SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

COURSE NAME : 23ITT101- PROBLEM SOLVING & C PROGRAMMING

I YEAR /I SEMESTER

Unit II – FUNCTIONS AND POINTERS

Topic : FUNCTION

Topics Covered

- **Function**
  - Introduction
  - Advantages
  - Built-in functions
  - Need for user-defined functions
  - Elements of user-defined functions

Modules in C

- Functions
  - Modules in C
  - Programs combine user-defined functions with library functions
    - C standard library has a wide variety of functions
- Function calls
  - Invoking functions
    - Provide function name and arguments (data)
    - Function performs operations or manipulations
    - Function returns results

Modules in C

- A large program in c can be divided to many subprogram
- The subprogram posses a self contain components and have well define purpose. The subprogram is called as a ***function***
- Basically a job of ***function*** is to do something
- C program contain at least one function which is main().

Advantages

- Functions
  - –Modularize a program
  - –All variables declared inside functions are local variables
    - •Known only in function defined
  - –Parameters
    - •Communicate information between functions
    - •Local variables
- Benefits of functions
  - –Divide and conquer
    - •Manageable program development
  - –Software reusability
    - •Use existing functions as building blocks for new programs
    - •Abstraction - hide internal details (library functions)
  - –Avoid code repetition

Built – in Functions

Math library functions

perform common mathematical calculations

#include <math.h>

Format for calling functions

FunctionName(*argument* **);**

If multiple arguments, use comma-separated list

printf("%.2f", sqrt(900.0));

Calls function **sqrt**, which returns the square root of its argument

All math functions return data type **double**

Arguments may be constants, variables, or expressions

Example

```
1:  #include <stdio.h>
2:
3:  long cube(long x);/* Function prototype*/
4:
5:  long input, answer;
6:
7:  int main( void )  8:       {
9:  printf("Enter an integer value: ");
10:             scanf("%d", &input);
11: answer = cube(input);
12: printf("\nThe cube of %ld is %ld.\n", input,
answer);
13:
14: return 0;
15: }
16:
17: long cube(long  /* Function definition*/
x)
18: {
19:   long x_cubed;
20:
21:  x_cubed = x * x *
22:   x;  return
23: }x_cubed;
```

Arguments/formal parameter

Return data type

Actual parameters

- Function names is cube
- Variable that are requires is long
- The variable to be passed on is X(has single arguments)— value can be passed to function so it can perform the specific task. It is called arguments

Output

Enter an integer

value:4 The cube of 4

is 64.

How the function works

- C program doesn't execute the statement in function until the function is called.
- When function is called the program can send the function information in the form of one or more argument.
- When the function is used it is referred to as the ***called function***
- Functions often use data that is passed to them from the ***calling function***
- Data is passed from the calling function to a called function by specifying the variables in a argument list.
- **Argument** list cannot be used to send data. Its only copy data / value / variable that pass from the calling function.
- The called function then performs its operation using the copies.

NEED FOR USER-DEFINED FUNCTIONS

- Every program must have a main function
- It is possible to code any program utilizing only main function, it leads to a number of problems
- The program may become too large and complex and as a result the task of debugging, testing, and maintaining becomes difficult
- If a program is divided into functional parts, then each part may be independently coded and later combined into a single unit
- These subprograms called ‘functions’ are much easier to understand, debug, and test

NEED FOR USER-DEFINED FUNCTIONS

- There are times when some types of operation or calculation is repeated at many points throughout a program
- In such situations, we may repeat the program statements whenever they are needed
- Another approach is to design a function that can be called and used whenever required
- This saves both time and space

NEED FOR USER-DEFINED FUNCTIONS

- This sub-sectioning approach clearly results in a number of advantages
  - It facilitates top-down modular programming.
  - The length of a source program can be reduced by using functions at appropriate places. This factor is particularly critical with microcomputers where memory space is limited

NEED FOR USER-DEFINED FUNCTIONS

–It is easy to locate and isolate a faulty function for further investigations

–A function may be used by many other programs, this means that a C program can build on what others have already done, instead of starting over, from scratch.

A MULTI-FUNCTION PROGRAM

- A function is a self-contained block of code that performs a particular task
- Once a function has been designed and packed, it can be treated as a ‘black box’ that takes some data from the main program and returns a value
- The inner details are invisible to the rest of the program

A MULTI-FUNCTION PROGRAM

Example:

```
main()
{
printline();
printf("This illustrated the use of C functions \n");  printline();
}
printline()
{
int i;
for(i=1; i<40; i++)
pintf("-");
printf("\n");
}
```

ELEMENTS OF USER-DEFINED FUNCTIONS

- Functions are classified as one of the derived data types in C
- Can define functions and use them like any other variables in C programs.
- Similarities between functions and variables in C

– Both function name and variable names are considered identifiers and therefore they must adhere to the rules for identifiers.

– Like variables, functions have types (such as int) associated with them

– Like variables, function names and their types must be declared and defined before they are used in a program

ELEMENTS OF USER-DEFINED FUNCTIONS

- There are **three elements** related to functions
  - Function definition
  - Function call
  - Function declaration
- **The function definition** is an independent program module that is specially written to implement the requirements of the function
- To use this function we need to invoke it at a required place in the program. This is known as **the function call**.
- The program that calls the function is referred to as the **calling program or calling function**.
- The calling program should declare any function that is to be used later in the program. This is known as the **function declaration or function prototype**.

Function prototypes

- Provides the compiler with the description of functions that will be used later in the program
- Its define the function before it been used/called

Function prototypes need to be written at the beginning of the program.

The function prototype must have :

A return type indicating the variable that the function will be return

Function prototypes

Syntax for Function Prototype

return-type function_name(arg-type name-1,...,arg-type name-n);

Function Prototype Examples

- double squared(double number);
- void print_report(int report_number);
- int get_menu_choice(void);

Summary

- A function is a block of code that performs a specific task.
- Both user-defined and standard library functions included in C programming. The standard library functions are built-in functions in C programming.
- These functions are defined in header files.
- Such functions created by the user as per requirement are known as user-defined functions.


Thank
you