. In simple words, RDBMS is the software system that allows storing data in the form of tables. As we discussed, data stored within RDBMS is in the form of rows and columns and this kind of a setup is often referred to as tuples and attributes.

List some advantages of DBMS? 2015

Advantages of Database Management System (DBMS)

1. Improved data sharing

An advantage of the database management approach is, the DBMS helps to create an environment in which end users have better access to more and better-managed data.

Such access makes it possible for end users to respond quickly to changes in their environment.

2. Improved data security

Chapter # 15 past papers with answers2nd year

The more users access the data, the greater the risks of data security breaches. Corporations invest considerable amounts of time, effort, and money to ensure that corporate data are used properly. A DBMS provides a framework for better enforcement of data privacy and security policies.

3. *Better data integration*

Wider access to well-managed data promotes an integrated view of the organization's operations and a clearer view of the big picture. It becomes much easier to see how actions in one segment of the company affect other segments.

4. *Minimized data inconsistency*

Data inconsistency exists when different versions of the same data appear in different places. For example, data inconsistency exists when a company's sales department stores a sales representative's name as "Bill Brown" and the company's personnel department stores that same person's name as "William G. Brown," or when the company's regional sales office shows the price of a product as \$45.95 and its national sales office shows the same product's price as \$43.95. The probability of data inconsistency is greatly reduced in a properly designed database.

5. *Improved data access*

The DBMS makes it possible to produce quick answers to ad hoc queries. From a database perspective, a query is a specific request issued to the DBMS for data manipulation—for example, to read or update the data. Simply put, a query is a question, and an ad hoc query is a spur-of-the-moment question. The DBMS sends back an answer (called the query result set) to the application. For example, end users, when dealing with large amounts of sales data, might want quick answers to questions (ad hoc queries) such as:

- What was the dollar volume of sales by product during the past six months?
- What is the sales bonus figure for each of our salespeople during the past three months?
- How many of our customers have credit balances of 3,000 or more?

6. *Improved decision making*

Chapter # 15 past papers with answers 2nd year

Better-managed data and improved data access make it possible to generate better-quality information, on which better decisions are based. The quality of the information generated depends on the quality of the underlying data. Data quality is a comprehensive approach to promoting the accuracy, validity, and timeliness of the data. While the DBMS does not guarantee data quality, it provides a framework to facilitate data quality initiatives.

7. Increased end-user productivity

The availability of data, combined with the tools that transform data into usable information, empowers end users to make quick, informed decisions that can make the difference between success and failure in the global economy.

Write full form of DML, DDL, SQL, IDE, RDBMS, DBMS. 2015

DML: Data manipulation language

DDL: Data definition language

SQL: Structured query language

IDE: Integrated development environment

RDBMS: Relational database management system

DBMS: Database management system

What is a key field? Why is it important in DBMS? 2015

Key Field:

A **field** in a record that holds unique data which identifies that record from all the other records in the file or **database**. Account number, product code and customer name are typical **key fields**. As an identifier, each **key** value must be unique in each record. **Keys** are a major component of table-level integrity and relationship-level integrity. For instance, they enable you to ensure that a table has unique records and that the fields you use to establish a relationship between a pair of tables always contain **matching** values. They serve to establish table relationships.

Explain various types of database Model?

Elucidate the term database model. Explain relational and database model. 2011

Define relational database model. 2018



Chapter # 15 past papers with answers2nd year

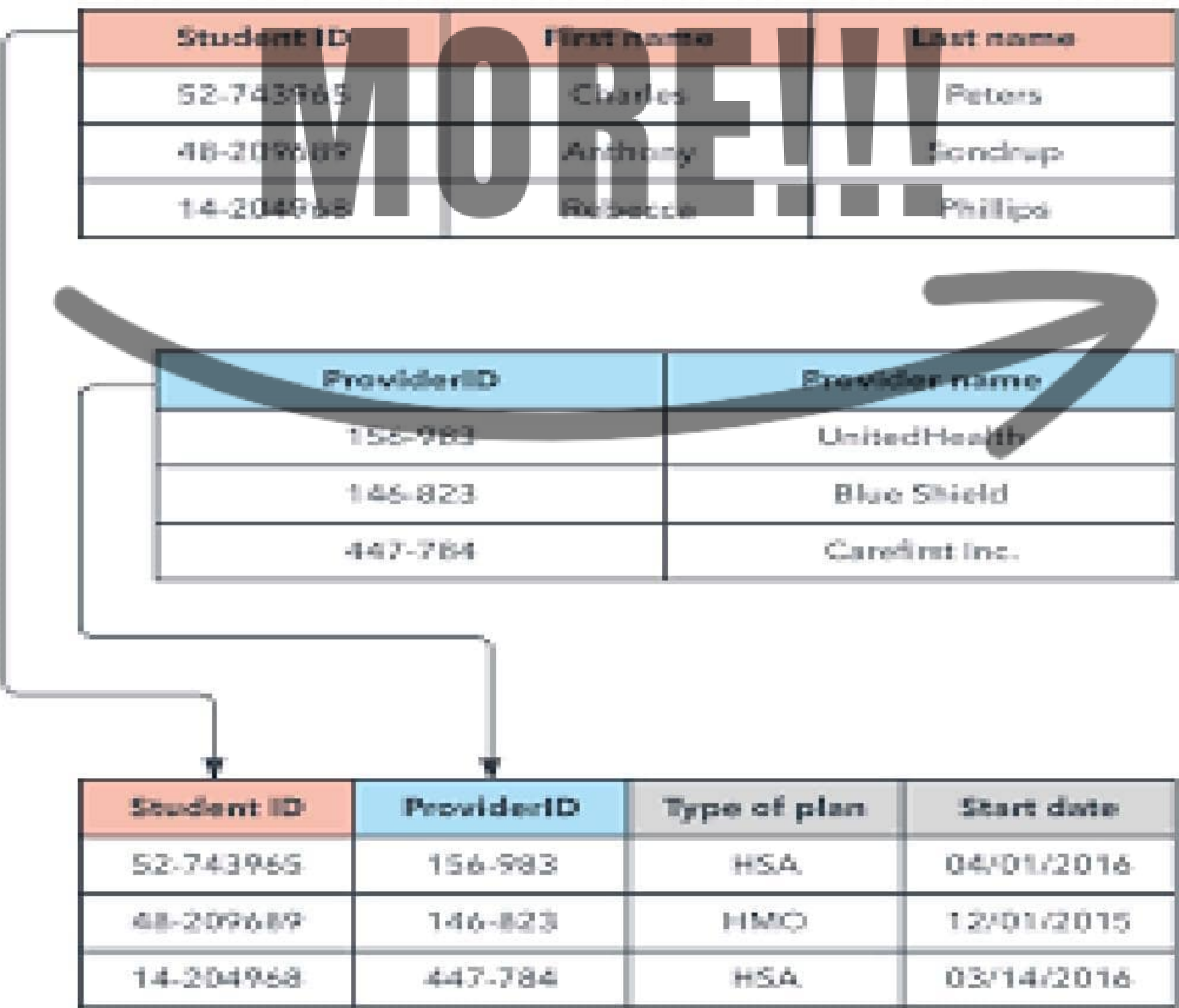
Types of database models

There are many kinds of data models. Some of the most common ones include:

- Hierarchical database model
- Relational model
- Network model

Relational model

The most common model, the relational model sorts data into tables, also known as relations, each of which consists of columns and rows. Each column lists an attribute of the entity in question, such as price, zip code, or birth date. Together, the attributes in a relation are called a domain. A particular attribute or combination of attributes is chosen as a primary key that can be referred to in other tables, when it's called a foreign key. Each row, also called a tuple, includes data about a specific instance of the entity in question, such as a particular employee. The model also accounts for the types of relationships between those tables, including one-to-one, one-to-many, and many-to-many relationships. Here's an example:



Hierarchical model

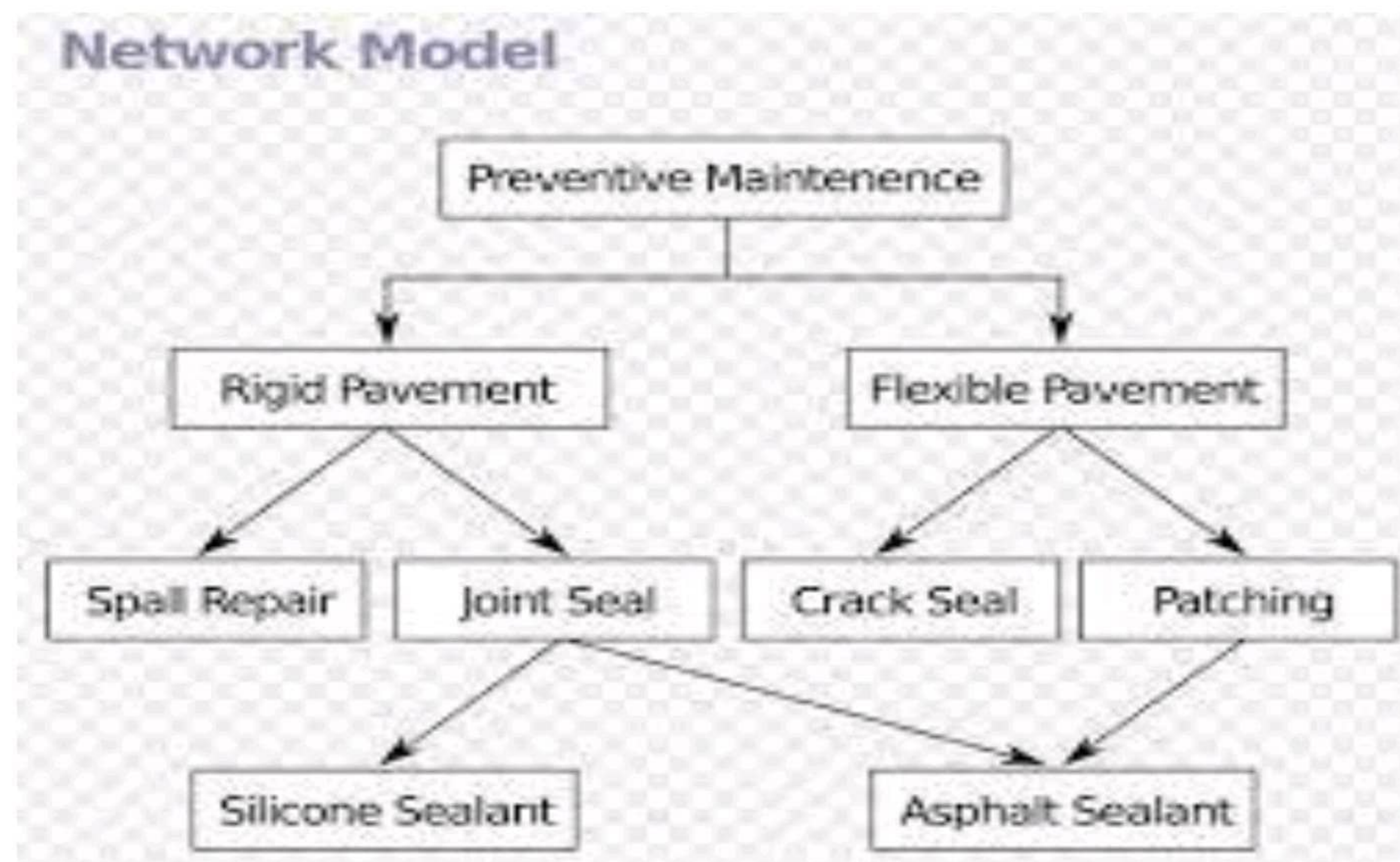
The hierarchical model organizes data into a tree-like structure, where each record has a single parent or root. Sibling records are sorted in a particular order. That order is used as the physical order for storing the database. This model is good for describing many real-world relationships.



