Nested Loops and Jumping Statements

Gagan Deep

Founder & Director

Rozy Computech Services

Kurukshetra, rozygag@yahoo.com

Visit us at : www.rozyph.com

Nested Loops

- Loops, like if-else statements, can be nested, one within another.
- The inner and outer loops need not be generated by the same type of control structure.
- It is essential, however, that one loop be completely embedded within the other - there can be no overlap.
- Each loop must be controlled by different index.
- Moreover, nested control structure can involve both loops and if-else statement.
- Thus, a loop can be nested within an if-else statement, and an if-else statement can be nested within a loop.
- The nested structure may be complex as necessary, as determined by the program logic.

Tables of 1-10

Program 1-4

```
#include <stdio.h>
void main()
{ int i,j;
for (i=1; i<=10; i++)
      for(j=1; j<=10; j++)
         printf("%d  ", i*j);
```

Prints 1 2 3 ..10 2 4 6 ..20 .....10 20 30 100

```
#include <stdio.h>
void main()
{ int i=1,j;
while(i<=10)
   { for (j=1; j<=10; j++)
        printf(" %d\t", i*j);
     i++; } }
```

```
#include <stdio.h>
void main()
{ int i=1,j=1;
while(i<=10)
  { while(j<=10)
      { printf("  %d", i*j);
        j++;}
    i++;} }
```

```
#include <stdio.h>
void main()
{ int i,j =1;
for (i=1; i<=10; i++)
    do
       { printf("%d\n ", i*j);
       j++; }
     while(j<=10) ; }
```

Tables of 1-10

Program 5

```
#include <stdio.h>
void main()
{ int i,j;
for (i=1; i<=10; i++)
{
      for(j=1; j<=10; j++)
         printf("%d\t", i*j);

    printf(" \n");

}
}
```

```
1   2   3   4   5   6   7   8   9   10
2   4   6   8   10  12  14  16  18  20
3   6   9   12  15  18  21  24  27  30
4   8   12 ..... 36  40
5   10  15 ..... 5   50
6   12  18 .....54  60
7   14  21 .....63  70
8   16  24 .....72  80
9   18  27 .....81  90
10  20  30 ..... 90  100
```

Program 6-7

Check Palindrome Number from 11 to 500

```
#include<stdio.h>
void main()
{int i, r, T, rn=0;
for ( i = 11; i<=500; i++) {
 T=i;
   while (T!=0) {
        r=T%10;     T=T/10 ;
        rn=rn*10+r; }
if (rn==i)
printf("%d is Palindrome Number", i );
else
printf("%d is Non-Palindrome Number", i.);
} }
```

Check Palindrome Number from 11 to 500

```
#include<stdio.h>
void main()
{int n=11,r, T, rn=0;
while(n<=500) {
 T=n;
while (T!=0) {
r=T%10;     T=T/10 ;
rn=rn*10+r; }
if (rn==n)
printf("%d is Palindrome Number", );
} }
```

Jumping Statements

- Jumping statements are also known as Loop Control Statements.
- Jumping statements are of different types
  - break
  - continue
  - goto
  - return
  - exit ()

break Statements

- break statement simply terminates the loop and takes control out of the loop. Here explained break statement for for Loop

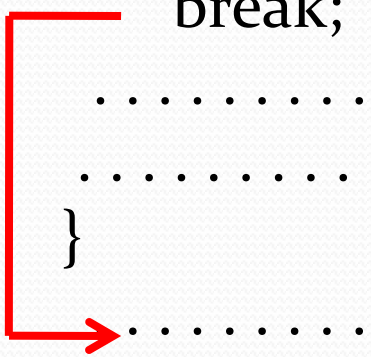
```
for(.....)
  {
     .. . . . . . . .
     .. . . . . . . .
  if (condition)
     break;
     .. . . . . . . .
     .. . . . . . . .
  }
```

