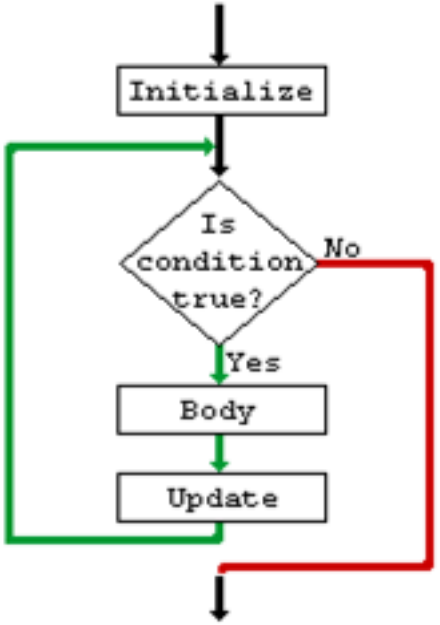
S8_1

Loop Control Structures



Iterative (loop) control structures

- Each loop control structure will have
  - ✓ Program loop: body of loop.
  - ✓ control statement → tests certain conditions & then directs repeated execution of statements within the body of loop.

- Two types: Based on position of control statement.

1) Entry controlled loop: control is tested before the start of the loop. If false, body will not be executed.

2) Exit controlled loop: test is performed at the end of the body. i.e. body of loop executed at least once.

Learning Objectives

- To learn and appreciate the following concepts
  - The `do-while` Statement
  - Nesting of Loops
  - Sample Programs

Learning Outcome

- At the end of session the student will be able to

  - The `do-while` Statement
  - Nesting of loops
  - Write programs

The `do – while` statement

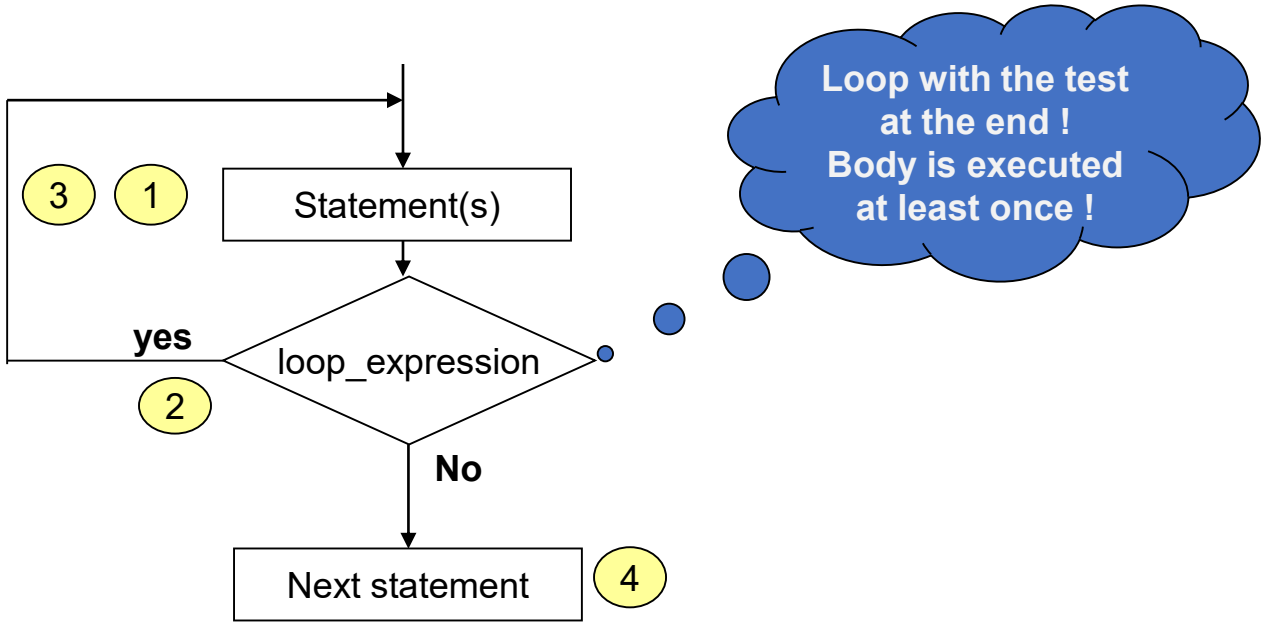
General form:

```
do
{
    body of the loop
}
while(test condition);
```

- ✓**Exit controlled loop**. At the end of the loop, the test condition is evaluated.
- ✓After do statement, program executes the body of the Loop.
- ✓Then, the condition is tested, if it is true, body of the loop is executed once again & this process continues as long as the condition is true.
- ✓**Body of the loop is executed at least once.**
- ✓**do-while loop can be nested.**

The do statement

```
do
        program statement(s)
while ( loop_expression );
```



Finding sum & mean of natural numbers up to N

If N=100 ; 1+2+3+ ... 100

Name: Sum and Mean of natural numbers.

