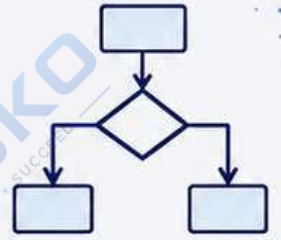


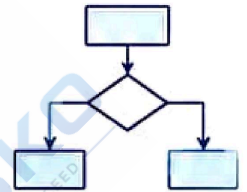
1010101010
0101010101
1010101010
0101010101



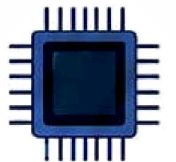
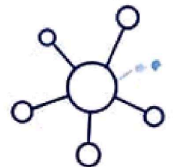
Notes



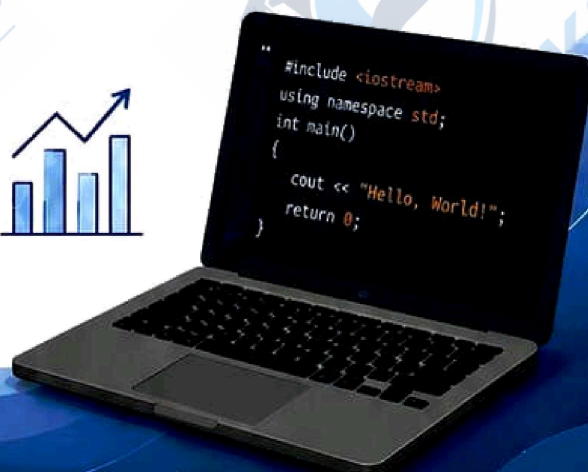
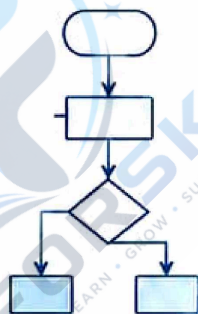
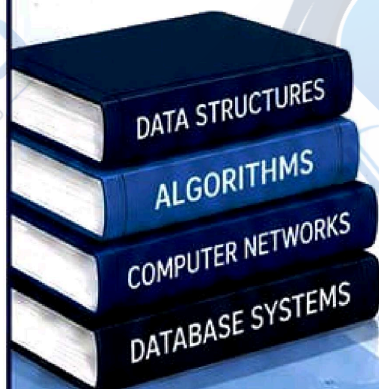
1010101010
0101010101
1010101010
0101010101



10110
01001
10101



Unit - IV



CH-4

7M

classmate

Date _____
Page _____

1. What is function?

→ A function in C is a block of code that performs a specific task. It improves code organization, reusability, and modularity.

2. What is prototyping?

→ Prototyping means declaring a function before its actual definition. It tells the compiler about the function name, return type, and parameters.

3. What is recursion?

→ Recursion is a programming technique where a function calls itself to solve a problem.

4. What is actual argument?

→ Actual arguments are the real values or variables that are passed to a function when it is called.

5. What is formal argument?

→ Formal arguments are the variables defined in the function header that receive values from the actual arguments during a function call.

6. What is local variable?

→ Local variables are variables that are declared inside a function or block and can be accessed only within that function or block.

7. What is global variable?

A global variable is a variable that stores is declared outside all functions and is accessible from any function within the same program.

8. What is library or built-in function?

A library function or built-in function provided by the C standard library to perform common tasks like input/output, math operators, etc.

9. What is user-defined function?

A user defined function in C is ~~a function in C~~ is a function that created by the programmer to perform a specific task allowing code reuse and better structure.

10. Write an example on declaration of global variables used in C languages.

→ #include <stdio.h>

int x = 5;

void show()

{

printf("x = %d\n", x);

}

int main()

{

show();

return 0;

}

11.

What is the use of void function?

→ A void function is a function that does not return any value. It is used when a function performs an action (like printing output) but does not need to send any result back.

12.

What are the two general types of function?

→

~~1. Library~~

1. Library (Built-in) Functions - predefined functions provided by C libraries.
eg - printf(), scanf(), sqrt().

2. User-defined Functions -

These functions are written by the programmer to perform specific tasks.

eg -

```
int add (int a, int b)
{
    return a+b;
}
```

13.

What is the purpose of main() function?

→

The main() function is the starting point of every C program. It is where program execution begins and usually controls the overall flow of the program.

14.

What is a string in C?

→

A string in C is a sequence of characters terminated by a null character ('\0').

It is typically stored in a character array.

