CHAPTER NO 5: -(FUNCTIONS)
SHORTS QUESTIONS/ANSWERS

1. Define problem solving?
Problem solving is the process of identifying a problem, developing possible solution paths, and taking the appropriate course of action. It involves analyzing the situation, breaking down the problem into manageable parts, and systematically working through the steps to reach a solution.

2. Define divide and conquer?
Divide and conquer is an algorithm design paradigm that involves breaking a problem into smaller sub problems of the same type, solving each sub problem recursively, and then combining their solutions to solve the original problem.
Common examples: - include merge sort and quicksort

3. Define function?
A function is a self-contained block of code designed to perform a specific task. It takes inputs (called parameters or arguments), processes them, and returns an output (return value).

4. How many types of a function?
Functions can be broadly categorized into:
1. Built-in functions (provided by the programming language)
2. User-defined functions (created by the programmer)
3. Lambda functions (anonymous functions)

5. Differentiate between built-in and user defined function?
Built-in functions are predefined and provided by the programming language (e.g., print() in Python).

User-defined functions are created by the programmer to perform specific tasks.

6. How many advantages of a functions?
Advantages of using functions include:
a. Code reusability
b. Improved readability
c. Easier maintenance
d. Modular programming
e. Efficient debugging and testing
f. Handling the complexity
g. Divide and conquer
h. Separation of task

7. Differentiate between reusability and readability?
Reusability refers to the ability to use code across multiple parts of a program or in different programs without modification.
Readability refers to how easily the code can be read and understood by humans

8. Define separation of task?
Separation of task involves dividing a program into distinct functions, each responsible for a specific task. This makes the code more organized, easier to manage, and maintain

9. Define parameter and return value?
A **parameter** is an input variable passed to a function, while a **return value** is the output that the function provides after execution

10. Define function signature?

A function signature defines the function's
name, its parameter types, and the return type. It specifies what inputs the function takes and what output
it produces

syntax: -

  Return_type function_name(data_type1, data_type2,.....,,,);

11. Write down the syntax of a function signature?

Syntax: -

  return_type function_name(data_type1, data_type2, ...);

12. Enlist the parts of a function definitions with syntax?

Parts of a function definition include:

- Function signature (return type, function name, variable)
- Function body (the block of code to execute)

return_type function_name(data_type1 var, daya_type2 var, ...) {
    // Function body
}

Example: -int add(int a, int b);

13. What general structure is used to make a function call?

The general structure of calling a function are below: -

  function_name(argument1, argument2, ...);

14. Differentiate between parameters and arguments?

Parameters are variables listed in the
function definition.

Arguments are the actual values passed to the
function.

15. What points must be kept in mind for the arrangement of functions in a program?

Points to consider:

- Group related functions together
- Place the main function at the beginning or end
- Keep the program flow logical
- Document each function with comments

16. What do u know about the function of C level language?

In C, functions are used to modularize
code. C functions can be standard library
functions or user-defined. They are declared
with a return type, function name, and
parameters, and defined with a function body.

17. What is the use of function signature?

A function signature helps in identifying the
function, ensuring the correct types and
number of parameters are used, and
aids in function overloading (in languages that
support it).

18. What is int square(int): function signature?

It is a function signature indicating a
function named square that takes one
integer parameter and returns an integer.

19. What is int float parameter (float, float): function signature?

It appears there might be a typo. Correctly, it
should be: -
int some_function(float param1, float param2);

20. What is int largest (int, int, int): function signature?

It is a function signature indicating a function
named largest that takes three integer parameters
and returns an integer.

21. What is float area (float): function signature?

It is a function signature indicating a function
named area that takes one float parameter and
returns a float.

22. What is int isvowel(char): function signature?

It is a function signature indicating a function
named isvowel that takes one-character parameter
and returns an integer.