Hierarc

Reconstru

Routin



Network model

The network model builds on the hierarchical model by allowing many-to-many relationships between linked records, implying multiple parent records. Based on mathematical set theory, the model is constructed with sets of related records. Each set consists of one owner or parent record and one or more member or child records. A record can be a member or child in multiple sets, allowing this model to convey complex relationships.

**JOIN
FOR
MORE!!!**

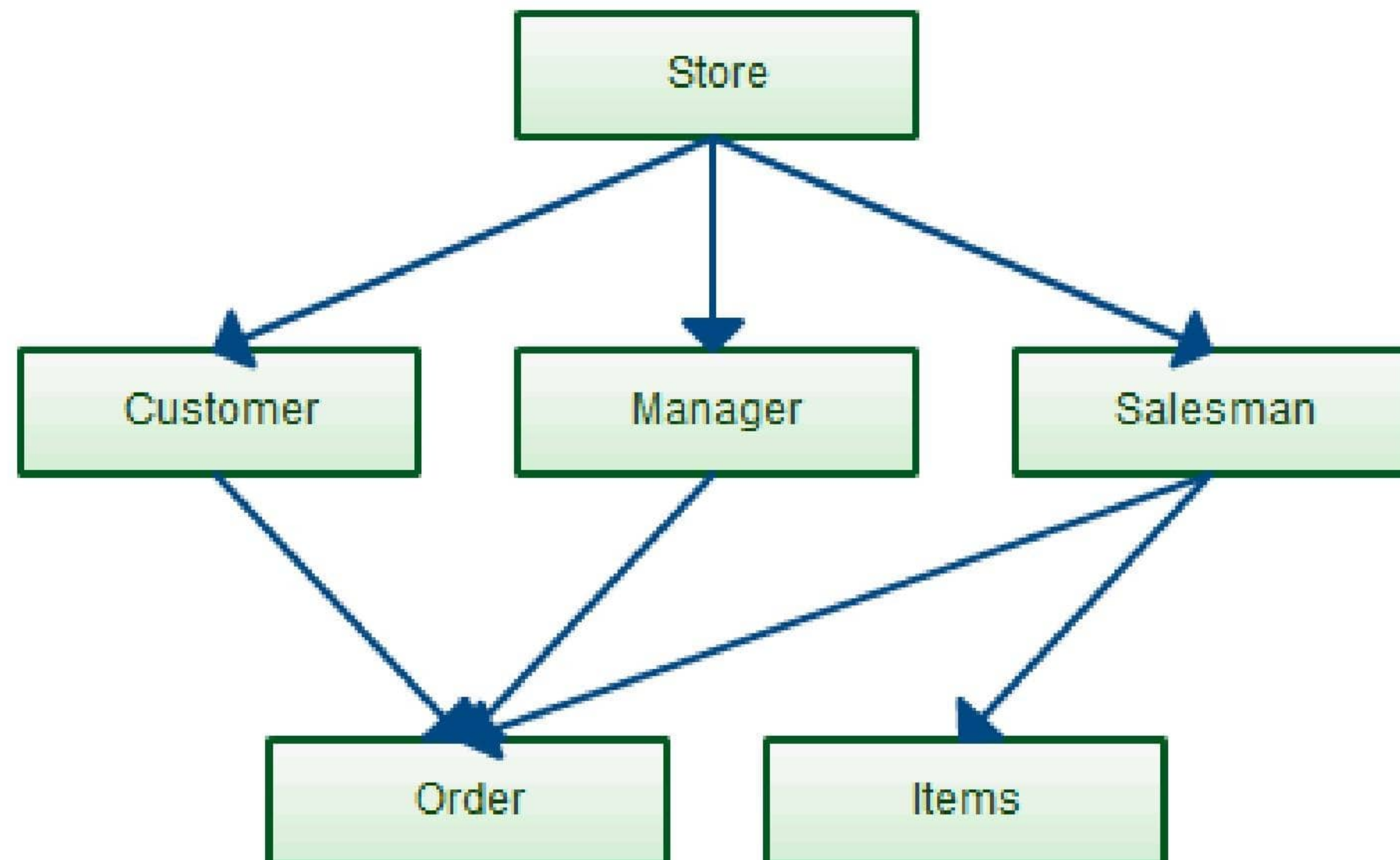


Define database Model with diagram. 2013

A **database model** is a type of **data model** that determines the logical structure of a **database** and fundamentally determines in which manner **data** can be stored, organized and manipulated. The most popular example of a **database model** is the relational **model**, which uses a table-based format.



Chapter # 15 past papers with answers2nd year



Elaborate the term **primary key** with its characteristics. 2011

What is a primary key?

In the [world of databases](#), the primary key of a relational table uniquely identifies each record in the table. Databases use keys to compare, sort, and store records, and to create relationships between records. A **primary key** is a minimal set of attributes (columns) in a table that uniquely identifies tuples (rows) in that table.

- Choosing the primary key in a database is one of the most important steps in the process.
- The value of primary key should be unique for each row of the table. The column(s) that makes the key cannot contain duplicate values.
- The attribute(s) that is marked as primary key is not allowed to have null values.
- Primary keys are not necessarily to be a single attribute (column). It can be a set of more than one attributes (columns).

Primary Keys



<u>StudentId</u>	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042
L0023487	Peter	Murray	P301
L0018453	Anne	Norris	S042

What is data dictionary in DBMS. 2017



Data Dictionary

This is a reserved space within a database used to store information about the database itself. A data dictionary is a set of read-only table and views, containing the different information about the data used in the enterprise to ensure that database representation of the data follow one standard as defined in the dictionary.

Explain how primary and foreign key are used in database management system? 2009

A **primary key** is **used** to ensure data in the specific column is unique. You can only set constraints with **primary keys**, by setting a **foreign key** to another column which creates a relationship with the column that has the **primary key** set. A prime use of a **primary key** is in the case of a user's table.

**JOIN
FOR
MORE!!!**



Chapter# 16 2nd Year

Brief Introduction of MS Access:

Microsoft Access is a computer application used to create and manage computer-based databases on desktop computers and/or on connected computers (a network). Microsoft Access can be used for personal information management (PIM), in a small business to organize and manage data, or in an enterprise to communicate with servers.

Microsoft Access stores information which is called a database. To use MS Access, you will need to follow these four steps –

- **Database Creation** – Create your Microsoft Access database and specify what kind of data you will be storing.
- **Data Input** – After your database is created, the data of every business day can be entered into the Access database.
- **Query** – This is a fancy term to basically describe the process of retrieving information from the database.
- **Report** (optional) – Information from the database is organized in a nice presentation that can be printed in an Access Report.

Data Types:

MS Access supports different types of data, each with a specific purpose.

- The data type determines the kind of the values that users can store in any given field. • Each field can store data consisting of only a single data type.

Here are some of the most common data types you will find used in a typical Microsoft Access database.

Type of Data	Description	Size
Short Text	Text or combinations of text and numbers, including numbers that do not require calculating (e.g. phone numbers).	Up to 255 characters.
Long Text	Lengthy text or combinations of text and numbers.	Up to 63, 999 characters.
Number	Numeric data used in mathematical calculations.	1, 2, 4, or 8 bytes (16 bytes if set to Replication ID).

Date/Time	Date and time values for the years 100 through 9999.	8 bytes
------------------	--	---------

PREPARED BY SANA MAHMOOD **1**

Chapter# 16 2nd Year

Currency	Currency values and numeric data used in mathematical calculations involving data with one to four decimal places.	8 bytes
AutoNumber	A unique sequential (incremented by 1) number or random number assigned by Microsoft Access whenever a new record is added to a table.	4 bytes (16 bytes if set to Replication ID).
Yes/No	Yes and No values and fields that contain only one of two values (Yes/No, True/False, or On/Off).	1 bit.