15. What is the purpose of ~~main~~ the null character ($\backslash 0$) in C string?

→ The null character ($\backslash 0$) is used to mark the end of a string in C. It helps the compiler and string function know where the string ends in memory.

16. What is null string?

→ A null string is a string that contains only the null characters before it. It represents an empty string.

17. What is string constant?

→ A string constant is a sequence of characters enclosed in double quotes ($" "$). It is automatically terminated with a null character ($\backslash 0$).

18. What is character constant?

→ A character constant is a single character enclosed in single quotes ($' '$). It represents one character and is stored as an integer value in memory.

Long Answer.

Q1.

What is function? Explain its format and the term function calling and function returning.

→

A function in C is a block of code that performs a specific task. It helps in organizing programs into smaller, reusable sections, making code easier to write, read, and maintain.

Format of a Function

A function typically includes the following.

1. Function Declaration (Prototype) -

Declare the function before it is used.

```
return_type  
function_name (parameter_list);
```

2. Function Definition -

Contain actual code that perform the task

```
return_type  
function_name (parameter_list)  
{  
    // body of the function  
}
```


3. Function Call -
Executes the function.

Function name (arguments);

Function Calling -

This is the process of invoking a function from another place. Actual values are passed to the function.

Eg- `int result = add(5, 3);` // calling add for

Function Returning

A function can return a value to the place where it was called using the return keyword

```
int add(int a, int b)
{
    return a + b;
    // Returning the result
}
```

Functions help in writing modular, organized and reusable code.

✓ 2. Explain the advantages of using functions.

1. Modularity -

Functions divide program into smaller modules, making it easier to understand and manage.

2. Code Reusability -

Once a function is written, it can be reused multiple times without rewriting the same code.

3. Improved Readability -

Functions with meaningful names make the program easier to read and understand.

4. Easier Debugging -

Errors can be isolated and fixed within individual functions without affecting the whole program.

5. Reduced Code Duplication -

Repeated tasks can be handled by a single function call instead of rewriting the same code.

6. Better Collaboration -

In large projects, functions allow multiple programmers to work on different parts of the program simultaneously.

What is need and purpose of user of defined function?

➤

1. Code Reusability.

User-defined functions allow you to write a block of code once and reuse it multiple times in the program, reducing duplication.

2. Modularity.

They help break down a large program into a smaller, manageable parts, making the code more organized and easier to handle.

3. Easy Maintenance and Debugging

Since each function performs a specific task, errors can be found and fixed within that function without affecting the rest of the program.

4. Improved Readability and Clarity.

Functions with meaningful names describe what they do, making the program easier to read and understand.

5. Better Testing.

Functions can be tested separately for correctness improving the reliability of the entire program.

9m?

4.

What is function? Explain function prototype with suitable example.

→ A function in C is a block of code designed to perform a specific task. It improves program structure by breaking the program into smaller, reusable parts.

Function Prototype.

A function prototype is a declaration of a function that tells the compiler about the function's name, return type, and parameters before its actual definition. It ensures the correct usage of the function during compilation.

Syntax-

return_type
function_name (type 1, type 2, ...);

eg-

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

```
int add(int, int); //Function prototype.
```

```
int main()
```

```
{ int result = add(5, 3); //Function call  
  printf("Sum %d", result);  
  return 0; }
```

```
//Function definition
```

```
int add (int a, int b)  
{ return a+b; }
```


5. Explain different categories of function used in C?

→ In C, user-defined functions can be categorized based on whether they accept arguments and return a value.

There are 4 main categories.

1. Function with No Arguments and no return value

The function has no arguments. It does not receive any data from the calling function. Similarly, it does not return any value.

The calling function does not receive any data from the called function. So, there is no communication between calling and called function.

eg-

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

```
void greet C); //Function declaration  
int main C)
```

```
{ int greet C); // call the function  
  return 0; }
```

```
//Function definition
```

```
void greet C)  
{  
  printf("Hello ");  
}
```


classmate

Date _____
Page _____

2. Functions with arguments and no return value

The function has some arguments, its received data from the calling function but it does not return a value to the calling function.

The calling function does not receive any data from the called function. So, it is one way data communication between called and calling function.

eg-

3. Function with arguments and return values

Functions have some arguments and it receives data from the calling function. Similarly, it returns a value to the calling function. The calling function receives data from the called function. So, it is 2 way data communication between calling and called functions.

eg-

classmate

Date _____

Page _____

4. Functions with no arguments and return values.

The functions has no arguments and it does not receive any data from the calling function, but it returns data from the called function. The calling function receives data from the called function. So, it one way data communication between calling and called functions.

