CH # 5

FUNCTIONS

5.1 FUNCTIONS

5.1.1 TYPES OF FUNCTIONS
5.1.2 ADVANTAGES OF FUNCTIONS
5.1.3 SIGNATURE OF A FUNCTION
5.1.4 DEFINING A FUNCTION

LONG QUESTIONS

1. Define Functions. Explain its different types. (K.B+U.B)

Ans: "A function is a block of statements which performs a particular task"

Example:

printf is function that is used to display anything on computer screen, ***scanf*** is another function that is used to take input from the user. Each program has a main function which performs the tasks programmed by the user. Similarly, we can write other functions and use them multiple times.

General Syntax of a Function:

A function definition has the following general structure.

return_type function_name (data_type var1, data_type var2,..,data_type varN)
{
Body of the function
}

Types of Functions:

There are basically two types of function:

1. Built-in Functions
2. User Defined Functions

Built-in Functions:

The functions which are available in 'C' Standard Library are called Built-in Functions. These functions perform commonly used mathematical calculations, string operations, input/ output operations etc. For example ***printf*** and ***scanf*** are built-in functions.

User Defined Functions:

The functions which are defined by a programmer are called user-defined functions.

2. Define functions. Also discuss the advantages of function. (K.B+U.B)

Ans: "A function is a block of statements which performs a particular task"

Advantages of Functions:

Functions provide us several advantages.

1. Reusability:

Functions provide reusability of code. It means that whenever we need to use the functionality provided by the function, we just call the functions. We do not need to write the same set of statements again and again.

2. Separation of tasks:

Functions allow us to separate the code of one task from the code of other tasks. If we have a problem in one function, then we do not need to check the whole program for removing the problem. We just need to focus at one single function.

3. Handling the complexity of the problem:

If we write the whole program as a single procedure, management of the program becomes difficult. Functions divide the program into smaller units, and thus reduce the complexity of the problem.

4. Readability:

Dividing the program into multiple functions, improves the readability of the program.

3. Define Functions. How can a function be defined? (K.B+U.B)

Ans: "A function is a block of statements which performs a particular task"

The function signature does not describe how the function performs the task assigned to it. Function definition does that.

General Syntax of a Function:

A function definition has the following general structure.

return_type function_name (data_type var1, data_type var2,..,data_type varN)
{
Body of the function
}

Body of the function is the set of statements which are executed in the function to perform the specified task.

We need to call a function, so that it performs the programmed task. Following is the general structure used to make a function call.

function_name (value1, value2,.., valueN);

There may be multiple ***return*** statement in a function but as soon as the first return statement is executed, the function call returns and further statements in the body of function are not executed. Return is a keyword that is used to return a value to the calling function. Output of the function is called its ***return value.*** A function cannot return more than one value. If we try to get more than one value then compiler gives an error. There may be multiple return statement in a function but as soon as the first return statement is executed, the function call returns and further statements in the body of function are not executed.

Example:

```
void show pangram ()
{
printf ("\nA quick brown fox jumps over the lazy dog. \n");
}
```

As the above function does not return anything thus return type of the function is void.

SHORT QUESTIONS

1. Define Divide and Conquer Rule. (K.B)

Ans: In divide and conquer rule the problem is decomposed into sub problems. Rather on concentrating the bigger problem as a whole we try to solve each sub problem separately. This leads to a simple solution.

2. Define Functions. (K.B)

Ans: A function is a block of statements that performs a particular task. e.g. ***printf*** is function that is used to display anything on computer screen, ***scanf*** is another function that is used to take input from the user. Each program has a main function which performs the tasks programmed by the user. Similarly, we can write other functions and use them multiple times.

3. Give general syntax of Functions. (K.B+U.B+A.B)

Ans: A function definition has the following general structure.

return_type function_name (data_type var1, data_type var2,..,data_type varN)
{
Body of the function
}

4. Write down the name of types of Functions. (K.B)

Ans: There are two types of function:
1. Built-in Functions **2.** User Defined Functions

5. Define Built-in Functions. (K.B)

Ans: The functions that are available in C standard library are called built-in Functions. These functions perform commonly used mathematical calculations, string operations, input/ output operations etc. For example, ***printf*** and ***scanf*** are built-in functions.

