Introduction to programming language using c

Unit-3 Control Structure and Array

by Shailee Shah
Assistant professor
President Institute of Computer application

3.1 Loop Control Structure

- C language support loop or iteration.
- When we have to perform same task multiple time it requires loop structure.
- For ex: if we want to write “welcome” 10 times,it means same task is repeated 10 times.
- C has 2types of loops:
  1. Entry control loop [while , for]
  2. Exit control loop [do…while]

3.1.1 while loop

- The syntax of while loop

```
while ( condition)
    {
        statement;
    }
```

- 'While'is a keyword.
- When compiler find 'while', it will execute condition. If the condition is correct the statements written within while loop will executed.
- After competition of loop statements control will jump to the while keyword again and the condition will be check again.
- This task will be performed repeatedly till the condition is correct.
- If the condition is incorrect the statement after while loop will be executed.
- In while loop before executing statement of the loop ,condition is evaluated and if it is correct then only loop will be executed.
- Otherwise loop will never executed so it is called entry controlled loop.

/*C Program to demonstrate the working while loop*/

WAP to print “hello world” 10 times.

```
#include <stdio.h>
void main()
{
    int  i=1;
     while(i<=10)
       {
            printf(“hello world \n”);
           i++;
       }
}
```

- I variable that we can identify as control variable.
- It controls the loop.
- I must be initialized by any specific number, otherwise it may contain garbage value and loop may never executed.
- When I become 11 the condition will be false.

3.1.2 Do while loop

- The syntax of do..while loop

```
do
{
        statement;
} while( condition)
```

- 'do While'is a keyword.
- It is exit controlled loop.
- The condition is checked after the execution of the statements of the loop.
- It means that the statements within loop will be executed at least once,without checking the condition.
- In this loop if condition is correct the loop will execute else the loop exits.
- It is also callled post test loops.

/*C Program to demonstrate the working do …while loop*/

WAP to print “hello world” 10 times.

```
#include <stdio.h>
void main()
{
   int  i=1;
      do
      {
            printf(“hello world \n”);
           i++;
      } while(i<=10)
}
```

- I is initialized by 1 and loop will be executed.
- 1st the loop will be executed and then the condition will be checked at last position.

WAP to print to find sum of numbers till the user want to continue.

```
#include <stdio.h>
void main()
{
   int  sum=0,no;
   char ch=‘’;

            do
            {
            printf(“\n enter a number”);
            scanf(“%d”,&no);

            sum=sum+no;

            printf(“DO YOU WANT TO CONTINUE ?\n
            Press Y for yes Press N for no”)
            scanf(“%c”,&ch);

            }whille(ch==‘y’ || ch=‘Y’);

            printf(“\n The sum is %d”, sum);

}
```

3.1.3 for loop

- The syntax of for loop

```
for( initialization ; condition ; increment/decrement)
    {
            statement;
    }
```

- It is entry control loop
- Here initialization ,condition evaluation and alteration of control variable are performed in a single statement.
- It is separated by semicolon[;].
- For loop execution is divided in 3 steps:
  - The initialization of control variable. The initialization is carried out only once.
  - The test expression is checked, if the condition is true the statement in the loop body is executed.
  - The control variable is incremented/decrement . when the condition become false , it exits the loop and the next statement after loop executed.

/*C Program to demonstrate the working for loop*/

WAP to print “hello world” 10 times.

```
#include <stdio.h>
void main()
{
    for(int i=1;i<=10;i++)
    {
        printf(“hello world \n”);
    }
}
```

3.1.4 Nested Loop

- The syntax of nested loop

```
1) for( initialization ; condition ; increment/decrement)
    {
        for( initialization ; condition ; increment/decrement)
        {
            statement;
        }
        statement;
    }

2) do
    {
        statement;
        do
        {
            statement;
        } while( condition)
    } while( condition)
```

3.1.4 Nested Loop

- The syntax of nested loop

```
3) While
    {
            statement;
            do
            {
                    statement;
            } while( condition)
    } while( condition)
```

- In c language we can write loop within loop , it is called nested loop of nesting of a loop.
- In nested loops , if outer loop executed for m times and the inner loop execute for n times for each value of m so the output will generate m *n values.

