Programming and Problem Solving through C Language
O Level / A Level

Chapter - 6 : Functions

A function is a block of statements that performs a specific task.

Types of functions

1) **Predefined standard library functions** – such as puts(), gets(), printf(), scanf() etc – These are the functions which already have a definition in header files (.h files like stdio.h), so we just call them whenever there is a need to use them.
2) **User Defined functions** – The functions that we create in a program are known as user defined functions.

Need functions in C

Functions are used because of following reasons –

1) To improve the readability of code.
2) Improves the reusability of the code, same function can be used in any program rather than writing the same code from scratch.
3) Debugging of the code would be easier if you use functions, as errors are easy to be traced.
4) Reduces the size of the code, duplicate set of statements are replaced by function calls.

Syntax of a function

```
return_type  function_name (argument list)
{
  Local Variable Declaration
   Set of statements – Block of code
}
```

return_type: Return type can be of any data type such as int, double, char, void, short etc. Don’t worry you will understand these terms better once you go through the examples below.

function_name: It can be anything, however it is advised to have a meaningful name for the functions so that it would be easy to understand the purpose of function just by seeing it’s name.

argument list: Argument list contains variables names along with their data types. These arguments are kind of inputs for the function. For example – A function which is used to add two integer variables, will be having two integer argument.

Block of code: Set of C statements, which will be executed whenever a call will be made to the function.

Prototype of a Function

return_typefunction_name(arg type name-1,...,arg type name-n);

- A function prototype provides the compiler with the description of a function that will be defined at a later point in the program.
- **The function prototype** includes a return type indicating the type of variable that the functions will return; It also includes the function name, which should describe what the function does.
- The prototype also contains the variable types of the arguments (arg type) that will be passed to the function.
- Optionally, it can contain the names of the variable that will be passed.
- A prototype should always end with a semicolon.
- **Example** -
    - double squared(double number);
    - void print_report(int report_number);
    - int get_menu_choice(void);

Function Definition

- A function definition is the actual function.
- The definition contains the code that will be executed.
- The first line of a function definition, called the function header, should be identical to the function prototype, with the exception of the semicolon.
- A function header shouldn't end with a semicolon.
- The argument variable names the optional in the prototype, they must be included in the function header.
- Following the header is the function body, containing the statements that the function will perform.
- The function body should start with an opening bracket and end with a closing bracket.
- If the function return type is anything other than void, a return statement should be included, returning a value matching the return type.

```
return_typefunction_name ( arg type name-1,..., arg type name-n)
{
    /* statements; */
}
```

Few Points to Note regarding functions in C:

1) main() in C program is also a function.
2) Each C program must have at least one function, which is main().
3) There is no limit on number of functions; A C program can have any number of functions.
4) A function can call itself and it is known as “**Recursion**“.

Program : This program calculates the sum of 2 number by using function

```
#include<stdio.h>

int sum ( int , int ) ;      // Function Prototype

void main ()                // Main Function from where Execution Begins
{
        int total;
        total=sum(2,3);         // Function Call
        printf("total is %d \n", total);
}

int sum ( int a , int b)      // Function Definition
{
 return (a + b) ;
}
```

Output :- Total is 5

Program : Creating a void user defined function that doesn't return anything

```
#include <stdio.h>

/* function return type is void and it doesn't have parameters*/
void introduction()
{
   printf("Hi\n");
   printf("My name is Chaitanya\n");
   printf("How are you?");
   /* There is no return statement inside this function, since its
    * return type is void
    */
}

void  main()
{
    /*calling function*/
    introduction();
}
```

Program : function returning the max between two numbers

```
#include <stdio.h>

/* function declaration */
int max(int num1, int num2);

void  main () {

   /* local variable definition */
   int a = 100;
   int b = 200;
   int ret;

   /* calling a function to get max value */
   ret = max(a, b);

   printf( "Max value is : %d\n", ret );

}

/* function returning the max between two numbers */
int max(int num1, int num2) {

   /* local variable declaration */
   int result;

   if (num1 > num2)
      result = num1;
   else
      result = num2;

   return result;
}
```

Assignment

1. Write a function to print your name , date of birth and city name.
2. Write a function to print the A B C D E pattern 5 times.
3. Write a function to print the sum of 3 numbers.
4. Write a function to print the area of a rectangle.
5. Write a function to print the area of circle. The function prototype is int area_circle(int);