6. Define User- Define Functions. (K.B)

Ans: The functions that are defined by a programmer are called user-defined functions.

7. Define reusability. (K.B)

Ans: Functions provide reusability of code. It means that whenever we need to use the functionality provided by the function, we just call the functions. We do not need to write the same set of statements again and again.

8. What is separation of tasks? (K.B)

Ans: Functions allow us to separate the code of one task from the code of other tasks. If we have a problem in one function, then we do not need to check the whole program for removing the problem. We just need to focus at one single function.

9. How can a function handle the complexity of a program? (U.B)

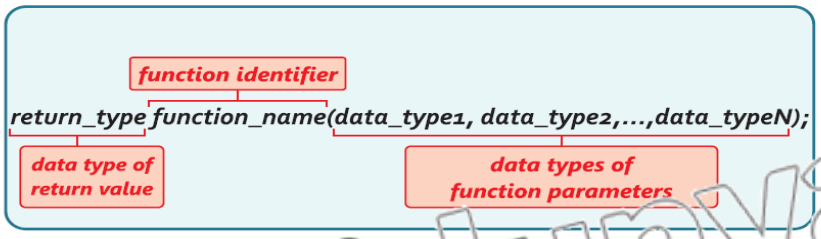
Ans: If we write the whole program as a single procedure, management of the program becomes difficult. Functions divide the program into smaller units, and thus reduce the complexity of the problem.

10. Define readability of a program. (K.B)

Ans: Dividing the program into multiple functions, improves the readability of the program.

11. What is signature of a function? (K.B)

Ans: A function is a block of statement that gets some inputs and provides some output. Inputs of a function are called ***parameters*** of the function and output of the function is called its ***return value.*** A function can have multiple parameters, but it cannot return more than one values. ***Function signature*** is used define the inputs and output of a function. The general structure of a function signature is as follows:



12. Show the descriptions of the following functions.

i. int square ii. float perimeter iii. int largest
iv. float area v. int isVowel

Ans:

Function Description	Function Signature
A function that takes an integer as input and returns its square.	int square (int);
A function that takes length and width of a rectangle as input and returns the perimeter of the rectangle	float perimeter (float , float);

A functions that takes three integers as input and returns the largest value among them.	int largest (int, int, int);
A function that takes radius of a circle as input and returns the area of circle.	float area (float);
A function that takes a character as input and returns 1, if the character is a vowel, otherwise returns 0.	int isvowel (char);

13. How can a function be called? (U.B)

Ans: Calling a function is the set of statements which are executed in the function to fulfil the specified task. Calling a function means to transfer the control to that particular function. During the function call, the values passed to the function are called ***arguments***. We can call a user-defined function from another user defined function, same as we call other functions in main function.

14. What is the purpose of the keyword 'return'? (K.B)

Ans: ***return*** is a keyword that is used to return a value to the calling function. Output of the function is called its **return value.** A function cannot return more than one value. If we try to get more than one value, then compiler gives an error. There may be multiple return statement in a function but as soon as the first return statement is executed, the function call returns and further statements in the body of function are not executed.

15. Define arguments. (K.B)

Ans: The values passed to the function are called ***arguments*.**

16. Define parameters. (K.B)

Ans: The variables in the function definition that receive these value is called ***parameters*** of the function.

17. Write down the difference between arguments and parameters. (K.B+U.B)

Ans: The values passed to the function are called ***arguments,*** whereas variables in the function definition that receive these value are called ***parameters*** of the function.

18. What points should be kept in mind for the arrangement of function? (K.B+U.B)

Ans: Following points must be kept in mind for the arrangement of function in a program.

1. If the definition of called function appears before the definition of called function, then function signature is not required.
2. If the definition of called function appears after the definition of calling function, then function signature of called function must be written before the definition of calling function.