Database Objects:

MS Access uses “objects” to help the user list and organize information, as well as prepare specially designed reports. When you create a database, Access offers you Tables, Queries, Forms, Reports, Macros, and Modules. Databases in Access are composed of many objects but the following are the major objects –

- Tables
- Queries
- Forms
- Reports

Together, these objects allow you to enter, store, analyze, and compile your data. Here is a summary of the major objects in an Access database;

Table

Table is an object that is used to define and store data. When you create a new table, Access asks you to define fields which is also known as column headings.

- Each field must have a unique name, and data type.
- Tables contain fields or columns that store different kinds of data, such as a name or an address, and records or rows that collect all the information about a particular instance of the subject, such as all the information about a customer or employee etc.



- You can define a primary key, one or more fields that have a unique value for each record, and one or more indexes on each table to help retrieve your data more quickly.

Query

PREPARED BY SANA MAHMOOD 2

Chapter# 16 2nd Year

An object that provides a custom view of data from one or more tables. Queries are a way of searching for and compiling data from one or more tables.

- Running a query is like asking a detailed question of your database.
- When you build a query in Access, you are defining specific search conditions to find exactly the data you want.

Form

Form is an object in a desktop database designed primarily for data input or display or for control of application execution. You use forms to customize the presentation of data that your application extracts from queries or tables.

- Forms are used for entering, modifying, and viewing records.
- The reason forms are used so often is that they are an easy way to guide people toward entering data correctly.
- When you enter information into a form in Access, the data goes exactly where the database designer wants it to go in one or more related tables.



Report

Report is an object in desktop databases designed for formatting, calculating, printing, and summarizing selected data.

- You can view a report on your screen before you print it.
- If forms are for input purposes, then reports are for output.
- Anything you plan to print deserves a report, whether it is a list of names and addresses, a financial summary for a period, or a set of mailing labels.
- Reports are useful because they allow you to present components of your database in an easy-to-read format.



Creating Tables:

When you create a database, you store your data in tables. Because other database objects depend so heavily on tables, you should always start your design of a database by creating all of its tables and then creating any other object. Before you create tables, carefully consider

your requirements and determine all the tables that you need.

Let us try and create the first table that will store the basic contact information concerning the employees as shown in the following table –

Field Name	Data Type
EmployeeID	AutoNumber
FirstName	Short Text
LastName	Short Text
Address1	Short Text
Address2	Short Text
City	Short Text
State	Short Text
Zip	Short Text
Phone	Short Text
Phone Type	Short Text



Advance Ms Access

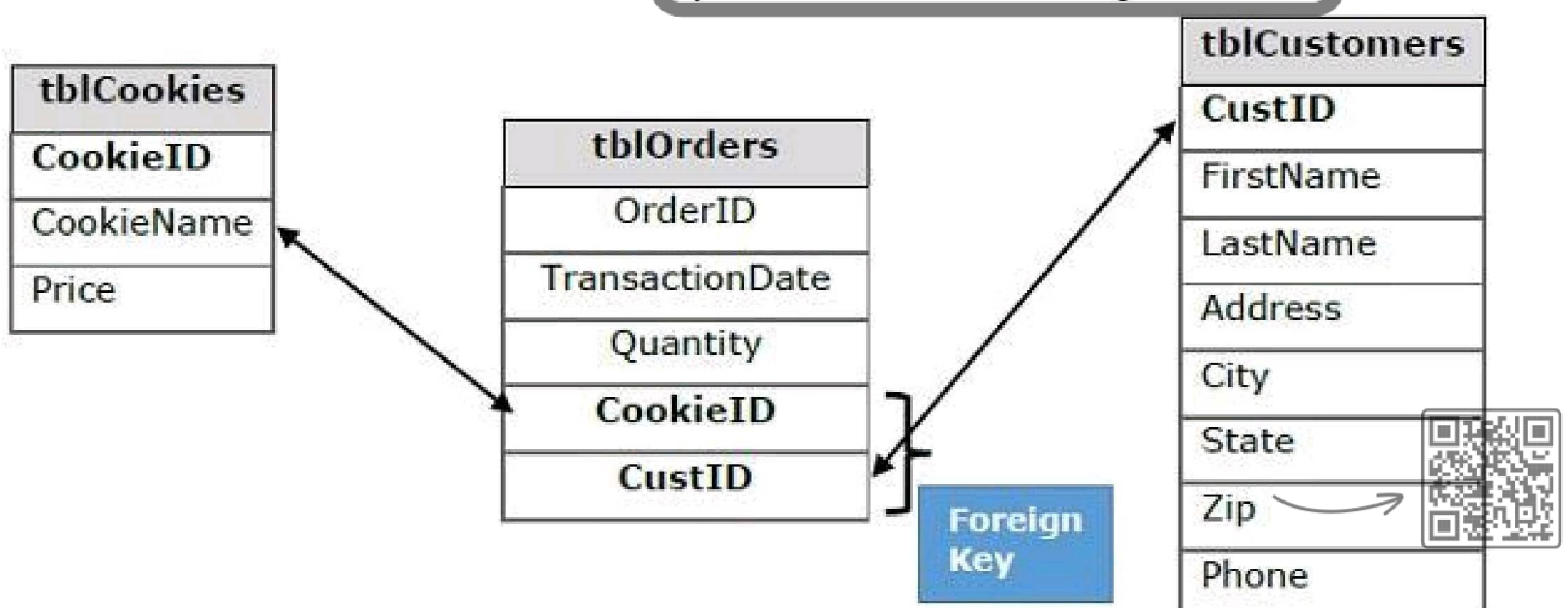
Normalization

Database normalization, or simply normalization, is the process of organizing columns (attributes) and tables (relations) of a relational database to minimize data redundancy. It is the process of splitting data across multiple tables to improve overall performance, integrity and longevity.

- Normalization is the process of organizing data in a database.
- This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.
- Let us now look into the following table which contains data, but the problem is that this data is quite redundant which increases the chances of typo and inconsistent phrasing during data entry.

CustID	Name	Address	Cookie	Quantity	Price	Total
1	Ethel Smith	12 Main St, Arlington, VA 22201 S	Chocolate Chip	5	\$2.00	\$10.00
2	Tom Wilber	1234 Oak Dr., Pekin, IL 61555	Choc Chip	3	\$2.00	\$6.00
3	Ethil Smithy	12 Main St., Arlington, VA 22201	Chocolate Chip	5	\$2.00	\$10.00

- To solve this problem, we need to restructure our data and break it down into multiple tables to eliminate some of those redundancy as shown in the following three tables.



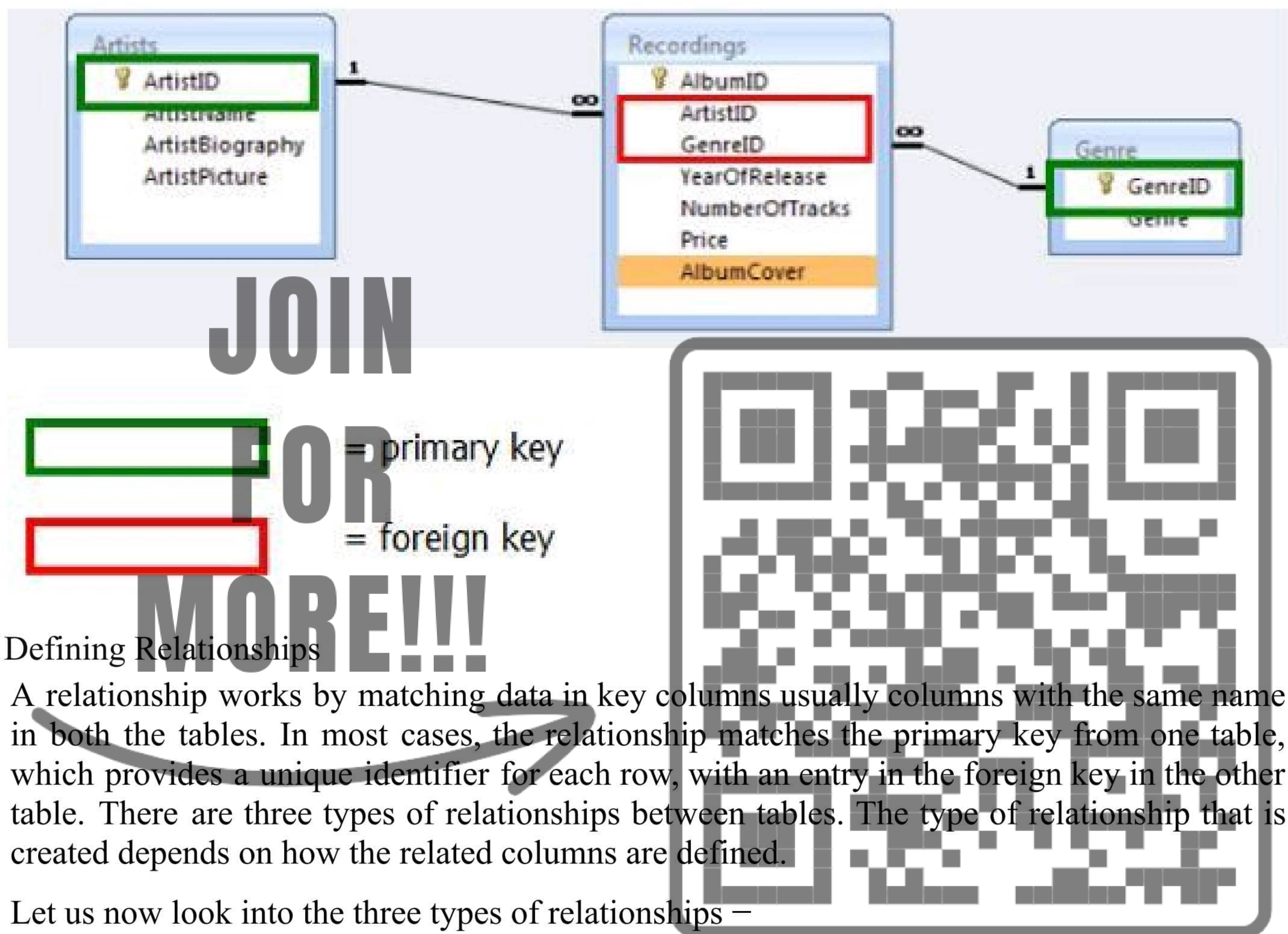
- Here, we have one table for Customers, the 2nd one is for Orders and the 3rd one is for Cookies.

