Pointers

DELTA | Pg No. 01
Date / /

⇒ Introduction :-

- Pointers in c language is a variable that stores or points the address of another variable. It is used to allocate memory dynamically.
- The pointer variable might be belonging to any of the data type such as Int, float, char, double, short etc.
- pointer Syntax : data type * var. name ;
  example : int *p ; char *p;
  where * is used to denote that "p" is pointer variable and Not a normal variable.

⇒ Key points to remember about pointers in c :-

- Normal variable stores the value whereas pointer variable stores the adress of the variable.
- The content of c pointer always be a whole Number
- Always c pointer is initilazied to Null. i.e. zero.

- The '&' operator :-
  The '&' Symbol is used to get the address of the variable

- The * Operator :-
  The * Symbol is used to get the value of the variable that the pointer is pointing to.

- Size of any pointer is 2 byte (for 16 bit complier)

Type	Amount of storage
character	1 byte
integer	2 byte
Float	4 byte
long	4 byte
double	8 byte

(*) Example of some valid pointer declarations :-

i) int *p;

ii) char *i, *k;

iii) Float *pt;

iv) int **ptr;

Example iv) means the ptr is a pointer to a location which itself is a pointer to a variable of type integer.

- Example Program for Pointers in C :-

⇒
```
# include <stdio.h>
int main()
{
int *ptr, q;
q = 50;
/* address of q is assigned to ptr */
ptr = &q;
/* display q's value using ptr variable */
Printf ("%d", *ptr);
return 0;
}
```

- Pointers and Arrays

⇒ Pointers and Array are very closley related to each other. In c the name of an array is a pointer that cointains the base adress of the array. Infact it is a pointer to the first element of the array. The array is stored in the contiguous location of the main memory. The array list of integer type, each element of this array occupies two bytes. The name of the array conta the starting adress of the array.

Functions

DELTA | Pg No. 03
Date / /

• Introduction

⇒ A function is a group of statements that together perform a task. Every C program has at least one function, which is 'main()'.

⇒ A function declaration tells the complier about a function's name, return type and parameters. A function definition provides the actual body of the function.

• A function has following four elements :-

⇒ 1) Return type :- A function may return a value. The return type is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return type is the keyword void.

⇒ 2) Function name :- This is the actual name of the function. The function name and the parameter list together constitute the function signature.

⇒ 3) parameters :- A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is reffered to as actual parameter or argument. The parameter list refers to the type, order and NO. of parameter of a function.

⇒ 4) Function body :- function body contains a collection of statements that define what the function does.

- Example :-

```
/* 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;
}
```

- Function Declarations

⇒ A function declaration tells the complier about a function name and how to call the function. The actual body of the function can be defined seperately.

- A function declaration has the following parts :-

⇒ return type function name (parameter list);

for the above defined function max(), the function declaration is as follows -

⇒ int max (int num1, int num2);

parameter names are not important in function declaration only their type is required, so the following is also a valid declaration.

⇒ int max (int, int);

when you define a function in one source file an you

call that function in another file. In such case you should declear the function at the top of the file calling the function.

Calling a function

⇒ A function can be called or invoked from another function by using its name. The function name must be followed by a set of actual parameters, enclosed in paranthesis seperated by commas.

for examples :-

```
#include <stdio.h>
/* function declaration */
int max(int num1, int num2);
int main()
{
   /* local variable definiation */
   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);
   return 0;
}
```

Max value is : 200

DELTA Pg No. 06
Date / /

• PARAMETER PASSING IN FUNCTIONS

⇒ 1) Call by value :-

This meathod copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument.

Example :-

```
/* This function compares two variables and
returns the bigger of the two */
int big (int x, int y)
{
    if (x>y)
        return x;
    else
        return y;
}
```

By defautt, C uses 'call by valve' to pass arguments.

⇒ 2) Call bey Refrence :-

This meathod copies the adress of an argument into the formal parameter. inside the function, the adress is used to acess the actual argument used in the call. This means that changes made to the parameter affect the argument.

Example :-

```
void Exchange (int *val1, int *val2)
{
    int temp;
    temp = *val1
    *val1 = *val2
    *val2 = temp; }
```

ARRAYS

DELTA | Pg No. 07
Date / /

- Introduction

⇒ An array is a collection of data items, all of the same type, accessed using a common name. It has two types :-
1) one-dimensional array and 2) two-dimensional array : A 1d array is like a list and 2D array is like a table. The C languages places no limits on the number of dimensions in an array.

- Initializing Arrays

⇒ Array may be initalized when they are decleared, just as any other variables.
⇒ place the initialization data in curly {} brackets following the equal signs. Note the use of commas.
⇒ An array may be pairtially initalized by providing fewer data items than the size of the array.
⇒ If an array is to be completely initilazied, the dimension of array is not required. the complier will automatically size the array to fit the initalized data.

- Using Arrays

⇒ Elements of an array are accessed by specifing the index of the desired element within square [] brackets after the array name.
⇒ Array subscripts must be of integer type (int, int long, char etc).
⇒ Array indicates start at zero in C and go to one less than the size of array.

DELTA Pg No. 08
Date / /

Sample program using 1-D Array

```
# include <stdio.h>
# include <conio.h>
# include <math.h>
    void main( )
{
  int avg, sum = 0
  int i;
  int marks [30];
  clrscr ( );
  For (i=0; i<=29; i++)
{
  printf (" Enter marks");
  scanf ("%d", &marks [i]);
}
    for (i=0; i<=29; i++)
{
    sum = sum + marks [i];
}
    avg = sum / 30;
    printf (" Avg marks = %d \n", avg);
}
    getch ( );
```

int m ( )	0	1	2	3	
float m [ ]	0.0	0.1	0.2	0.3	a [ ] → 1 D array
char. m [ ]	a	b	c	d	

Sample Program Using 2-D Arrays

```
#include <stdio.h>
    int main (void)
{
   /* Program to add two multidimensional array */
   /* written may 2018 by Kumar Sundaram */
   int a[2][3] = { {5,6,7}, {10,20,30} };
   int b[2][3] = { {1,2,3}, {3,2,1} };
   int sum [2][3], row, column;
      /* first the addition */
   for (row = 0; row < 2; row++)
      for (column = 0; column < 3; column++)
       sum [row] [column] = a [row] [column] +
                            b [row] [column];
   /* Then print the results */
    Printf ("The sum is \n\n");
      For (row=0; row<2; row++)
   {
      for (column = 0; column<3; column++)
      Printf ("\t%d", sum [row] [column]);
      printf ('\n');
   /* at end of each row */
   }
      return 0;
}
```