Step 1: Start

Step 2: [Read limit N]
Input N

Step 3: [Set sum equal to 0]
Sum ← 0

Step 4: [Compute sum]
i=1
do
begin
Sum ← Sum + i
i++;
end
while(i<=N)

Step 5: [Compute mean]
Mean ← Sum / N

Step 6: [Print Sum and Mean]
Print 'Sum =',Sum
Print 'Mean =',Mean

Step 7: [End of algorithm]
Stop

```
#include <stdio.h>
int main()
{
  int i, sum, mean;
  printf("Enter limit N: ");
  scanf("%d", &N);
  sum=0; //initialize sum
  i=1;
  do{
    sum= sum + i;
    i = i + 1;
  } while (i <=N);
  mean = sum/N;
  printf("Sum = %d",sum);
  printf("Mean=%d", mean);
    return 0;
}
```

```
Enter limit N: 100
Sum = 5050 Mean=50
```

Program to reverse the digits of a number

```
#include <stdio.h>
int main()
{
    int number, rev=0, right_digit;

    printf("Enter your number.\n");
    scanf("%d", &number);

    do
    {
        right_digit = number % 10;
        rev=rev*10 + right_digit;
        number = number / 10;
    }
    while ( number != 0 );

    printf("The reversed number is %d", rev);
    return 0;
}
```

number = 123
Reverse = 321

```
Enter your number.
12345
The reversed number is 54321
```

Count the number of digits in a given number

```
scanf("%d",&num);
no = num;
do
   {
     rem=num%10;
     num =num/10;
     dcnt++;
    } while(num > 0);

printf(" The no of digits in %d: %d",no, dcnt);
```

e.g.- num = 31467
OUTPUT
5

```
Enter the num:
 42534
The no of digits in 42534: 5
```

Poll Question

- Go to chat box/posts for the link to the Poll question
- Submit your solution in next 2 minutes
- Click the result button to view your score

Count the even and odd digits in a given 'n' digit number

```
scanf("%d",&num);
do
   {
     rem=num%10;
     num =num/10;
     if(rem%2==0)
       ecnt++;
     else
       ocnt++;
    } while(num > 0);

printf("%d even & %d odd digits",ecnt,ocnt);
```

e.g.- num = 31467
OUTPUT
2 even & 3 odd digits

```
Enter the num
 32451
2 even & 3 odd digits
```

Nesting of loop

`do-while` **Lop**

```
i=0;
  do
    {       .....
            ....
             j=0;
      do {
             other statement(s);
             j++;
       } while(j<n);
// end of  inner 'do' statement
       i++;
 } while(i<m) ;
// end of  outer 'do' statement
```

`while` **Loop**

```
i=0;
  while(i<m)
    {       .....
            ....
              j=0;
            while(j<n)
              {
                  other statement (s);
                  j++;
              } // end of  inner 'while'
     i++;
}// end of  outer 'while'
```

Armstrong nos for a given limit 'n'

```
scanf("%d",&lim);
n=1;
do {
   sum = 0;
   num = n;
  do{
      dig = num%10;
      sum = sum+pow(dig,3);
      num = num/10;
    } while(num>0);
    if(sum == n)
      printf("%d\n\t",n);
   n++;
} while(n<lim);
```

Armstrong Number
e.g. - 371
Σ (cubes of digits)= number
$3^3 + 7^3 + 1^3 = 371$

```
Enter the limit:
 500
The armstrong numbers:
 1
 153
 370
 371
 407
```

Convert binary to decimal

$dec = bd*2^n + bd*2^{n-1} + \ldots + bd*2^1 + bd*2^0$

e.g.-given $n=101 \rightarrow 1*2^2 + 0*2^1 + 1*2^0 = 5$

```
int  n, p=0, sum=0, k;
printf("Enter a binary number : ");
scanf("%d",&n);

do {
   k=n%10; // binary number  in n
   sum= sum + k * pow(2,p);//decimal number in sum
   p++;
   n= n/10;
   } while (n!=0);

printf("Decimal Equivalent = %d",sum);
```

Summary

- The `do-while` Statement
- Nesting of Loops
- Programs