Q7. Explain calling function and called function.
→ In C programming, functions interact using function calls.

1. Calling Function

- i) The calling function is the function that calls or invokes another function.
- ii) It transfers the control to the called function and may also pass arguments.
- iii) After, the called function completes, control returns to the calling function.

2. Called Function.

- i) The function that is invoked or executed by another function.
- ii) It performs a specific task and may return a value to the calling function.

eg-

```
#include <stdio.h>
void display Message ( )
{
    printf("Welcome"); }
int main ( )
{
    display Message ( );
    return 0;
}
```


(9.)
→

Explain actual and formal argument?

Arguments in C are the variables that are passed to a function when it is called.

These argument allow data to be sent from the calling function to the called function for processing.

12.

Function prototype -

→ A function prototype in C declares a function before its definition. It specifies the function's name, return type, and the number and types of its parameters. The prototype acts as a blueprint, informing the compiler about the function's interface before it is actually used.

```
return_type  
function_name (data_type parameter 1,  
               data_type parameter 2,  
               ...);
```

Purpose of a Function Prototype

- i) A function prototype ensures that calls to the function are made with correct number and types of arguments.
- ii) A function prototype specifies the number of arguments.
- iii) It states the data type of each of the passed arguments.
- iv) It gives the order in which the arguments are passed to the function.

Q3. What is the difference between a function declaration and a function definition?

Function Declaration	Function Definition.
1. It is also called as function prototype	1. It is also called as function implementation.
2. It tells the compiler about the function name, return type, and parameters	2. It contains the actual code that performs the task.
3. It ends with a semicolon (;)	3. Enclosed in curly braces {} with code inside.
4. No executable code is written	4. Contains the execution body of the function.
5. It is used to declare a function before its use	5. It is used to define how the function behaves.
6. It is optional in small programs.	6. Mandatory to make the function work.
7. Helps the compiler check for correct usage	7. Actually allocates memory and runs when called
8. Eg- <code>int sum(int, int);</code>	8. eg- <code>int sum (int a, int b) {return a+b;}</code>

What is recursion? Explain with suitable examples.
 Recursion is a technique of programming where a function calls itself directly or indirectly to solve a problem. It breaks a complex problem into smaller sub-problems solving each one using same function.

Syntax -

```

return_type
function_name (parameters)
{
    if (base_condition)
        return value;
    else
        return function_name (smaller_input);
}
  
```

eg- Factorial Using Recursion:

Factorial of a number n is:

- $n! = n \times (n-1) \times (n-2) \times \dots \times 1$
- Base case - $0! = 1$

```

#include <stdio.h>

int factorial (int n)
{
    if (n == 0) // base case
        return 1;
    else
        return n *
        factorial (n-1); // recursive case
}

int main () {
    int num = 5;
    printf ("Factorial of %d, num, factorial (num));
    return 0;
}
  
```


15. What is function recursion? Write a program to calculate factorial of given number using recursion?



Function recursion is a process in which a function calls itself directly or indirectly to solve a problem. It is commonly used for problems that can be broken into smaller, similar subproblems.

Program to Calculate Factorial Using Recur. - Stop

```
#include <stdio.h>
// Recursive function to calculate factorial

int factorial(int n)
{
    if (n == 0) // Base case
        return 1;
    else
        return n *
        factorial (n-1); // Recursive call
}

int main()
{
    int num;
    printf ("Enter a number");
    scanf ("%d", &num);

    if (num < 0)
        printf ("Factorial is not defined");
    else
        printf ("Factorial is %d", num, factorial (num));
    return 0;
}
```


16. What are advantages of Recursion?

→

1. Simplifies Complex Problems.

Recursion is useful for solving problems that can be broken down into smaller sub-problem such as factorial, Fibonacci series, or tree traversal.

2. Reduces Code Length.

Recursive functions often require fewer lines of code compared to their iterative equivalents, making the code cleaner and easier to read.

3. Natural for Certain Problems.

Some problems like directory structures, tree and graph traversal and mathematical computations.

4. Supports Divide and Conquer.

Recursive techniques are ideal for divide and conquer algorithms like merge sort, quick sort, and binary search.

5. No need for loops in some cases.

Recursion can eliminate the need for complex loops, especially when the logic is naturally recursive.