MULTIPLE CHOICE QUESTIONS

1. Instead of solving the whole problem at once we try to apply _____ approach. (K.B+U.B)

(A) Divide & Conquer (B) Reusability
(C) Readability (D) None of these

2 _____ is a block of statements, which performs a particular task. (K.B)

(A) Program (B) Function (C) Both A & B (D) None of these

3. There are _____ type of function: (K.B)

(A) 1 (B) 2 (C) 3 (D) 4

4. _____ is an input function. (K.B)

(A) *printf* *(b) scanf* (C) Both A & B (D) None of these

5. _____ is an output function. (K.B)

(A) *printf* (B) *scanf* (C) Both A & B (D) None of these

6. Each program has _____ function. (K.B)

(A) *printf* (B) *scanf* (C) main (D) None of these

7. A function is a _____ of statements. (K.B)

(A) Block (B) Pattern (C) Standard (D) None of these

8. _____ are also known as library function. (K.B)

(A) Built in (B) User define function

(C) Both (D) None of these

9. Built in function is called_____ function. (K.B)

(A) Standard (B) Library (C) Predefined (D) All of these

10. Built in functions commonly performs _____. (K.B+U.B)

(A) Mathematical problems (B) String operations

(C) Input/ Output operation (D) All of these

11. _____ is type of built in function. (K.B)

(A) *printf* (B) *scanf* (C) Both A & B (D) None of these

12. User defined functions are defined by _____. (K.B)

(A) Programmer (B) Library (C) Developer (D) None of these

13. _____ functions are written by user. (K.B)

(A) User define (B) Built in (C) Both A & B (D) None of these

14. Functions provides _____ of code. (K.B+U.B)

(A) Reusability (B) Readability (C) Both A & B (D) None of these

15. We _____ need to write the code again and again in function. (K.B)

(A) Do (B) Do not (C) Can be both (D) None of these

16. We can increase the readability of program by _____ it. (K.B)

(A) Dividing (B) Multiplying (C) Reusing (D) None of these

17. Dividing a program increases _____ of program. (K.B+U.B)

(A) Reliability (B) Readability (C) Handling (D) None of these

18. Parameters are _____ of function. (K.B)

(A) Input (B) Output (C) Both A & B (D) None of these

19. ____ are inputs of function. (K.B)

(A) Parameters (B) Arguments (C) Both A & B (D) None of these

20. A function can have multiple _____. (K.B)

(A) Returns (B) Values (C) Parameters (D) None of these

21. A function can return _____ values. (K.B)

(A) One (B) Two (C) Three (D) Multiple

22. Function signature is used to defined_____. (K.B)

(A) Input (B) Output (C) Both A & B (D) None of these

23. int square(int); will _____. (K.B+U.B)
(A) Takes integer as input
(B) Takes the square of number
(C) Takes integer as input and return its square
(D) None of these

24. How many data types can be used while defining a function? (K.B+U.B)
(A) 1 (B) 2 (C) 3 (D) Multiple

25. What will be happen if we try to get more than one value from a function? (K.B+U.B)
(A) It will return multiple values (B) It will not return any value
(C) Compiler gives an error (D) None of these

26. How many return statements can be used with a function? (K.B)
(A) 1 (B) 2 (C) 3 (D) Multiple

27. When the first return statement is executed. (K.B)
(A) Further statement in body returns one by one
(B) Further statement stops executions
(C) Further statement in body executes in group
(D) None of these

28. Following is the general structure used to make a function call. (K.B)
(A) function _name (value1, value2…..valueN);
(B) function_name (value);
(C) function_name (value1, value2….valueN)
(D) None of these

29. _____ are the values passed to the functions. (K.B+U.B)
(A) Arguments (B) Parameters (C) Functions (D) None of these

30. Variables that receives the values in function defined are called _____.(K.B+U.B)
(A) Arguments (B) String case (C) Parameters (D) None of these

31. If the definition of function appears before the definition of called function then _____. (K.B+U.B)
(A) Signature function is required (B) Function signature is not required
(C) Error occurs (D) None of these

32. If the function of called function appears after the definition of calling function then _____. (K.B+U.B)
(A) Signature function is not required
(B) Function signature of called function must be written before the definition of calling function
(C) Error occurs
(D) None of these

33. The name of function should relate its _____. (K.B+U.B)
(A) Arguments (B) Parameters (C) Tasks (D) None of these

34. Calling a function means transfer the _____ to that Particular function. (K.B+U.B)
(A) Control (B) Path (C) Route (D) None of these

PROGRAMMING TIME (A.B)

Programming Time 5.1

Problem:
Write a function is Prime( ) that takes a number as input and returns 1 if the input number is prime, otherwise returns 0. Use this function in main( ).