Chapter# 172nd year

- The problem here is that just by splitting the data in multiple tables will not help to tell how data from one table relates to data in another table. To connect data in multiple tables, we have to add foreign keys to the **Orders** table.

Foreign key:

In the context of relational databases, a foreign key is a field in one table that uniquely identifies a row of another table or the same table. In simpler words, the foreign key is defined in a second table, but it refers to the primary key or a unique key in the first table.



One-to-Many Relationships

A one-to-many relationship is the most common type of relationship. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A.

For example, the Customers and Orders tables have a one-to-many relationship: each customer can place many orders, but each order comes from only one customer.

Many-to-Many Relationships

In a many-to-many relationship, a row in table A can have many matching rows in table B, and vice versa.

You create such a relationship by defining a third table, called a **junction table**, whose primary key consists of the foreign keys from both table A and table B.

For example, the Customers table and the Cookies table have a many-to-many relationship that is defined by a one-to-many relationship from each of these tables to the Orders table.

One-to-One Relationships

In a one-to-one relationship, a row in table A can have no more than one matching row in table B, and vice versa. A one-to-one relationship is created if both the related columns are primary keys or have unique constraints.

**JOIN
FOR
MORE!!!**



Chapter# 17 Past Paper Question with Answers 2nd year

What is relationship? State kinds of relationships used in MS Access. (2012)

What are the steps involved in creating reports in MS Access? (2012) What are the main objects in MS Access? (2011-12)

What is foreign key? Where do we use it? (2017)

In which condition is one to many relationships used? (2017)

Define database. Define major data types used in MS Access. (2017)

What is relationship? State kinds of relationships used in MS Access. (2012)

Defining Relationships

A relationship works by matching data in key columns usually columns with the same name in both the tables. In most cases, the relationship matches the primary key from one table, which provides a unique identifier for each row, with an entry in the foreign key in the other table. There are three types of relationships between tables. The type of relationship that is created depends on how the related columns are defined.

Let us now look into the three types of relationships –

One-to-Many Relationships

A one-to-many relationship is the most common type of relationship. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A.

For example, the Customers and Orders tables have a one-to-many relationship: each customer can place many orders, but each order comes from only one customer.

Many-to-Many Relationships

In a many-to-many relationship, a row in table A can have many matching rows in table B, and vice versa.

You create such a relationship by defining a third table, called a **junction table**, whose primary key consists of the foreign keys from both table A and table B.

For example, the Customers table and the Cookies table have a many-to-many relationship that is defined by a one-to-many relationship from each of these tables to the Orders table.

One-to-One Relationships

In a one-to-one relationship, a row in table A can have no more than one matching row in table B, and vice versa. A one-to-one relationship is created if both the related columns are primary keys or have unique constraints.



Chapter# 17 Past Paper Question with Answers 2nd year

What are the steps involved in creating reports in MS Access? (2012)

1. On the **Create** tab in the **Reports** group, click **Report Wizard**. The wizard starts.

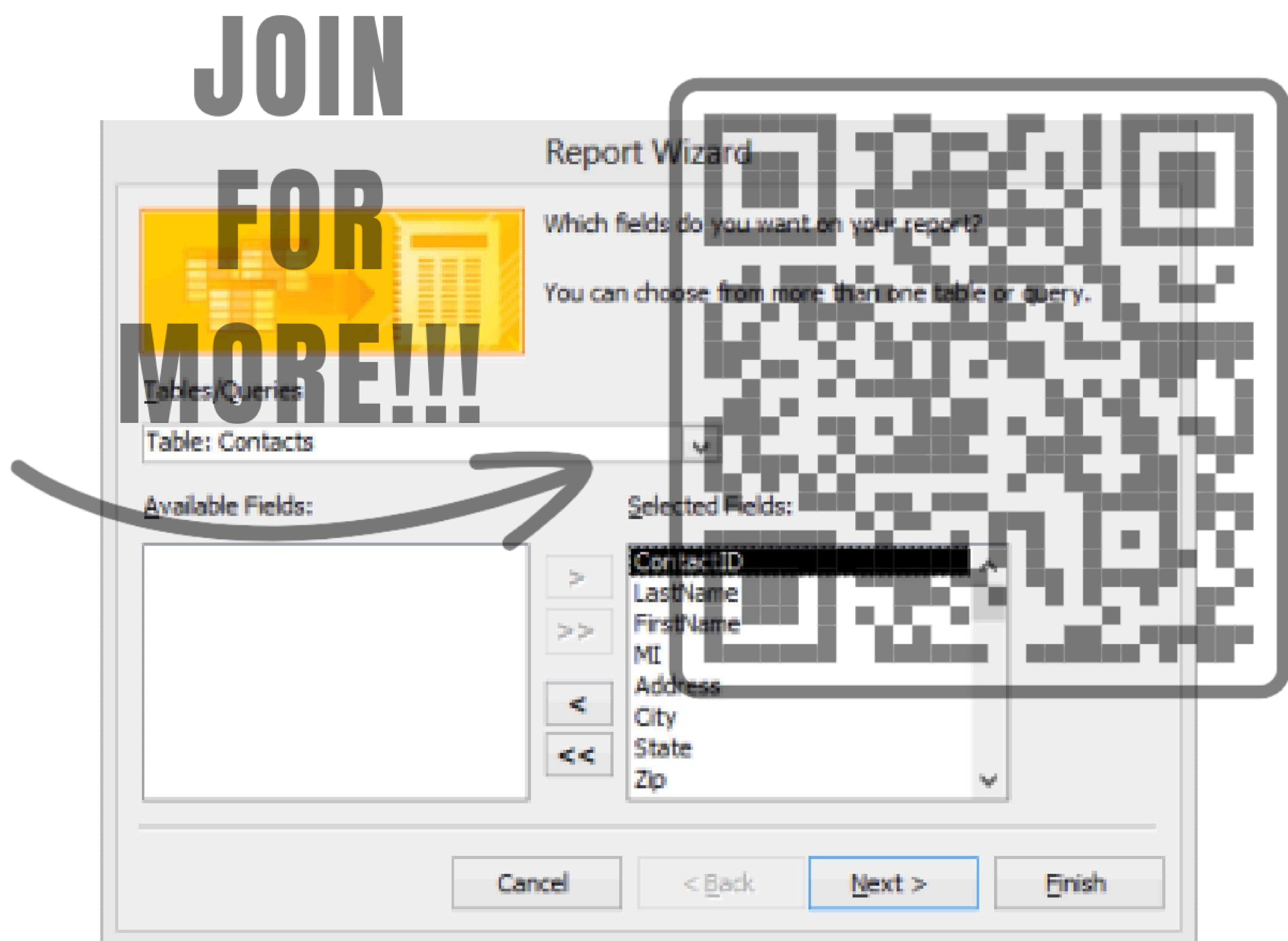


2. From the **Tables/Queries** drop-down list, select the table (or query) to base the report on. The fields for the selected table load in the **Available Fields** list box.

3. Move the fields to include on the report from the **Available Fields** list box to the **Selected Fields** list box. To do so, double-click a field name to move it or highlight the field name and click **>**. To move all fields at once, click **>>**.