23. What is the general structure of defining a function?
   General structure/syntax
   return_type function_name(parameter_type1 parameter1, parameter_type2 parameter2, ...) {
     // Function body
   }

24. Which points important for the arrangement of a function in a program?
Important points:
- *Function declarations should precede their usage*
- *Group related functions*
- *Keep utility functions together*
- *Use comments for clarity*
- *Maintain logical flow and readability*

LONG QUESTIONS

1. Write a note on advantages of function?
   There are many advantages of a function
   a) *Code reusability*
      Reusability *refers to the ability to use code across multiple parts of a program or in different programs without modification.*
   b) *Improved readability*
      Readability *refers to how easily the code can be read and understood by humans*
   c) *Easier maintenance*
   d) *Modular programming*
   e) *Efficient debugging and testing*
   f) *Handling the complexity*
   g) *Divide and conquer*
   h) *Separation of task*
      Separation *of task involves dividing a program into distinct functions, each responsible for a specific task. This makes the code more organized, easier to manage, and maintain*

2. Write a note on types of a function?
   Functions can be broadly categorized into:
   1. *Built-in functions (provided by the programming language)*
         Built-in *functions are predefined and provided by the programming language (e.g., print() in Python).*
   2. *User-defined functions (created by the programmer)*
         User-defined *functions are created by the programmer to perform specific tasks.*
   3. *Lambda functions (anonymous functions)*
   4. *Calls back Function*

3. Write a note on function?
   A function is a self-contained block of code designed to perform a specific task. It takes inputs (called parameters or arguments), processes them, and returns an output (return value).
   In C, functions are used to modularize code. C functions can be standard library functions or user-defined. They are declared with a return type, function name, and parameters, and defined with a function body.

Important for the arrangement of a function in a program
Important points:
- *Function declarations should precede their usage*
- *Group related functions*
- *Keep utility functions together*
- *Use comments for clarity*
- *Maintain logical flow and readability*

Types of a function
Functions can be broadly categorized into:
1. *Built-in functions (provided by the programming language)*
2. *User-defined functions (created by the programmer)*
3. *Lambda functions (anonymous functions)*
4. *Calls back Function*

Built-in *functions are predefined and provided by the programming language (e.g., print() in Python).*

User-defined *functions are created by the programmer to perform specific tasks.*

Advantages of a functions
Advantages of using functions include:
- i) *Code reusability*
- j) *Improved readability*
- k) *Easier maintenance*
- l) *Modular programming*
- m) *Efficient debugging and testing*
- n) *Handling the complexity*
- o) *Divide and conquer*
- p) *Separation of task*

Function signature
A function signature defines the function's name, its parameter types, and the return type. It specifies what inputs the function takes and what output it produces.
syntax: -
   Return_type function_name(data_type1, data_type2,.....,,,);

Parts of a function definitions with syntax
Parts of a function definition include:
- *Function signature (return type, function name, variable)*
- *Function body (the block of code to execute)*

return_type function_name(data_type1 var, daya_type2 var, ...) {
   // Function body
}
Example: -int add(int a, int b);

LONG CODING QUESTIONS

1. **Write a function to calculate the factorial of any number?**
   Code: -
   ```
   #include <stdio.h>
   unsigned long long factorial(int n) {
       if (n < 0) {
           printf("Factorial of negative number doesn't exist.\n");
           return 0;
   ```

```c
    }
    unsigned long long fact = 1;
    for (int i = 1; i <= n; ++i) {
        fact *= i;
    }
    return fact;
}
int main() {
    int number;
    printf("Enter a number to calculate its factorial: ");
    scanf("%d", &number);
    unsigned long long result = factorial(number);
    if (result != 0) {
        printf("Factorial of %d is %llu\n", number, result);
    }
    return 0;
}
```

2.   Write a function to print the table?
Code: -
```c
#include <stdio.h>
void printTable(int number) {
    for (int i = 1; i <= 10; ++i) {
        printf("%d x %d = %d\n", number, i, number * i);
    }
}
int main() {
    int number;
    printf("Enter a number to print its multiplication table: ");
    scanf("%d", &number);
    printTable(number);
    return 0;
}
```