Program:

```
#include <stdio.h>
int prime (int n)
{
        for (int i = 2; i < n; i++)
                if(n % i == 0)
                        return 0;
        return 1;
}
void main ( )
{
        int x;
        printf ("Please enter a number:  ");
        scanf ("%d", &x);
        if(prime(x))
                printf ("%d is a Prime Number", x);
        else
                printf ("%d is not a Prime Number",x);
}
```

Programming Time 5.2

Problem:
Write a function which takes a positive number as input and returns the sum of numbers from 0 to that number.

Program:

```
#include<stdio.h>
int digitsSum(int n)
{
        int sum=0;
        for(int i = 0; i <= n; i++)
        {
                sum = sum + i;
        }
        return sum;
}
void main( )
{
        int number;
        printf("Please enter a positive number:  ");
        scanf("%d", &number);
        if(number >= 0)
        {
                int sum = digitsSum(number);
                printf("The sum of numbers upto given number is %d", sum);
```

```
        }
        else
                printf("You entered a negative number.");
}
```

EXERCISE

Q1. Multiple Choice Questions

1. Function could be built-in or _____. (K.B)
A) Admin Defined B) Server Defined C) User Defined D) Both A & C

2. The functions which are available in C Standard Library are called_____. (K.B)
A) User-Defined B) Built-In C) Recursive D) Repetitive

3. The values passed to a function are called_____. (K.B+U.B)
A) Bodies B) Built-In C) Arrays D) Arguments

4. Char cd() {return 'a'}. in this function "char" is_____. (K.B)
A) Body B) Return Type C) Array D) Arguments

5. The advantages of using functions are_____. (K.B)
A) Readability B) Reusability C) Easy Debugging D) All

6. If there are three return statements in the function body, _____ of them will be executed. (K.B+U.B)
A) One B) Two C) Three D) First & Last

7. Readability helps to _____ the code. (K.B)
A) Understand B) Modify C) Debug D) All

8. _____ means to transfer the control to another function. (K.B)
A) Calling B) Defining C) Re-Writing D) Including

ANSWER KEY

1	2	3	4	5	6	7	8
C	B	D	B	D	A	D	A

Q2. Define the following terms. (K.B)

1) Functions.

Ans: A function is a block of statements that performs a particular task, e.g. ***printf*** is function that is used to display anything on computer screen, ***scanf*** is another function that is used to take input from the user. Each program has a main function which performs the tasks programmed by the user. Similarly, we can write other functions and use them multiple times.

2) Built-in functions.

Ans The functions which are available in C standard Library are called built-in Functions. These functions perform commonly used mathematical calculations, string operations, input/ output operations etc. For example, ***printf*** and ***scanf*** are built-in functions.

3) Functions Parameters.

Ans: A function is a block of statement that gets some inputs and provides some output. Inputs of a function are called **parameters** of the function. A function can have multiple parameters, but it cannot return more than one values.

4) Reusability.

Ans: Functions provide reusability of code. It means that whenever we need to use the functionality provided by the function, we just call the functions. We do not need to write

the same set of statements repeatedly.

5) Calling a function.

Ans: We need to call a function, so that it performs the programmed task. Following is the general structure used to make a function call.

Function_name (value1, value2,.., valueN);

Q3. Briefly answer the following questions. (K.B+U.B)

1) What is the difference between arguments and parameters? Give an example.

Ans: The values passed to the function are called **arguments,** whereas variables in the function definition that receive these value are called **parameters** of the function.

2) Enlist the parts of a function definition.

Ans: The function signature does not describe how the function performs the task assigned to it. Function definition does that. A function definition has the following general structure.