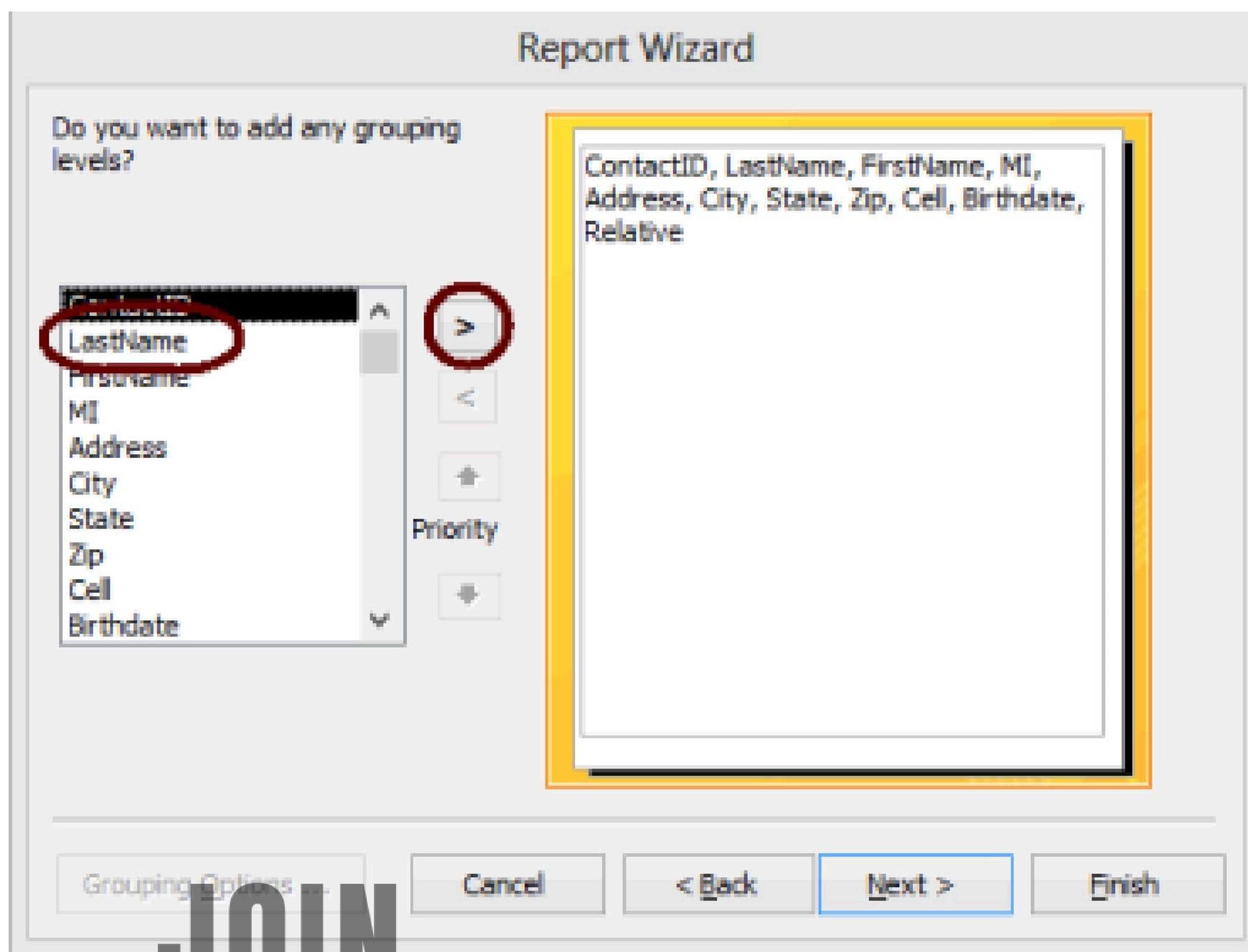


Chapter# 17 Past Paper Question with Answers 2nd year



4. Click **Next >**.

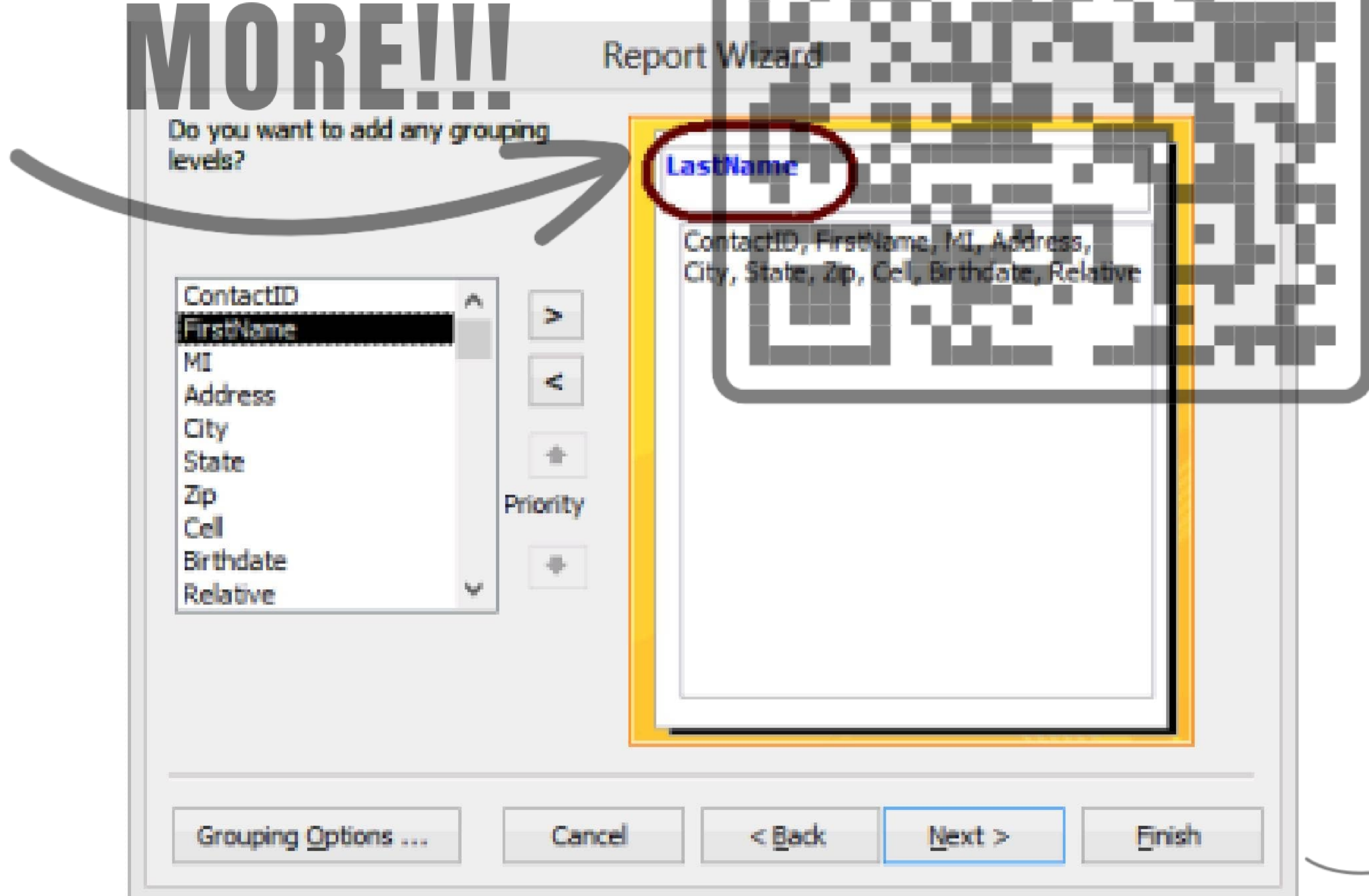




PREPARED
BY SANA MAHMOOD 3

Chapter# 17 Past Paper Question with Answers 2nd year

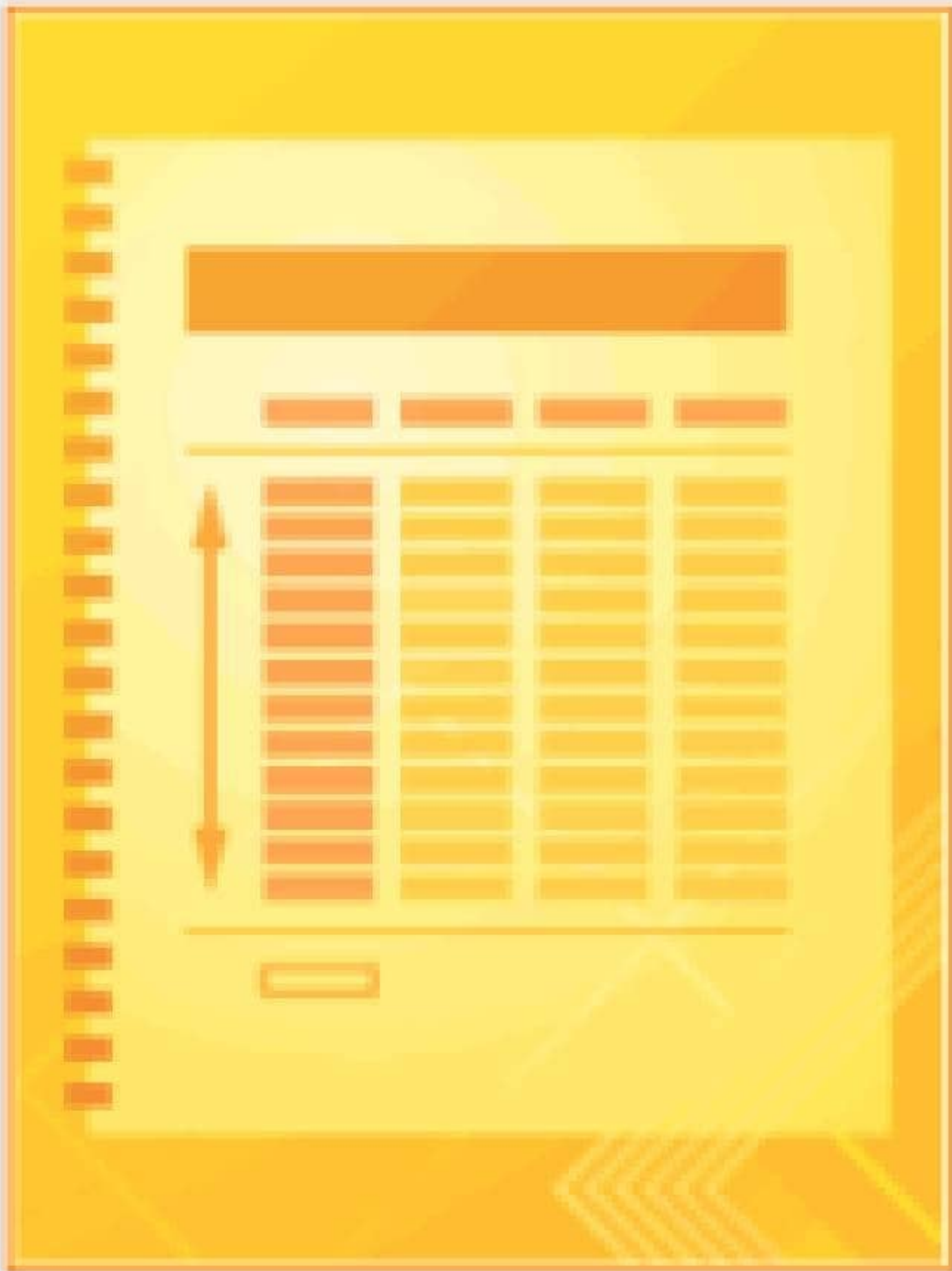
5. To group records on the report by a particular field, highlight the field in the list box and click >.



6. Add more grouping levels if desired. You can use the arrows to change the order of the grouping levels if needed.

Report Wizard

What sort order and summary information do you want for detail records?



You can sort records by up to four fields, in either ascending or descending order.

1		v	Ascending
2		v	Ascending
3		v	Ascending
4		v	Ascending

Summary Options ...