18. declare and initialize.

✓ 17. What is a string in C, and how is it represented in memory?

→ 1. In C, a string is a sequence of characters terminated by a special character called the null character ('`\0`').

2. Declaration & Initialization of a String.

There are two common ways to declare and initialize strings in C.

a. Using double quotes.

`char str1[] = "Hello";`

This automatically adds the null terminator '`\0`' at the end.

b. Using character array manually.

`char str2[] = {'H', 'e', 'l', 'l', 'o', '\0'};`

3. Memory Representation.

eg- `char str[] = "Code";`

This will be stored in memory as.

Index	0	1	2	3	4	5
Value	'C'	'o'	'd'	'e'	'\0'	

Each character occupied 1 byte of memory.

49. Explain `strcat()`, and `strcmp()` function with suitable example.

→ 1. `strcat()` - String Concatenation.

This function concatenates the source string at the end of the target string.

eg - `#include <stdio.h>`
`#include <string.h>`

`void main()`

```
{  
    char source[] = "Friends!";  
    char target[30] = "Hello";  
    strcat(target, source);  
    printf("\n source string = %s", source);  
    printf("\n target string = %s", target);  
}
```

Output - Source ^{string} ~~code~~ = Friends!
 targeted string = Hello Friends!

200 Explain with example: strcpy() & strlen().

1. strcpy() - This function copies the contents of one string into another. The base addresses of the source and target strings should be supplied to this function.

eg-

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

```
#include <string.h>
```

```
int main ()
```

```
{
```

```
    char source[] = "Sayonara";
```

```
    char target[20];
```

```
    strcpy (target, source);
```

```
    printf ("Source string = %s", source);
```

```
    printf ("Target string = %s", target);
```

```
}
```

Output-

Source string = Sayonara

target string = Sayonara

On supplying the base addresses, strcpy() goes on copy the characters in source string in to target string.

2. strlen() -

This function counts the number of characters present in a string. Its usage is illustrated in the following program.

eg-

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

```
void main()
{
```

```
    char arr[] = "Amaravati";
```

```
    int len1, len2;
```

```
    len1 = strlen(arr);
```

```
    len2 = strlen("Jack & Jerry");
```

```
    printf("\n Length of string %s is = %d", arr, len1);
```

```
    printf("\n length of string %s is = %d", "Jack & Jerry", len2);
```

```
}
```

The Output

Length of string Amaravati is = 8.

Length of string Jack & Jerry is = 14.

22.

Explain the following string functions.

i) $\rightarrow$

strcat () -

This function concatenates the source string at the end of the target string.

For example -

"Bombay" and "Nagpur" on concatenates would be result into a string "BombayNagpur".

```
#include <stdio.h>
#include <string.h>
void main()
{
    char source[ ] = "Bombay";
    char target[20] = "Nagpur";
    strcat ( target, source );

    printf ( "\n source string %s, source );
    printf ( "\n target string %s, target );
}
```

Output

source string ~~+~~ Bombay
target string NagpurBombay.

string "

ii) $\rightarrow$ strcpy C) - This function copies the contents of one string into another.

eg-

```
#include <stdio.h>
#include <string.h>
int main()
{
    char source[] = "Sayonara";
    char target[20];
    strcpy(target, source);
```

```
printf("source string %s", source);
printf("target string %s", target);
}
```

Output

source string Sayonara.
target string Sayonara.

iii)
→strcmp()

This function compares two strings to find out whether they are same or different. The two strings are compared character by character until there is a mismatched or end of one of the string is reached. If the two strings are identical, strcmp() return the value zero. If they are not - it return the first value of non-matching characters.

eg -

```
#include <stdio.h>
#include <string.h>
void main ()
{
    char s1[] = "Rinky";
    char s2[] = "Pinky";
    int i, j;
    i = strcmp(s1, "Rinky");
    j = strcmp(s1, s2);

    printf("In %d %d", i, j);
}
```

Output.

0 2

iv) strlen() -

→ This function counts the number of characters present in a string. Its illustrated in following program.

eg -

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

```
void main()
{
    char arr[] = "Amravati";
    int l1, l2;

    l1 = strlen(arr);
    l2 = strlen("Jack and Jerry");
```

```
printf("In length of string %s is %d", arr, l1);
printf("In length of string %s is %d", "Jack
and Jerry", l2);
}
```

Output.

Length of string Amravati is 8.

Length of string Jack and Jerry is 14.