Return_type function_name (data_type var1, data_type var2,.., data_type VarN)
{
Body of the function
}

Body of the function is the set of statements which are executed in the function to perform the specified task.

3) Is it necessary to use compatible data types in function definition and function call? Justify your answer with an example.

Ans: Yes, it is necessary to use compatible data types in function definition and function call because parameters pass in function call must have same data type of arguments declared in the function definition otherwise type mismatch error will be occur during compilation time.This is illustrated here through following example:

Example:

```
#include <stdio.h>
void fun (int x, int y)
}
X = 20;
Y = 10;
printf ("Values of x and y in fun (): %d %d", x, y);
}
void main ()
{
int x = 10, y = 20;
fun (x, y);
printf ("Values of x and y in main (): %d %d", x, y);
}
```

Output:

Values of x and y in fun (): 20 10
Values of x and y in main (): 10 20

4) Describe the advantages of using functions.

Ans: Functions provide us several advantages.

1. **<u>Reusability:</u>**

Functions provide reusability of code. It means that whenever we need to use the functionality provided by the function, we just call the functions. We do not need to write the same set of statements again and again.

2. **<u>Separation of tasks:</u>**

Functions allow us to separate the code of one task from the code of other tasks. If we have a problem in one function, then we do not need to check the whole program for removing the problem. We just need to focus at one single function.

3. Handing the complexity of the problem:

If we write the whole program as a single procedure, management of the program becomes difficult. Functions divide the program into smaller units and thus reduce the complexity of the problem.

4. Readability:

Dividing the program into multiple functions, improves the readability of the program.

5) What do you know about the return keyword?

Ans: Return is a keyword that is used to return a value to the calling function. Output of the function is called its **return value**. A function cannot return more than one value. If we try to get more than one value then compiler gives an error. There may be multiple return statement in a function but as soon as the first return statement is executed, the function call returns and further statements in the body of function are not executed.

Q4. Identify the errors in the following code segments. (K.B+U.B+A.B)

Ans:

Program	Error

a) void sum (int a, int b)
{
Return a + b;
}
	A function cannot return more than one value. e.g the following statement results in a compiler error.
	return a+b;

b) void message ();
{
printf ("Hope you are fine:)");
return 23;
}
	Statement terminator cannot be used at the end of function brackets. ( );

c) int max (int a; int b)
{
if (a > b)
return a;
return b;
}
	Error expected in parameterized function syntax (int a;int b) statement terminator cannot be used in brackets.

d) int product (int n1, int n2) return n1* n2;	A function cannot return more than one value. e.g the following statement results in a compiler error. Return n1*n2;

e) int totalDigits(int x)
{
 	
int count =0;
for(int i =x; i>=1, i=i/10)
count++;
return count
};
	• Syntax error expected in for loop structure for(int i=x;i>=1,i/10) comma cannot used at the place of statement terminator.
	• Statement terminator cannot be used at the end of body of function.

Q5. Write down output of the following code segments.

Ans:

Program	Output

a) int xyz (int n)
{
return n + n;
}
	20

int main ()
{
int p = xyx(5);
p = xyz(p);
printf("%d ",p);
}	

b) void abc (int a, int b, int c)
{
int sum = a + b + c;
}
int main()
{
int x = 4, y = 7, z = 23, sum1 = 0;
abc (x, y, z);
printf ("%d %d %d" x, y, z);
}
	4 7 23

c) int aa (int x)
{
int p = x / 10;
x++;
p = p + (p*x);
return p;
}
int main()
{
printf ("we got %d ", aa(aa(23)));
}
	We got 260

d) float f3(int n1, int n2)
{
n1 = n1 + n2;
n2 = n2 – n1;
return 0;
}
int main()
{
printf("%f\n", f3(3, 2));
printf("%f\n", f3(10, 6));
}
	0.000000
	0.000000

PROGRAMMING EXERCISES (A.B)

Exercise 1

Write a function int square(int x); to calculate the square of an integer x.