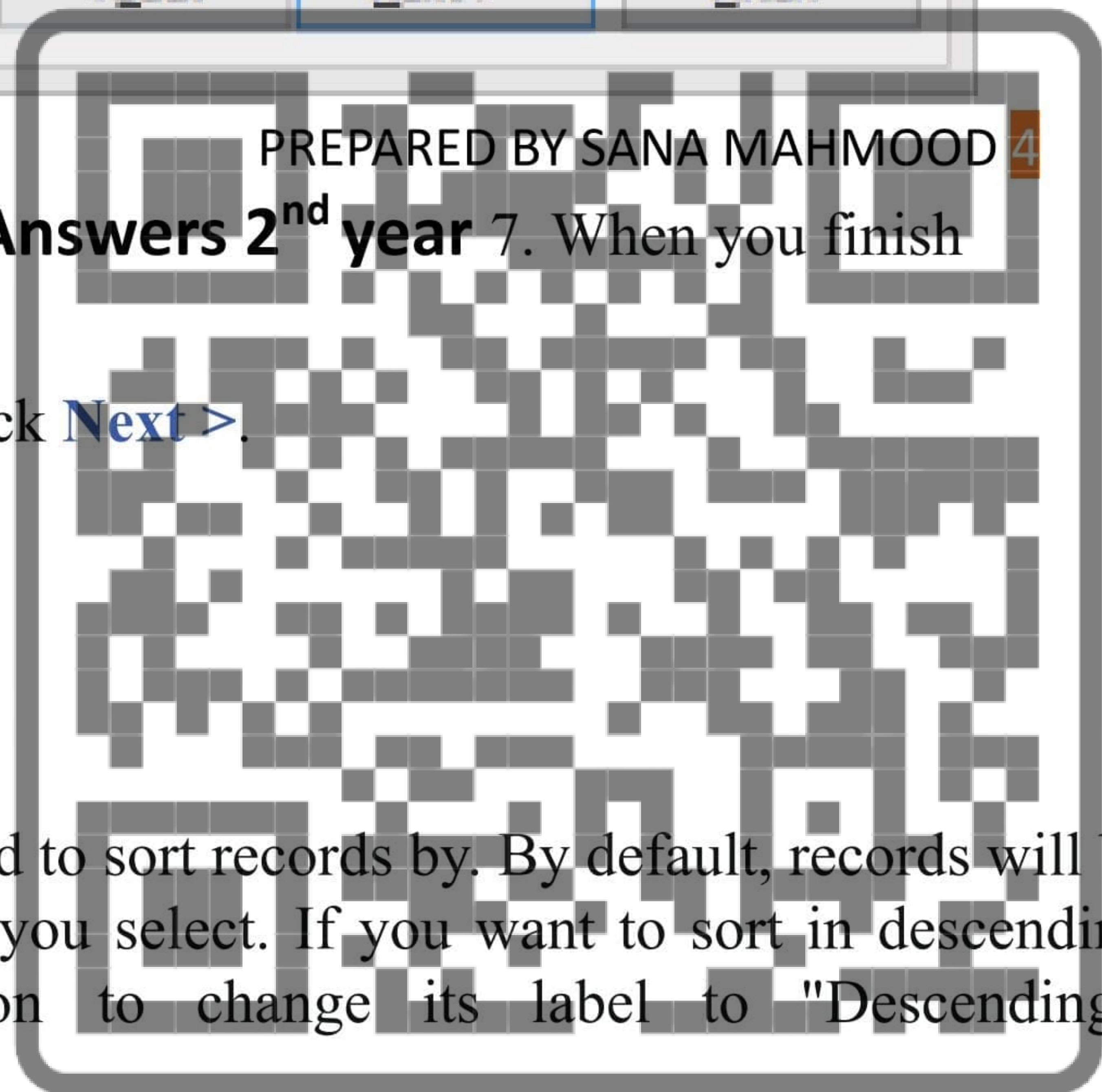
JOIN

Cancel
< Back
Next >
Finish

Chapter# 17 Past Paper Question with Answers 2nd year 7. When you finish

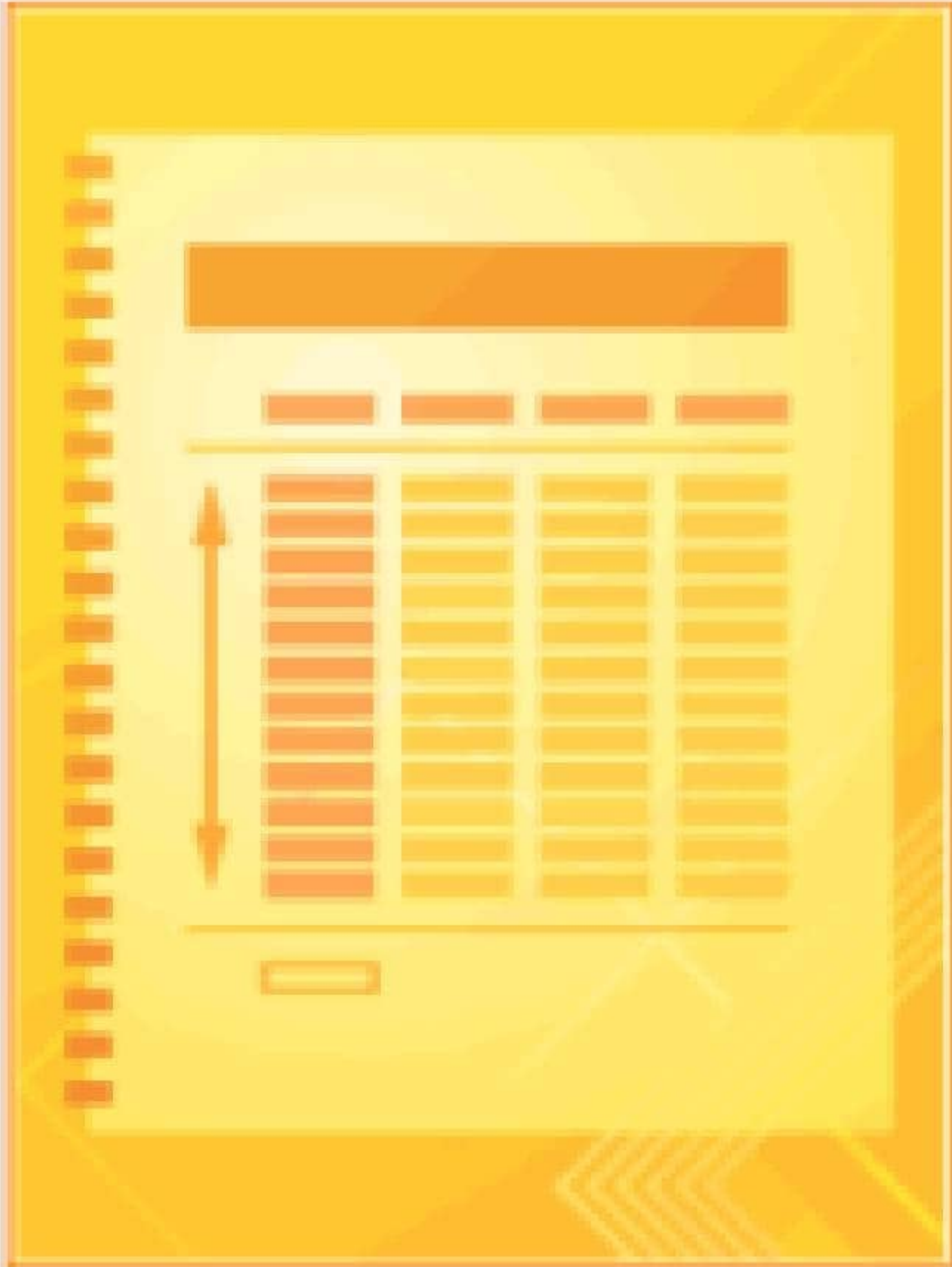
defining how you want records grouped, click [Next >](#).

8. In the first drop-down list, select the field to sort records by. By default, records will be sorted in ascending order by the field you select. If you want to sort in descending order, click the **Ascending** button to change its label to "Descending".



Report Wizard

What sort order and summary information do you want for detail records?



You can sort records by up to four fields, in either ascending or descending order.

1	FirstName	v	Ascending
2		v	Ascending
3		v	Ascending
4		v	Ascending

[Summary Options ...](#)

Cancel
< Back
Next >
Finish

JOIN
FOR
MORE!!!