3.   Write a function which takes the amount and the interest percentage and return the interest amount?
Code: -
```c
#include <stdio.h>
double calculateInterest(double amount, double interestRate) {
    return (amount * interestRate) / 100.0;
}
int main() {
    double amount, interestRate;
    printf("Enter the principal amount: ");
    scanf("%lf", &amount);
    printf("Enter the interest rate (percentage): ");
    scanf("%lf", &interestRate);
    double interest = calculateInterest(amount, interestRate);
    printf("The interest amount is: %.2lf\n", interest);
    return 0;
}
```

4. **Write a function that find and display the area of a triangle, parallelogram, rhombus of trapezium according to the choice of user.**

Code(if-else-if)	Code(Switch case)

```c
#include <stdio.h>
// Function prototypes
float calculateTriangleArea(float base, float height);
float calculateParallelogramArea(float base, float height)
float calculateRhombusArea(float diagonal1, float
diagonal2);
float calculateTrapeziumArea(float base1,
float base2, float height);
int main() {
    int choice;
    float base, height, base1, base2, diagonal1, diagonal2,
area;
    // Display the menu
    printf("Choose any shape on your mind (1-4):\n 1.
Triangle\n 2. Parallelogram\n 3. Rhombus\n 4.
Trapezium \n");
    scanf("%d", &choice);
    if (choice == 1) {
      printf("Enter the base of a Triangle:\t");
      scanf("%f", &base);
      printf("Enter the height of a Triangle:\t");
      scanf("%f", &height);
      area = calculateTriangleArea(base, height);
      printf("The area of a Triangle is: %.2f\n", area);
    } else if (choice == 2) {
      printf("Enter the base of a Parallelogram:\t");
      scanf("%f", &base);
      printf("Enter the height of a Parallelogram:\t");
      scanf("%f", &height);
      area = calculateParallelogramArea(base, height);
      printf("The area of a Parallelogram is: %.2f\n",
area);
    } else if (choice == 3) {
      printf("Enter the 1st diagonal of Rhombus:\t");
      scanf("%f", &diagonal1);
      printf("Enter the 2nd diagonal of Rhombus:\t");
      scanf("%f", &diagonal2);
      area = calculateRhombusArea(diagonal1,
diagonal2);
      printf("The area of a Rhombus is: %.2f\n", area);
    } else if (choice == 4) {
      printf("Enter the 1st base of Trapezium:\t");
      scanf("%f", &base1);
      printf("Enter the 2nd base of Trapezium:\t");
      scanf("%f", &base2);
      printf("Enter the height of Trapezium:\t");
      scanf("%f", &height);
      area = calculateTrapeziumArea(base1, base2,
height);
      printf("The area of a Trapezium is: %.2f\n", area);
    } else {
      printf("@@@@@ Invalid Choice @@@@@\n");
    }
    return 0;
```

```c
#include <stdio.h>
using namespace std;
// Function prototype
void findAndDisplayArea(int choice);

int main() {
    int choice;

    // Display the menu
    printf("1. Triangle:\n");
    printf("2. Parallelogram:\n");
    printf("3. Rhombus:\n");
    printf("4. Trapezium:\n");
    printf("Choose any shape on your mind (1-4):\t");
    scanf("%d", &choice);

    // Find and display the area based on user choice
    findAndDisplayArea(choice);

    return 0;
}

// Function definition
void findAndDisplayArea(int choice) {
    float base, height, base1, base2, diagonal1,
diagonal2, area;

    switch (choice) {
    case 1:
    printf("Enter the base of a Triangle:\t");
    scanf("%f", &base);
    printf("Enter the height of a Triangle:\t");
    scanf("%f", &height);
    area = 0.5 * base * height;
    printf("The area of a Triangle is: %.2f\n", area);
    break;

    case 2:
    printf("Enter the base of a Parallelogram:\t");
    scanf("%f", &base);
    printf("Enter the height of a Parallelogram:\t");
    scanf("%f", &height);
    area = base * height;
    printf("The area of a Parallelogram is: %.2f\n",
    area);
    break;

    case 3:
    printf("Enter the 1st diagonal of a Rhombus:\t");
    scanf("%f", &diagonal1);
    printf("Enter the 2nd diagonal of a Rhombus:\t");
    scanf("%f", &diagonal2);
    area = 0.5 * diagonal1 * diagonal2;
    printf("The area of a Rhombus is: %.2f\n", area);
```

```
                                                    break;
}
// Function definitions                               case 4:
float calculateTriangleArea(float base, float height) {   printf("Enter the 1st base of a Trapezium:\t");
    return 0.5 * base * height;                           scanf("%f", &base1);
}                                                         printf("Enter the 2nd base of a Trapezium:\t");
float calculateParallelogramArea(float base,              scanf("%f", &base2);
float height) {                                           printf("Enter the height of a Trapezium:\t");
    return base * height;                                 scanf("%f", &height);
}                                                         area = 0.5 * (base1 + base2) * height;
float calculateRhombusArea(float diagonal1,               printf("The area of a Trapezium is: %.2f\n", area);
float diagonal2) {                                        break;
    return 0.5 * diagonal1 * diagonal2;
}                                                      default:
float calculateTrapeziumArea(float base1, float base2,    printf("@@@@@Invalid choice@@@@@\n");
height) {                                                 break;
    return 0.5 * (base1 + base2) * height;            }
}
                                                  }
```