Solution:

```
#include<stdio.h>
        int square(int x)
        {
         return(x*x);
        }
        int main ( )
        {
                int x,n;
                printf("Input any number for square");
                scanf("%d",&x);
                n=square(x);
                printf("The square of %d is : %d",x,n);
                return 0;
         }
```

Exercise 2

Write a function int power(int x, inty); to calculate and return x^y.

Solution:

```
# include <stdio.h>
#include <math.h>
int power(int a, int b);
int main ( )
{
        int a,b,res;
        printf("Enter a : ");
        scanf ("%d",&a);
        printf("Enter b : ");
        scanf ("%d",&b);
   res=power(a,b);
   printf("Result: %d",res);
}
int power(int a,int b)
{
        int x;
        x=pow(a,b);
        return x;
 }
```

Exercise 3

Write a function to calculate factorial of a number.

Solution:

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

```
int main()
{
        printf("Enter a Number to Find Factorial: ");
        printf("\nFactorial of a Given Number is: %d", fact());
        return 0;
}
int fact()
{
        int i,fact=1,n;
        scanf("%d",&n);
        for(i=1; i<=n; i++)
    {
        fact=fact*i;
    }
        return fact;
 }
```

Exercise 4

Write a function which takes values for three angles of a triangle and prints whether the given values make a valid triangle or not. A valid triangle is the one, where the sum of three angles is equal to 180.

Solution:

```
#include <stdio.h>
int main( )
{
int side1, side2, side3;
printf("Enter the Lengths of Three Sides of a Triangle\n");
scanf("%d %d %d", &side1, &side2, &side3);
if((side1 + side2 > side3)&&(side2 + side3 > side1) &&(side3 + side1 > side2))
{
printf("It is a Valid Triangle\n");
}
else
{
printf("It is an invalid Triangle");
}
return 0;
}
```

Exercise 5

Write a function which takes the amount and the interest percentage and return the interest amount

Solution:

```
#include<stdio.h>
// function declaration
double calculateInterest(double p, double r);
// main function
int main()
{
```

```
                // declare variables
                 double p, r, interest;
                // take input from end-user
                printf("Enter principal amount and rate:");
                scanf("%lf %lf",&p,&r);
                 // calculate interest
                 interest = calculateInterest(p, r);
                 // display result
                 printf("Interest=%lf\n", interest);
                return 0;
}
// function to calculate interest value
double calculateInterest(double p, double r)
{
        return (p*r)/100;
 }
```

Exercise 6

Write a function which takes a number as input and displays its digits with spaces in between.

Solution:

```
#include <stdio.h>
#define MAX 100
// Function to print the digit of number N
void printDigit(int N)
                {
                 // To store the digit of the number N
                 int arr[MAX];
                 int i = 0;
                 int j, r;
                 // Till N becomes 0
                 while (N != 0) {
                 // Extract the last digit of N
                 r = N % 10;
                 // Put the digit in arr[]
                 arr[i] = r;
                 i++;
                 // Update N to N/10 to extract next last digit
                 N = N / 10;
                 }
                 // Print the digit of N by traversing arr[] reverse
                 for (j = i - 1; j > -1; j--)
                 {
                 printf("%d ", arr[j]);
                 }
```

```
}
int main()
{
int N = 3452897;
printDigit(N);
return 0;
}
```

Exercise 7

Write a function to print the table of a number.

```
#include<stdio.h>
int table(int n);
int main()
{
int n=0;
printf("Enter Number: ");
scanf("%d",&n);
table(n);
}
int table(int n)
{
int t=1;
printf("Table of  %d is:\n ",n);
for(int i=1; i<=10; i++)
{
t=n*i;
printf("\n%d x %d=%d",n,i,t);
}
return 0;
}
```

ANSWER KEY

5.1 FUNCTIONS

5.1.1 TYPES OF FUNCTIONS

5.1.2 ADVANTAGES OF FUNCTIONS

5.1.3 SIGNATURE OF A FUNCTION

5.1.4 DEFINING A FUNCTION

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	A	B	B	B	A	C	A	A	D	D	C	A	A	A
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
B	C	B	A	A	A	A	C	C	C	C	D	B	A	A
31	32	33	34	35										
C	B	B	C	A										