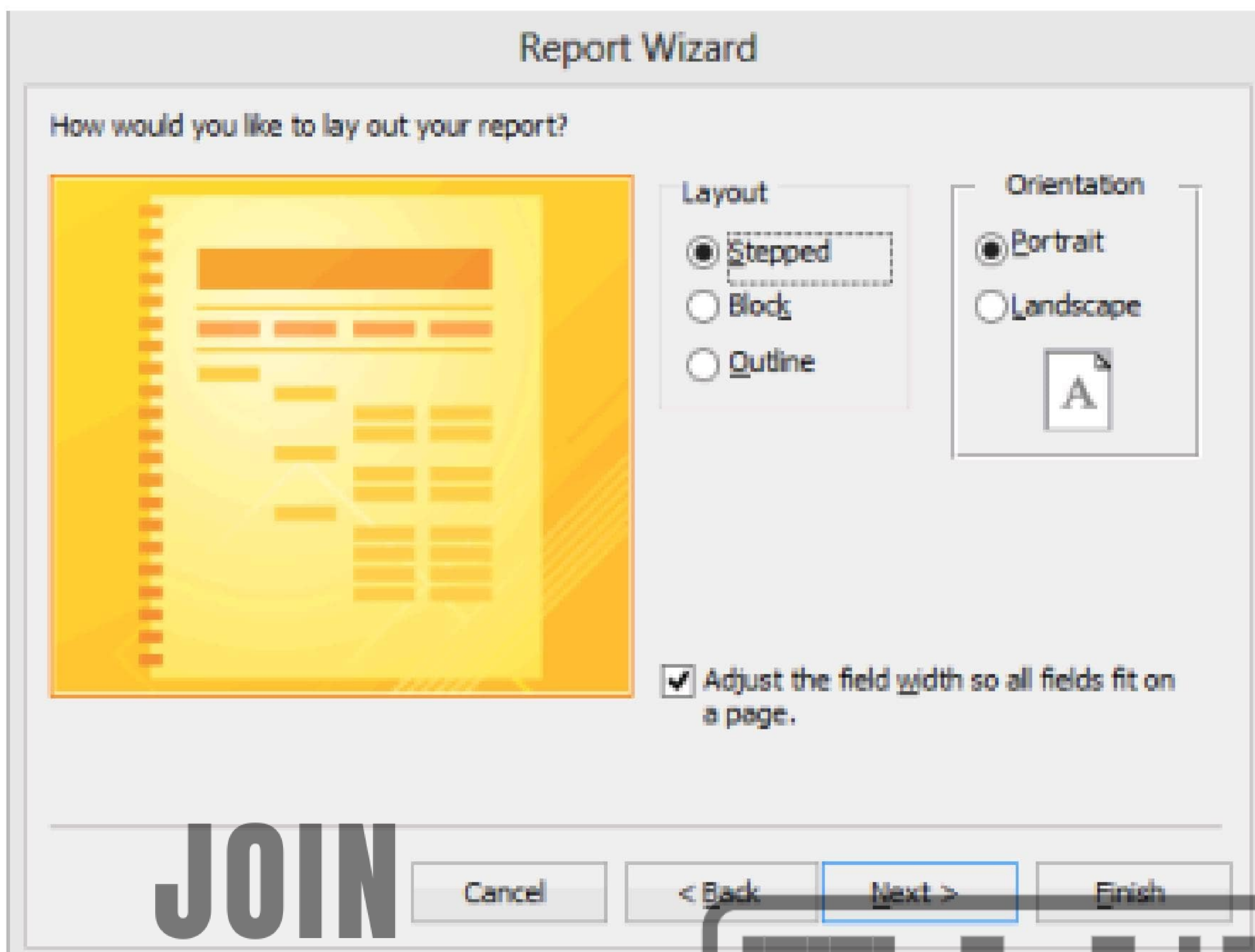


PREPARED BY SANA MAHMOOD 5

Chapter# 17 Past Paper Question with Answers 2nd year

9. You can specify up to four levels of sorting. When you finish specifying sorting options, click **Next >**.



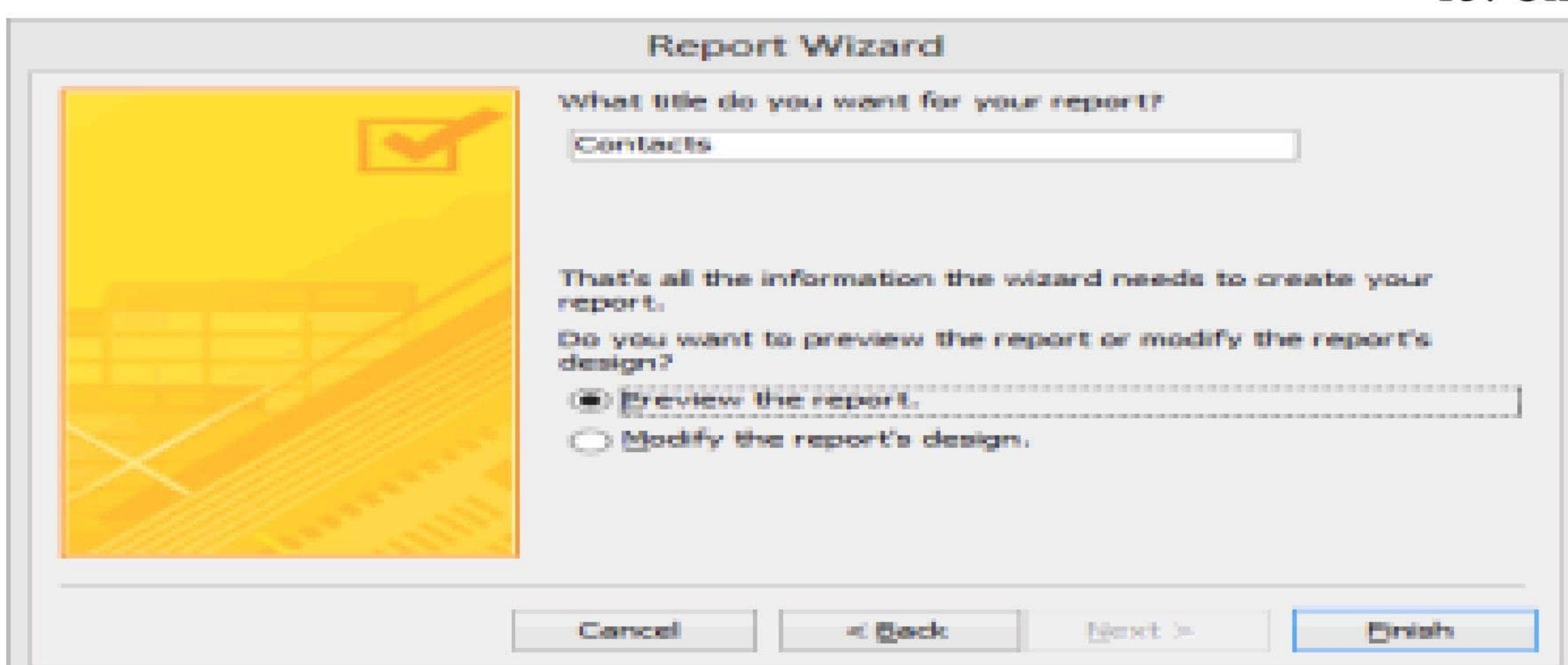


10. In the **Layout** field, select the format of the report. Your options are "Stepped", "Block", and "Outline". (Try the options to see a preview of the report layouts.)

11. In the **Orientation** field, select whether to lay out the report in portrait or landscape mode.

12. If you want all fields to fit on a single page, ensure the **Adjust the field width so all fields fit on a page** check box is marked.

13. Click **Next >**.



PREPARED
BY SANA MAHMOOD 6

Chapter# 17 Past Paper Question with Answers 2nd year



14. Enter a title for the report.

15. Select an option for the view you want to open the report in. Your options are:
- **Preview the report** (opens in Print Preview mode).
 - **Modify the report's design** (opens in Design view).

16. Click **Finish**. The report loads in the view you selected.



What are the main objects in MS Access? (2011-12)

Database Objects:

MS Access uses "objects" to help the user list and organize information, as well as prepare specially designed reports. When you create a database, Access offers you Tables, Queries, Forms, Reports, Macros, and Modules. Databases in Access are composed of many objects but the following are the major objects –

- Tables
- Queries



PREPARED BY SANA MAHMOOD **7**

Chapter# 17 Past Paper Question with Answers 2nd year

- Forms
- Reports

Together, these objects allow you to enter, store, analyze, and compile your data. Here is a summary of the major objects in an Access database;

Table

Table is an object that is used to define and store data. When you create a new table, Access asks you to define fields which is also known as column headings.

- Each field must have a unique name, and data type.
- Tables contain fields or columns that store different kinds of data, such as a name or an address, and records or rows that collect all the information about a particular instance of the subject, such as all the information about a customer or employee etc.
- You can define a primary key, one or more fields that have a unique value for each record, and one or more indexes on each table to help retrieve your data more quickly.

Query

An object that provides a custom view of data from one or more tables. Queries are a way of searching for and compiling data from one or more tables.

- Running a query is like asking a detailed question of your database.
- When you build a query in Access, you are defining specific search conditions to find exactly the data you want.

Form

Form is an object in a desktop database designed primarily for data input or display or for control of application execution. You use forms to customize the presentation of data that your application extracts from queries or tables.

- Forms are used for entering, modifying, and viewing records.
- The reason forms are used so often is that they are an easy way to guide people toward entering data correctly.
- When you enter information into a form in Access, the data goes exactly where the database designer wants it to go in one or more related tables.

Report

Report is an object in desktop databases designed for formatting, calculating, printing, and summarizing selected data.

- You can view a report on your screen before you print it.

JOIN
FOR
MORE!!!



Chapter# 17 Past Paper Question with Answers 2nd year

- If forms are for input purposes, then reports are for output.
- Anything you plan to print deserves a report, whether it is a list of names and addresses, a financial summary for a period, or a set of mailing labels.
- Reports are useful because they allow you to present components of your database in an easy-to-read format.

