C programming language:

Functions in C

Functions in C

OUTLINE

- Review of Functions in C
- Types of Function Calls
  - Call by Value
  - Call by Reference

FUNCTIONS

Modules in C are called functions. A function in C is defined to be the program segment that carries out some specific, well defined task. There are two types of functions:

- Library functions
- Programmer Defined functions

Library Functions

C standard library provides a rich collection of functions for performing I/O operations, mathematical calculations, string manipulation operations etc.

For example, sqrt(x) is a function to calculate the square root of a double number provided by the C standard library and included in the <math.h> header file.

Library Functions

Ex:

:

```
double a=9.9, b;
b = sqrt(a);
printf("The square root of %f is %f", a, b);
```

:

Other functions such as exp(x) (exponential function e^x) and pow(x,y) (x^y) ... can be used as they are needed. Note that each program in C has a function called main which is used as the root function of calling other library functions.

Programmer Defined Functions

In C, the programmers can write their own functions and use them in their programs. Ex:

The following program calls the programmer defined function called *square* to calculate the square of the numbers from 1 to 10.

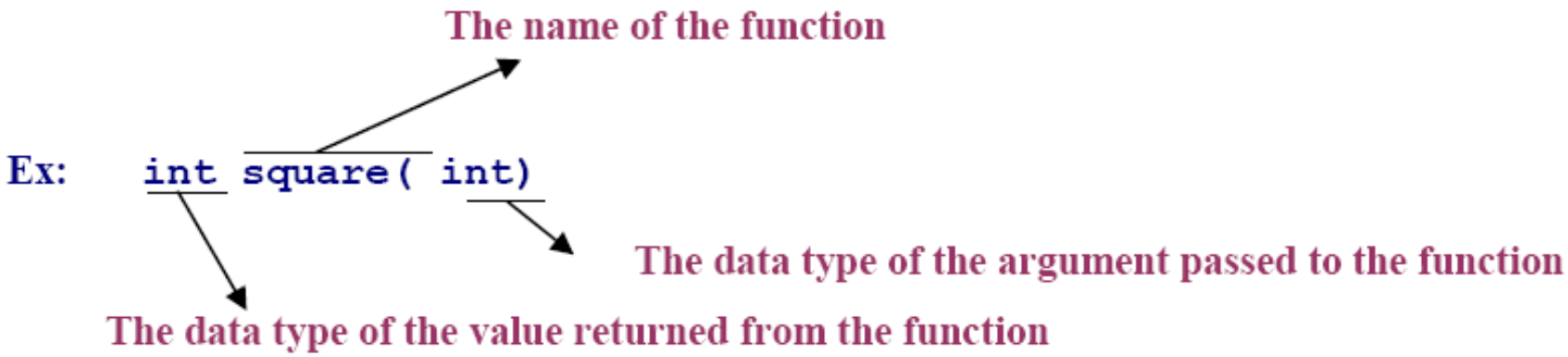
Programmer Defined Functions

```
#include<stdio.h>
int square( int); /* function prototype */
int main()
{
int x;
printf("the squares of numbers from 1 to 10 are:\n");
 for(x=1 ;x <= 10; x++)
{  y = square(x);   /*function call */
printf("the sqare of %d = %d\n",x, y); }
return 0;    }

/*function definition */
int square (int a)
   {
 int b;
   b = a * a;
return  b;
}
```

Function Prototype

- Function prototypes are always declared at the beginning of the program indicating the **name of the function**, the **data type of its arguments** which is passed to the function and the **data type of the returned value** from the function.



Example

- The following program calculates the average of the 3 float numbers entered by the user.
- **#include<stdio.h>**
- **float average(float, float, float); /*function prototype */**
- **int main( )**
- **{**
- **float a, b, c;**
- **printf(“Enter three numbers please\n”);**
- **scanf(“%f”,&a);**
- **scanf(“%f”,&b);**
- **scanf(“%f”,&c);**
- **printf(“the average of 3 numbers = %.3f\n”,average(a,b,c));**
- **return 0;**
- **}**

Example

- **/*function definition */**
- **float average(float x, float y, float z) /*local variables x,y,z */**
- **{**
- **float r;**
- **r = (x+y+z)/3;**
- **return r;**
- **}**

Example

- The following program displays all the integers between two integer numbers.
- **#include<stdio.h>**
- **void printNumbers(int , int); /*function prototype */**
- **int main( )**
- **{**
- **int a, b, c;**
- **printf(“Enter two integers and I will print the all the**
- **numbers in between\n”);**
- **scanf(“%d%d”,&n,&m);**
- **printNumbers(n,m); /*function call */**
- **}**

Example

```
void printNumbers(int x, int y)
{
int i; /*local variable */
if(x<=y)
for(i=x;i<=y;i++)
printf("%d \t",i);
else
for(i=y;i<=x;i++)
printf("%d \t",i);
return; /*optional */
}
```

TYPES OF FUNCTION CALLS

- **Call by Value:**

When a function is called by an argument/parameter which is not a pointer the copy of ***the argument*** is passed to the function. Therefore a possible change on the copy does not change the original value of the argument.

Example

- Write a program to calculate and print the area and the perimeter of a circle. Note that theradius is to be entered by the user. (Use ***Call by value*** approach)
- **#**
- **include<stdio.h> /*The function calls are Call by Value*/**
- **#define pi 3.14**
- **float area(float);**
- **float perimeter(float);**
- **int main( )**
- **{**
- **float r, a, p;**
- **printf(“Enter the radius\n”);**
- **scanf(“%f”,&r);**
- **a = area(r);**
- **p = perimeter(a);**
- **printf(“The area = %.2f, \n The Perimeter = %.2f”, a, p);**
- **return 0;**
- **}**

Example

- **float area(float x)**
- **{**
- **return pi*x*x;**
- **}**
- **float perimeter(float y)**
- **{**
- **return 2.0*pi*y;**
- **}**

TYPES OF FUNCTION CALLS

- **Call by Reference:**

When a function is called by an argument/parameter which is a pointer (address of the argument) the copy of ***the address of the argument*** is passed to the function. Therefore a possible change on the data at the referenced address change the original value of the argument.

Example

- Write a program to calculate and print the area and the perimeter of a circle. Note that the radius is to be entered by the user. (Use ***Call by reference*** approach)

```
#include<stdio.h> /*The function calls is Call by Reference*/
#define pi 3.14
void area_perimeter(float, float *, float *);
int main( )
{
float r, a, p;
printf("Enter the radius\n");
scanf("%f",&r);
area_perimeter(r,&a,&p);
printf("The area = %.2f, \n The Perimeter = %.2f", a, p);
return 0;
}
```

Example

- **void area_perimeter(float x, float *aptr, float *pptr);**
- **{**
- ***aptr = pi*x*x;**
- ***pptr = 2.0*pi*x;**
- **}**