/*C Program to demonstrate the working nested loop*/

<u>WAP to print table of 1 to 10.</u>

```
#include <stdio.h>
void main()
{
    int i,j,ans;
    for(i=1;i<=10;i++)
    {
        for(j=1;j<=10;j++)
        {

            ans=i*j;
            printf("\n %d * %d = %d",i,j,ans);
        }
    }
}
```

/*C Program to demonstrate the working nested loop*/

WAP to print the pattern.

```
1
1 2
1 2 3
1 2  3 4
```

```
#include <stdio.h>
void main()
{
        int  i,j;
        for(i=1;i<=4;i++)
        {
                for(j=1;j<=i;j++)
                {

                        ans=i*j;
                        printf("%d",j);
                }
         printf("\n");
        }
}
```

i=row and j=column

3.2 other statements

1) break:

- The syntax of break:

```
while(condition)
{
        if(condition)
        {
                break;
        }
}
```

- 'break'is a keyword.
- In loop, if we want to exit the loop instantly without performing the condition testing, break keyword is used.
- When compiler finds break keyword, the loop is terminated and the control passes to the outer loop or next statement after loop.
- Generally break statement is written with certain condition so it is associated with if statement.
- The keyword break is also used in switch statement .

/*C Program to demonstrate the working break */

WAP to read different 50 numbers and print that numbers.If user enter a number that is greater than 100,then terminated the loop.

```
#include <stdio.h>
void main()
{
   int  no,i=1;
        while(i<=50)
        {
                 printf("Enter a number\n");
                 sacnf("%d",&no);

                if(no>100)
                {
                                break;
                }
                else
                {
                                printf("\n The entered number is : %d",no);

                }
 i++;
        }
}
```

2) continue:

- The syntax of *continue* :

```
while(condition)
{
Loop continue      if(condition)
                   {
                              continue;
                   }
}
```

- '*continue*' *is* a keyword.
- The continue statement forces control to be transferred back to a re-evaluation of the condition controlling the loop.
- With the use of continue we can skip the execution of certain statements of loop.
- The continue is written with conditional statements like if.

/*C Program to demonstrate the working *continue* */

WAP to print only even numbers between 1 to 100.

```
#include <stdio.h>
void main()
{
   int  i=1;
            while(i<=100)
            {
                        if(i%2!=0)
                        {
                                    i++;
                                    continue;
                        }

            printf("\t %d",i);
            i++;

            }
}
```

3) Goto :

- The syntax of *continue* :

```
xyz:
if(condition)
{
        goto xyz;
}
```

- '*goto*' *is* a keyword.
- Goto statement is used to transfer the control to the location where a local label is specified by an identifier.
- With goto statement label is used and whenever goto finds the label described in the loop,it will jump on that label from anywhere.
- With goto foraward or backward aby direction jump is possible.
- When the program encounters goto xyz ; it transfers the control to label xyz; and then the statements after the label are executed.

/*C Program to demonstrate the working *continue* */

WAP to read only positive number.

```
#include <stdio.h>
void main()
{
   int  no;
        XYZ:
        printf("\n enter number");
        scanf("%d",&no);
        if(no<0)
                {
                        goto XYZ;
                }

        printf("The positive number is : %d",no);
}
```

4) exit :

- In c language exit() is a function available in stdlib.h header file.
- The exit function terminate the entire program.
- When compiler identifies exit the program will be terminated immediately.
- As an argument of exit function zero or non-zero value can be passed.

3.3 Array

- It is very important data structure and useful feature in c.
- Array is collection of related data types so it is called derived data type.
- Array can be 1-D,2-D and multidimensional.

3.1 Definition and declaration of an array

- If we want to read 15 integer values , we required 15 variables to store the values,if we want to read 150 integer values , we required 150 variables to store the values.
- We required to declare 150 varibale as a1,a2,a3…a150 and we need to remember that variable names.
- The syntax: int a1,a2,a3,a4…..a150;
- We need to scan value for 150 variable like:
- scanf("%d %d……150times..%d",&a1,&a2….,&a150);
- To overcome this issue array can be used.