5. **Write a function that takes two input as integers and ask the user to enter the choice 1 to 4. The program should perform the operation according to the given table and display the result:**

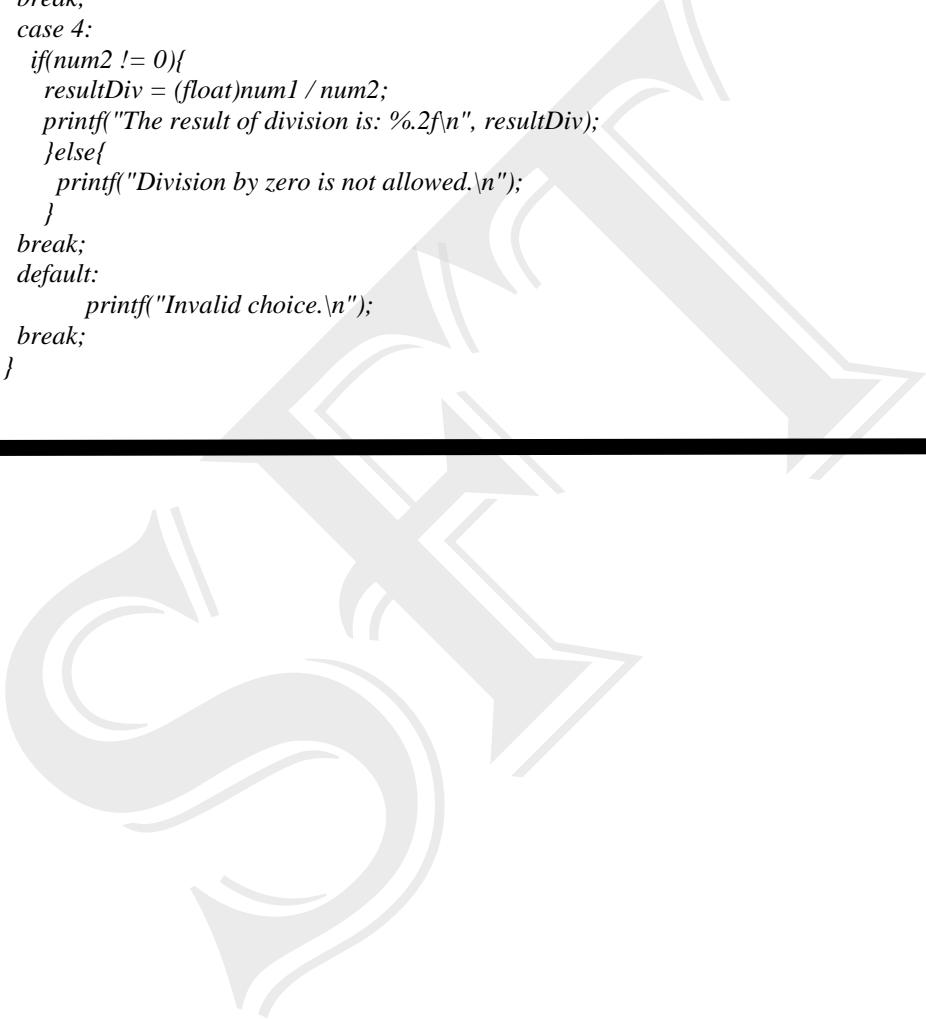| Choice	Operation
1	Addition
2	Subtraction
3	Multiplication
4	Division

