Repetition Structure

(CS 1002)

Dr. Muhammad Aleem,

Department of Computer Science,

National University of Computer & Emerging Sciences,
Islamabad Campus

Repetition Structure

- **Repetition Structure** or **Loops**: **Allows** you to **repeat a section** of your **program** a **certain number of times**
- **Repeats** until the **condition remains true**
- **Terminates** when the **condition** becomes **false**

Loops in C++

- **for** loop
- **while** loop
- **do** loop

Counter-controlled loop

Conditional loop

Loops

Counter-controlled Loops

Depends on the **value** of a **variable** known as **counter variable**. The **counter** is **changed** (**increased/decreased**) in **each iteration.**

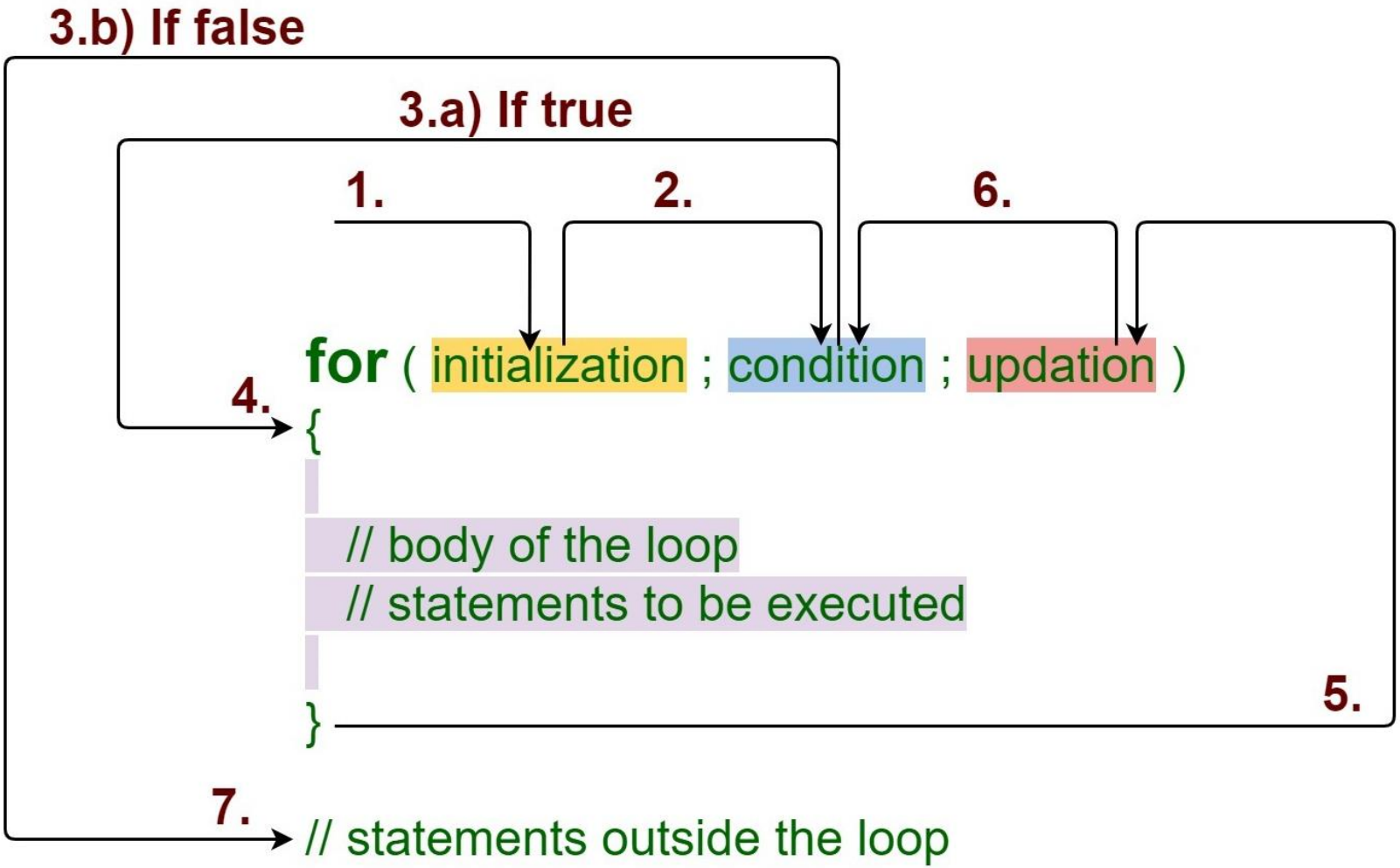
Example: ***for*** **loop**

Conditional loop

A conditional loop keeps repeating until a specific condition is met

Example: ***while*** **and** ***do*** **loops**

for Loop



https://media.geeksforgeeks.org/wp-content/uploads/20191108131134/For-Loop.jpg

for Loop - Example

Initialization expression

Test Condition

Update expression

```
for (int j=0;  j<10;  j++)
        cout << j * j <<endl;
```

(for loop) -- Class Exercise-1

- Get a **number** form **user** and **<u>calculate its factorial</u>**

(for loop) -- Class Exercise-2

Write a program that ask the user to enter a number. The program should print the table of thet number (up to 10 values). Example…

Enter a number: 7

```
7 x 1 =   7
7 x 2 =  14
7 x 3 =  21
7 x 4 =  28
7 x 5 =  35
7 x 6 =  42
7 x 7 =  49
7 x 8 =  56
7 x 9 =  63
7 x 10 = 70
```

(for loop) -- Class Exercise-3