3.1 Definition and declaration of an array

- Array is a collection of similar kind of data elements stored in adjacent memory location and are referred to by s single array-name.
- It is a data structure storing a group of elements ,all of which are of the same data type.
- All the elements in array share the same name , and different from one another with the help of an index.
- Any variable can be access by index number.
- We have to declare and define array before it can be used.
- The size of array can be declared using subscript operator[]

3.1 Definition and declaration of an array

- <u>Syntax of array declaration:</u>
  - Data-type array_name[constant size];
  - Data type refer to the type of element you want to store like int ,float ,char …etc.
  - The data type of an array applies to all the elements , so an array known as homogeneous data structure.
  - Constant-size is the number of elements you want to store.
  - This must be declared before execution.
  - This means that an array has a fixed size.
  - Ex: int A[10];
  - It defines an array of 10 elements having integer data type.

A[0]	A[1]	A[2]	A[3]	A[4]	…	A[10]
101	102	103	104	104……		110

3.1 Definition and declaration of an array

- As each interger uses two bytes,the contiguous memory allocations are **done** as mention.

3.2 1-D and 2-D Array

- 1-d array:
- The array elements can be either initialized at
  1. Compile time [at the time of declaration]
  2. Run time [using scanf statement or using assignment operator]

3.1 Definition and declaration of an array

- The array elements can be initialized at the time of declaration. The values are assigned to each array elements enclosed within the braces and separated by commas.
- Syntax:

  data type array_name[size]={value1,value2…value n};

  int a[5]={1,2,3,4,5};

1	2	3	4	5
a[0]	a[1]	a[2]	a[3]	a[4]

- If we are initializing the values at the time of declaration , then it is not required to specify the size of the array,
- int a[]={1,2,3,4,5};

2-D Array

- The 1-d array variable an store list of values but sometimes a table of values will have to be stored as an example: Marks of 3 Subjects of 5 Student.
- This table will contain total 15 values,3 each line.
- 5 Rows and 3 columns.
- It declare as:

  data type array_name[row size][column size]

	Column-0	Column-1	Column-2
Row 0	A[0][0]	A[0][1]	A[0][2]
Row 1	A[1][0]	A[1][1]	A[1][2]
Row 2	A[2][0]	A[2][1]	A[2][2]

Initialization of 2-D Array

- Int a[2][2];
- a[0][0]=1
- a[0][1]=2
- a[1][0]=3
- a[1][1]=4

	Column-0	Column-1
Row 0	1	2
Row 1	3	4

- The memory would be assigned as:

1	2	3	4
a[0][0]	a[0][1]	a[1][0]	a[1][1]
1010	1012	1014	1016

Initialization of 2-D Array

- Int a[2][2]={1,2,3,4};
- The order in which the initial values are assigned can be altered by including the groups in{} inside main enclosing brackets , like the following initialization as above:
- Int a[2][3]={{1,2,3} . {4,5,6}};

1	2	3
4	5	6

Examples

```
// write a program to read value in 1-d array and
        print it.
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[4];
        for(i=0;i<=4;i++)
        {
                printf("enter array element");
                scanf("%d",&a[i]);
        }
        for(i=0;i<=4;i++)
        {
                printf("%d",a[i]);
        }
        getch();
}
```

Examples

```
// write a program to create 2-d array and print it.

#include<stdio.h>

#include<conio.h>

void main()

{

        int a[2][2],i,j;

        for(i=0;i<2;i++)

        {

                for(j=0;j<2;j++)

                {

                        printf("enter array element");

                        scanf("%d ",&a[i][j]);

                }

        }

        printf("\n The array elements are : \n");

        for(i=0;i<2;i++)

        {

                for(j=0;j<2;j++)

                {

                        printf("%d",a[i][j]);

                }

                printf("\n");

        }

        getch();

}
```

Examples

```
// write a program to create 2-d array and print it.

#include<stdio.h>

#include<conio.h>

void main()

{

        int a[2][2];

        for(i=0;i<2;i++)

        {

                for(j=0;j<2;j++)

                {

                        printf("enter array element");

                        scanf("%d ",&a[i][j]);

                }

        }

        printf("\n The array elements are : \n");

        for(i=0;i<2;i++)

        {

                for(j=0;j<2;j++)

                {

                        printf("%d",a[i][j]);

                }

                printf("\n");

        }

        getch();

}
```

3.3.4 Multidimensional array

- The N dimensional array called as multidimensional array.
- data type array_name[size1] [size2] [size3] ….. [sizeN];
- Int a[2][3][4];
- It can stores 2*3*4 =24 elements.

Thank you

☺