8. Print sum of infinite numbers.

```
#include <stdio.h>
void main ()
{ int a, sum=0;
for( ; ; )
{ scanf("%d", &a);
if (a==-999)
break;
sum=sum+a; }
printf("The sum is %d",
sum); }
```

break statement for while Loop

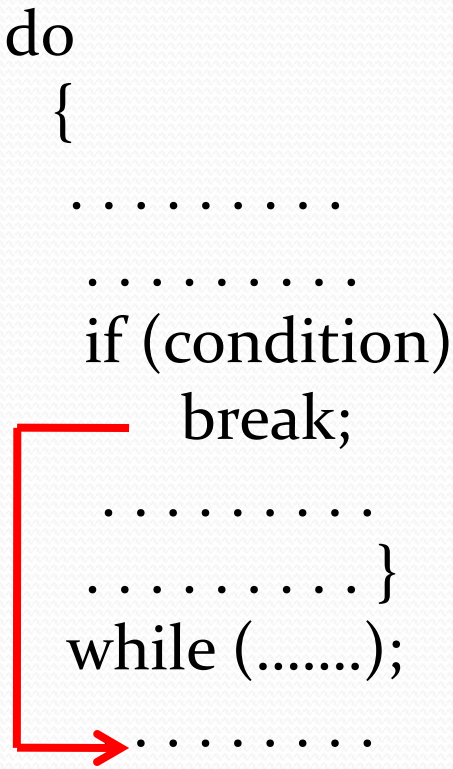
```
while(.....)
  {
   . . . . . . . . .
    . . . . . . . . .
    if (condition)
        break;
     . . . . . . . . .
    . . . . . . . . .
   }
   . . . . . . . . .
    . . . . . . . . .
```



- 9. Print sum of infinite numbers.

```
#include <stdio.h>
void main ()
{ int a, sum=0;
while(1)
{ scanf("%d", &a);
if (a==-999)
break;
sum=sum+a;  }
printf("The sum is %d", sum);  }
```

break statement for while Loop



- 10 Print sum of infinite numbers.

```
#include <stdio.h>
void main ()
{ int a, sum=0;
do
{ scanf("%d", &a);
if (a==-999)
break;
sum=sum+a;  }
while('A');
printf("The sum is %d", sum);  }
```

break Statement in Nested Loop

```
for(.................)
  {
      .. . . . . . . .
      .. . . . . . . .
for(.................)
    {
      .. . . . . . . .
      .. . . . . . . .
    if (condition)
      break;
      .. . . . . . . .
      .. . . . . . . .
     }

   }
```

11 Print Tables of 1 to 10 except 3

```
#include <stdio.h>
void main ()
{
for(int i=1; i<=10; i++)
  {
   for(int j=1; j<=10; j++)
     { if (i==3)
       break;
      printf("%d\t", i*j);
     }
 printf("\n");
 } }
```

Continue Statement

- Continue is used for skipping a part of loop for some condition.
- Continue causes the remaining code inside a loop block to be skipped and causes execution of jump to the top of loop block

```
for(.....)
    {
      .. . . . . . .
      .. . . . . . .
    if (condition)
      continue;
      .. . . . . . .
      .. . . . . . .
    }
```

```
12 Print 1-10 numbers except 3 and 7
#include <stdio.h>
void main ()
 {
 int i;
  for(i=1  ; 1<=10  ; i++)
  {
   if ((i==3)||(i==7))
     continue;
   printf(" %d\t", i);
  }    }
```

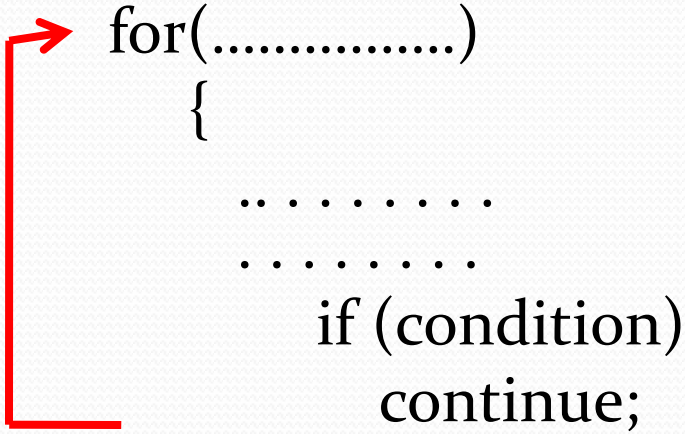
continue statement for while & do-while Loops