- Write a program that asks the user to enter two numbers (multiple of 10): ***speed1***, and ***speed2*** representing speeds in KPH (Kilo meters per Hour). Then the program should convert and show table of speeds in MPH (Miles per Hour) for all the speed values between ***speed1*** and ***speed2***.

MPH = KPH * 0.6214

speed1 and ***speed2*** variables should be multiple of 10. Each table entry (in KPH) should be updated by 5 in each iteration.

for loop – Multiple Expressions

Multiple Initialization expressions

Test Condition

Multiple Increment/Dec expressions

```
for (int j=0, k=9;  j<10, k>5;  j++,k--)
{
        cout << j * j <<endl;
        cout<< k*k <<endl;
}
```

(1) for loop – Multiple Expressions

```
int i, j;
for(i=1,j=2; i<=3,j<=12; i++,j=j+2)
    cout<<"\n i:"<<i<<", j:"<<j;
```

Output?

```
i:1, j:2
i:2, j:4
i:3, j:6
i:4, j:8
i:5, j:10
i:6, j:12
```

(1) for loop - Variable Visibility

```
int main()
{
      int j;
      for(int j=0; j<10; j++) {
            k = j*j;
            cout<<"\nValue of k: "<<k;
      }
      // j = 23; cannot do this!
      return 0;
}
```

(1) for loop – optional expressions

```
int j=0;
for(; j<10; j++)
        cout<<"\nHello world";
```

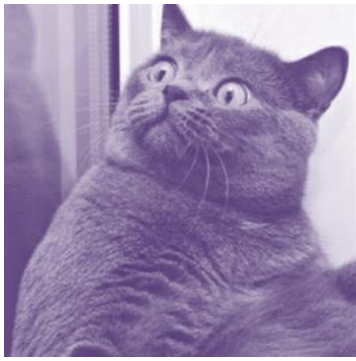
```
int j=0;
for(; j<10;)
{
    cout<<"\nHello world";
    j++;
}
```

```
for(; ;)
  cout<<"\nHello world";
```

Infinite loop
(it never terminates)

for loop

```
int i = 10;
for(cout<<“Starting…”;i;cout<<i<<endl)
--i;
```



Output?

```
starting...
9
8
7
6
5
4
3
2
1
0
```

while loop

while loop

- **for loop** **does** **something** a **fixed number of times**.