Code: -

```
#include <stdio.h>
using namespace std;
// Function prototype
void performOperation(int choice, int num1, int num2);
int main() {
  int choice, num1, num2;
  // Input two integers
  printf("Enter the first integer: ");
  scanf("%d", &num1);
  printf("Enter the second integer: ");
  scanf("%d", &num2);
  // Display the menu
  printf("1. Addition\n");
  printf("2. Subtraction\n");
  printf("3. Multiplication\n");
  printf("4. Division\n");
  printf("Choose an operation (1-4): ");
  scanf("%d", &choice);
  // Perform the operation based on user choice
  performOperation(choice, num1, num2);
  return 0;
}
// Function definition
void performOperation(int choice, int num1, int num2) {
  int result;
  float resultDiv;
```

```
switch(choice){
        case 1:
    result = num1 + num2;
    printf("The result of addition is: %d\n", result);
   break;
   case 2:
    result = num1 - num2;
    printf("The result of subtraction is: %d\n", result);
   break;
        case 3:
    result = num1 * num2;
    printf("The result of multiplication is: %d\n", result);
   break;
   case 4:
    if(num2 != 0){
      resultDiv = (float)num1 / num2;
      printf("The result of division is: %.2f\n", resultDiv);
     }else{
       printf("Division by zero is not allowed.\n");
     }
   break;
   default:
        printf("Invalid choice.\n");
   break;
  }
 }
```

MCQ'S, ERRORS & OUTPUTS

MCQs

Sr. No	Question	A	B	C	D
1	A good _____ approach is to divide the problem into multiple smaller parts of sub-problems.	functional	problem solving	definition a problem	none
2	The step to divide large problem into small problems is called ___	Divide and conquer	Dividing a problem	Searching a problem	Analyzing a problem
3	A _____ is a block of statements which performs a particular task.	Computer	program	function	Parameters
4	_____ is a function that is used to display anything on computer screen.	Printf	scanf	getch	printf
5	____ is another function that is used to take input from the user.	scanf	getch	int	printf
6	Types of function in C language ____	Built in Function	Both A and C	User defined function	None
7	Which one from the following is not type of function?	Built in Function	User defined function	Pre-defined function	C is not type of function
8	The functions which are available in C Standard Library are called _____	User defined fiction	Pre-defined function	built-in functions	All
9	Which functions used to perform mathematical calculations, string operations, input/output operations etc?	Built-in functions	User defined fiction	Pre-defined function	None
10	printf and scan are _____	Constant function	Arguments function	User defined function	Built-in functions
11	The functions which are defined by a programmer are called _____	library function	user-defined functions	one two function	returning function
12	What are the advantages of a functions?	Reusability	Separation of task	Readability	All
13	Which one from the following is not a advantage of a function?	Handling the complexity of the problem	Reusability	Accessibility	Readability
14	_____ functions provide reusability of code.	Reusability	Readability	Separation of task	none
15	_____ functions allow us to separate the code of one task from the code of other tasks.	Separation of problems	Separation of task	Alteration of code	Separation of function
16	_____ divide the program into smaller units, and thus reduce the complexity of the problem.	programmer	variables	Functions	statements