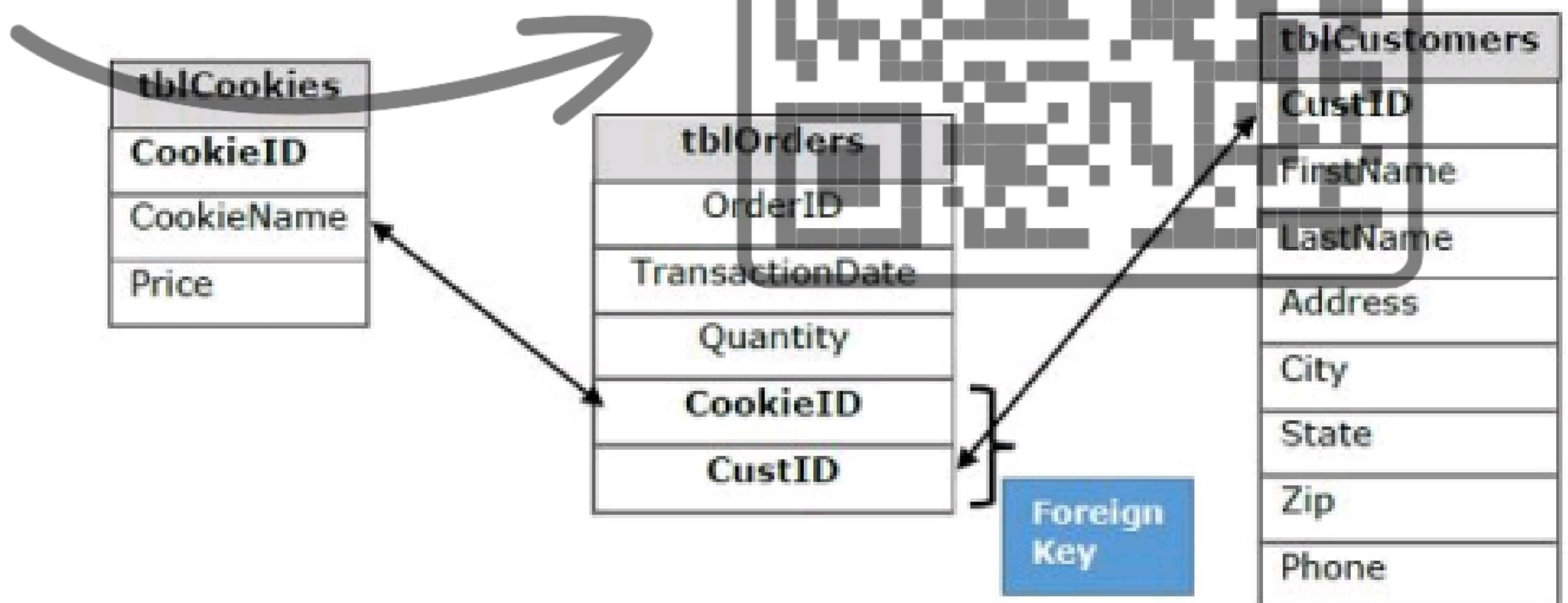
What is foreign key? Where do we use it? (2017)

Foreign key:

In the context of relational databases, a foreign key is a field in one table that uniquely identifies a row of another table or the same table. In simpler words, the foreign key is defined in a second table, but it refers to the primary key or a unique key in the first table.

When data is quite redundant which increases the chances of typo and inconsistent phrasing during data and data is quite redundant which increases the chances of typo and inconsistent phrasing during data entry.

- To solve this problem, we need to restructure our data and break it down into multiple tables to eliminate some of those redundancy as shown in the following three tables.

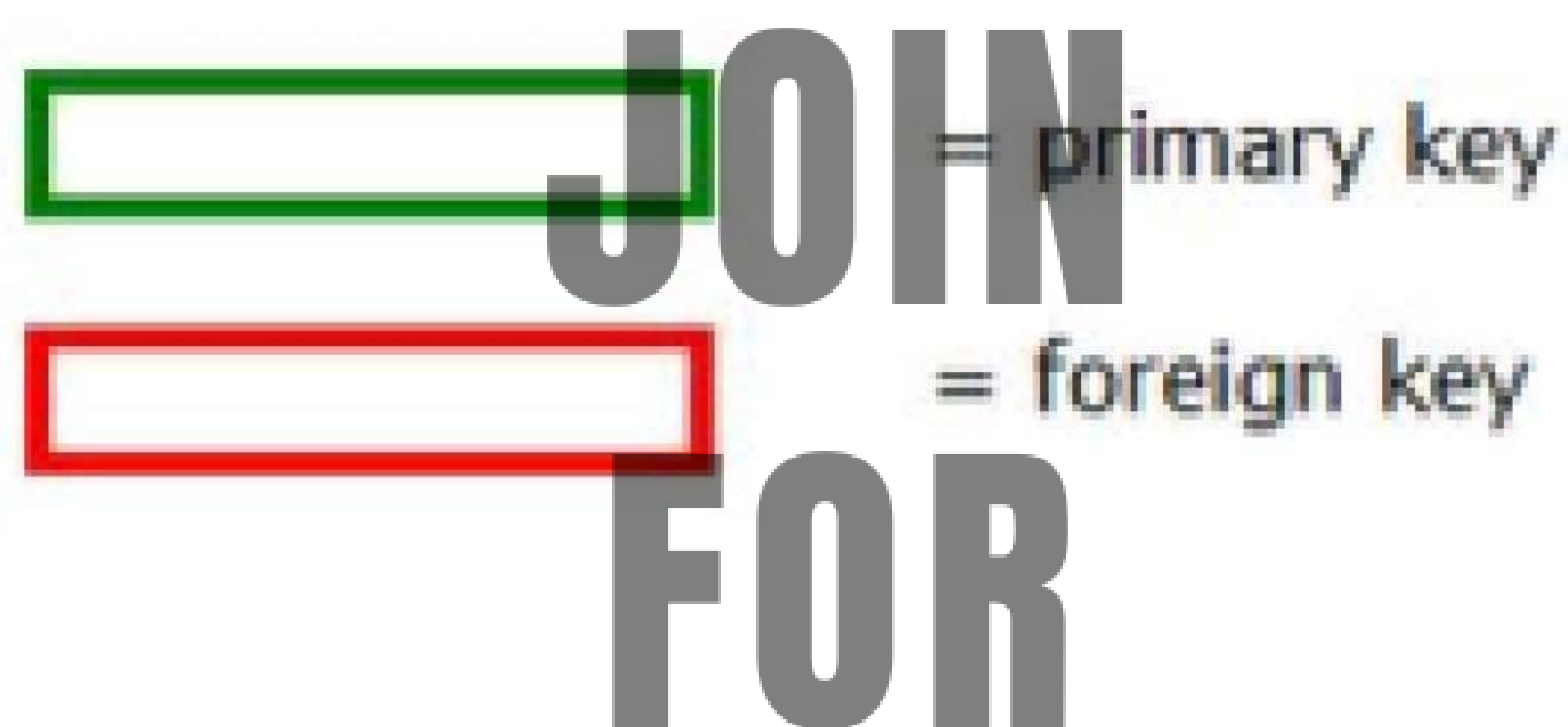
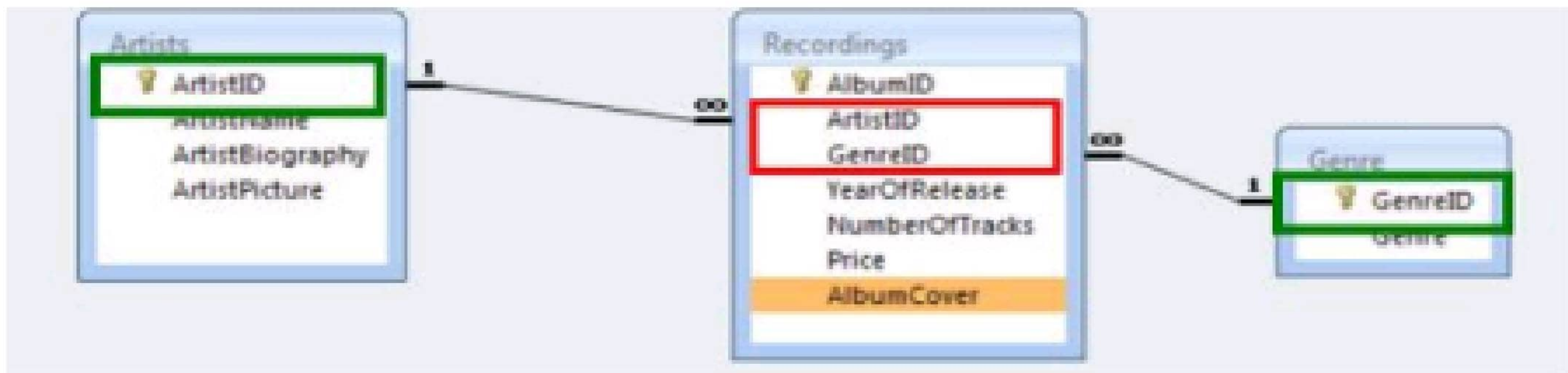


- Here, we have one table for Customers, the 2nd one is for Orders and the 3rd one is for Cookies.

The problem here is that just by splitting the data in multiple tables will not help to tell how data from one table relates to data in another table. To connect data in multiple tables, we have to add foreign keys to the **Orders** table.



Chapter# 17 Past Paper Question with Answers 2nd year



In which condition is one to many relationships used? (2017)

One-to-Many Relationships

A one-to-many relationship is the most common type of relationship. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A.

For example, the Customers and Orders tables have a one-to-many relationship: each customer can place many orders, but each order comes from only one customer.

Define database. Define major data types used in MS Access. (2017)

A **Database** is a collection of related data organised in a way that data can be easily accessed, managed and updated. A database is one of the important components for many applications and is used for storing a series of data in a single set. In other words, it is a group/package of information that is put in order so that it can be easily accessed, managed and updated. In a database, even the smallest portion of information becomes the data. Example, Student is a data, roll number is a data, and the address is a data, height, weight, marks everything is data. In brief, all the living and non-living objects in this world is a data.

Data Types:

MS Access supports different types of data, each with a specific purpose.

- The data type determines the kind of the values that users can store in any given field. •

Each field can store data consisting of only a single data type.

Here are some of the most common data types you will find used in a typical Microsoft Access database.

Type of Data	Description	Size
--------------	-------------	------

PREPARED BY SANA MAHMOOD 10

Chapter# 17 Past Paper Question with Answers 2nd year

Short Text	Text or combinations of text and numbers, including numbers that do not require calculating (e.g. phone numbers).	Up to 255 characters.
Long Text	Lengthy text or combinations of text and numbers.	Up to 63, 999 characters.
Number	Numeric data used in mathematical calculations.	1, 2, 4, or 8 bytes (16 bytes if set to Replication ID).
Date/Time	Date and time values for the years 100 through 9999.	8 bytes
Currency	Currency values and numeric data used in mathematical calculations involving data with one to four decimal places.	8 bytes
AutoNumber	A unique sequential (incremented by 1) number or random number assigned by Microsoft Access whenever a new record is added to a table.	4 bytes (16 bytes if set to Replication ID).
Yes/No	Yes and No values and fields that contain only one of two values (Yes/No, True/False, or On/Off).	1 bit.

PREPARED BY SANA MAHMOOD 11