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

```
do
  {
  . . . . . .
  . . . . . .
 if (condition)
    continue;
  . . . . . .
  . . . . . . }
 while (.....);
  . . . . . .
 . . . . . .
```

Continue Statement in Nested Loop

```
for(.................)
        {
            .. . . . . . . .
            .. . . . . . . .
          for(.................)
              {
                 .. . . . . . . .
                 . . . . . . . .
                     if (condition)
                         continue;
                 .. . . . . . . .
                 .. . . . . . . .
                 }
        }
```



13 Print Tables of 1 to 10 except 3

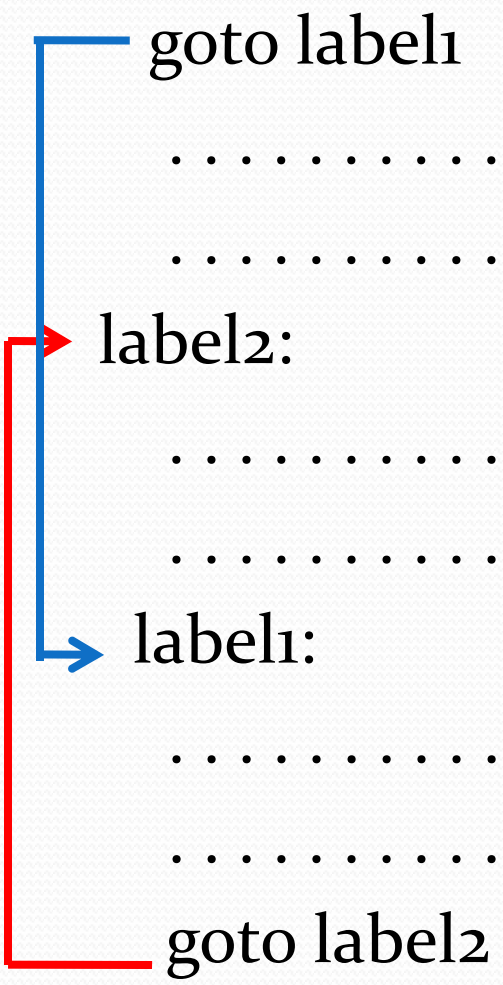
```
#include <stdio.h>
void main ()
    { int i,j;
        for( i=1; i<=10; i++)
         {
            if (i==3)
              continue;
                for(j=1; j<=10; j++)
                  printf("%d\t", i*j);

          printf("\n");
        }
    }
```

goto statement

- The goto statement is used to alter the normal sequence of program execution by transferring control to some other part of the program unconditionally/conditionally.
- In its general form, the goto statement is written as
- goto label;
- where the **label is an identifier** that is used to label the target statement to which the control is transferred.

  label : statement;
- Each labeled statement within the function must have a unique label, i.e., no two statement can have the same label.



return statement Program 14

- **return** is an instruction of the language that returns from a function call.

```
#include <stdio.h>
                    void f()
                      { printf("Executing f\n");
                      return; }
        int main()
       {  f();
       printf("Back from f\n");
       }
```

They will print

Executing f

Back from f

exit() library function

- **exit** is a system call (not a language statement) that terminates the current process.
- In computing, a **system call** is how a program requests a service from an operating **system's** kernel. This may include hardware related services (e.g. accessing the hard disk), creating and executing new processes, and communicating with integral kernel services (like scheduling).
- In the C Programming Language, the Standard Library Functions are divided into several header files.

exit() library function

The following is a list of functions found within the **<stdlib.h>** header file:

Communication with the Environment functions

exit	Exit from Program
abort	Abort Program
getenv	Get Environment String
system	Perform Operating System Command

Integer Arithmetic functions

Pseudo-Random Sequence Generation functions

String Conversion functions

Searching and Sorting functions

Dynamically Allocated Array functions

and so on....

exit() library function Program 15

```
#include <stdio.h>
#include <stdlib.h>
                void f()
               { printf("Executing f\n");
                        exit(0); }
         int main()
        { f();
        printf("Back from f\n");
        }
```

Output will be

Executing f

Print Prime numbers in between 2-100 Prog. 16

```
#include <stdio.h>
 void main ()
 {
    int i, j;
            for(i=2; i<100; i++)
            {
                    for(j=2; j <= (i/j); j++)
                            if(!(i%j))
                            break;
                    // if factor found, not prime
                                    if(j > (i/j))
                                    printf ("%d is prime\n", i);
            }
}
```

Examples of Pyramid Program 17

```
#include <stdio.h>
void main()
{
int i,j,l;
printf("Number of lines: ");
scanf("%d",&l);
   for(i=1;i<=l;++i)
   {
         for(j=1;j<=i;++j)
         printf("* "); // printf("%d ", j);
   printf("\n");
   }
}
```

```
*
**
***
****
*****
```

```
1
12
123
1234
12345
```

Pyramid Program 18

```
#include <stdio.h>
void main()
{ int i,sp,l,k=0;
printf("Number of lines: ");
scanf("%d",&l);
    for(i=1;i<=l;++i)
          { for(sp=1;sp<=l-i;++sp)
                  printf(" ");
     while(k!=2*i-1)
    { printf("* "); //printf("%d", k+1);
          ++k; }
          k=0;
          printf("\n"); }
}
```

```
          *                 1
        * * *             1 2 3
      * * * * *         1 2 3 4 5
    * * * * * * *     1 2 3 4 5 6 7
  * * * * * * * * *  1 2 3 4 5 6 7 8 9
```

Floyd's Triangle Program 19

```
#include<stdio.h>
main()
   {
   int l,i,j,k=0;
   printf("number of lines: ");
   scanf("%d",&l);
         for(i=1;i<=l;i++)
         {
               for(j=1;j<=i;++j)
               printf("%d ",k+j);
         k=k+j;
         printf("\n");
         }
   }
```

```
1
2 3
4 5 6
7 8 9 10
```

Pascal's Traingle Program 20

```
#include<stdio.h>
 void main()
{ int l,coef=1,s,i,j;
 printf("number of lines: ");
 scanf("%d",&l);
 for(i=0;i<l;i++)
        {for(s=1;s<=l-i;s++)
                printf(" ");
        for(j=0;j<=i;j++)
        { if (j==0||i==0)
                coef=1;
        else
        coef=coef*(i-j+1)/j;
        printf("%4d",coef); }
                printf("\n"); } }
```

```
   1
  1 1
 1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

Do Yourself

- Count even and odd digits of a number
- Sum of even and odd digits of a number
- Check for Armstrong Number
- Fibonacci Sequence
- Prime number when divide upto n/2 and sqrt of n

Pyramid

```
#include <stdio.h>
void main()
{
   int i,s,l,k=0,c=0, c1=0;
   printf("number of lines: ");
   scanf("%d",&l);
   for(i=1;i<=l;++i) {
          for(s=1;s<=l-i;++s)
          {
          printf(" ");
          ++c; }
          while(k!=2*i-1)
          { if (c<=l-1)
          { printf("%d ",(i+k));
          ++c; }
                else { ++c1;
                printf("%d ", (i+k-2*c1));
                      }
          ++k; }
   c1=c=k=0;
   printf("\n"); }
   }
```

```
          1
        2 3 2
      3 4 5 4 3
    4 5 6 7 6 5 4
  5 6 7 8 9 8 7 6 5
```

Thanks!

Mail us at : rozygag@yahoo.com

Visit us at : www.rozyph.com