#	Question	A	B	C	D
17	Dividing the program into multiple functions, improves the _____ of the program.	Complicity	Reusability	Accessibility	Readability
18	Inputs of a function are called _____ of the function.	parameters	value	Index	subscript
19	Output of the function is called its _____	index value	input value	return value	output value
20	_____ is used to define the inputs and output of a function.	Function index	Function signature	Function value	Function uses
21	What are the examples of function signature?	int square (int),	float area (float);	int largest (int, int, int);	All
22	A function that takes an integer as input and returns its square, its function signature is _____.	int square (int);	int largest (int, int, int);	float area (float);	int is Vowel (char);
23	A function that takes length and width of a rectangle as input and returns the perimeter of the rectangle, its function signature is _____	float area (float);	int isVowel (char);	float perimeter (float, float);	None
24	A function that takes three integers as input and returns the largest value among them, its function signature is _____	int isVowel (char);	int largest (int, int, int);	int square (int);	float area (float);
25	A function that takes radius of a circle as input and returns the area of circle, its function signature is _____	float area (float);	int is Vowel (char);	int isVowel (char);	All
26	A function that takes a character as input and returns 1, if the character is vowel, otherwise return 0, its function signature is _____	char area (float);	float area (float);	int isVowel (char);	All
27	_____ is the set of statements which are executed in the functions to perform the specified task.	Body of the function	Set of function	Header of function	None
28	A function cannot return more than _____ values.	One	Two	Three	Four
29	Following is the general structure used to make a function call.	Function_body(value1, value2, , valueN);	Function_name (value1, value2, , valueN);	Function_name(value 1, value2, , value3);	Function (value1, value2, , valueN);
30	The values passed to the function are called _____	terminator	functions	statement	arguments
31	Variables in the function definition that receive the values are called _____ of the function.	superscript	subscript	parameters	index

Keys:

1	2	3	4	5	6	7	8	9	10
B	A	C	D	A	B	D	C	A	D
11	12	13	14	15	16	17	18	19	20
B	D	C	A	B	C	D	A	C	B
21	22	23	24	25	26	27	28	29	30
D	A	C	B	A	C	A	A	B	D
31									
C									

29. What is the output of the following program?

```
main()
{
int x=20;
int y=10;
swap(x,y);
printf("%d %d",y,x+2);
}
swap(int x, int y)
{
int temp;
temp =x;
x=y;
y=temp;
}
```

Output:
10 22 (duplicate copy of arguments are generated on calling a function)

30. What will be the output of the following code ?

```
output()
{
printf("%p",output);
}
```

Ans. Some address will be printed as function names are just addresses. Similarly output() is also a function and its address will be printed.

31. What will be the output of the following code?

```
main()
{
int i;
printf("%d",scanf("%d",&i)); // value 10 is given as input here
}
```

Ans.1 as scanf returns number of items read successfully. So number of items read is 1.

32. What will be the output of the C program?

```
#include<stdio.h>
int main()
{
function();
return 0;
}
void function()
{
printf("Function in C is awesome");
}
```

Output: Compilation error.

33. What will be the error of the C program?

```c
void message ();
{
printf ("Hope you are fine :");
return 23;
```

Errors: After function definition there is no semicolon.

34. What will be the error of the C program?

```c
int max (int a; int b)
{
        if (a > b)
                return a;
        return b;
}
```

Errors: Between parameters use comma instead of semicolon.