- If you **don't know** how many **times** you **want to do something** before you start the loop?

- In this case a different kind of loop may be used: the **while loop**

while loop - syntax

Loop body contain single statement

```
while (n!=0)
    statement;
```

Test expression

Note: no semicolon here

Single-statement loop body

Loop body contain Multiple statement

```
while (v2<45)
    {
    statement;
    statement;
    statement;
    }
```

Test expression

Note: no semicolon here

Multiple-statement loop body

Note: no semicolon here

Example: Tracing a while Loop

Initialize count

```
int count = 0;

while (count < 2)

{

        cout << “\nWelcome to C++!";

        count++;

}
```

Example: Tracing a while Loop

(count < 2) is true

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Example: Tracing a while Loop

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Print “Welcome to C++”

Example: Tracing a while Loop

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Increase count by 1
count is 1 now

Example: Tracing a while Loop

(count < 2) is still true since count is 1

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Example: Tracing a while Loop

```
int count = 0;

while (count < 2)

{

        cout << "Welcome to C++!";

         count++;

}
```

Print “Welcome to C++”

Example: Tracing a while Loop

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Increase count by 1
count is 2 now

Example: Tracing a while Loop

(count < 2) is false since count is 2 now

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

Example: Tracing a while Loop

```
int count = 0;
while (count < 2)
{
        cout << "Welcome to C++!";
        count++;
}
```

The loop exits. Execute the next statement after the loop.

(while loop) – Example

- Write a program that inputs a value in an integer number from user. For this number the program returns the ***count*** for **how many times can we divide this number by 2** ***to get down to 1***".

```
int count = 0; int num;  cin>>num;

//count how many divisions we've done
while (num > 1)
{
      num = num / 2;
      count++;
}
cout<<"\nWe have to divide: "<<count<<" times";
```

(while loop) – Example

infinite while loops…

```
while(true)
{
  cout<<"\n Infinite loop";
}
```

```
while(10)
{
  cout<<"\n Infinite loop";
}
```

```
while(‘A’)
{
  cout<<"\n Infinite loop";
}
```

(while loop) – Example

```
while (numEntries = 3)  //always true
{
   cout <<"working … "; numEntries++;
}
```

```
while (numEntries = 0)  //always false
{
   cout << “never executed… ";
}
```

do loop

do loop

- In **while loop** if **condition is false** it is **never** entered or **executed**

- Sometime, **requirements** are that the **loop should be executed at least once**….

- For that, we use **do loop,** that **guarantees at least on execution of the loop body**

do while loop - Syntax

Loop body contain single statement

```
do
    statement;
while (ch!='n');
```

Note: no semicolon here

Single-statement loop body

Test expression

Note: semicolon

Loop body contain Multiple statement

```
do
    {
    statement;
    statement;
    statement;
    }
while (numb<96);
```

Note: no semicolon here

Multiple-statement loop body

Test expression

Note: semicolon

do loop – Example1

```
int main( )
{
  int counter, howmuch;
  cin>>howmuch;
  counter = 0;
  do {
          counter++;
          cout<<counter<<endl;
  } while ( counter < howmuch);

  return 0;
}
```

do loop – Example2

```
int main( )
{
  int num1, num2; char ch;
  do {
        cout<<"\nEnter a number:";
        cin>>num1;
        cout<<"\nEnter another number:";
        cin>>num2;
        cout<<"\nTheir sum is: "<<num1+num2;
        cout<<"\nDo another time (y/n):";
        cin.get(ch);
  } while(ch=='y');
  return 0;
}
```

break Statement

- **`break`** statement
  - Immediate exit from **`while`**, **`for`**, **`do/while`**, (also used in **`switch`**)
  - **break immediately ends** the **loop** that contains it.

- **Common uses:**
  - **Escape early** from a loop
  - **Skip remainder part of the loop and exit**

break Statement - Examples

```
for (int i=1; i<=5; i++)
{
      if (i==3)
            break;
      cout<<"Hello";
}
```

```
int n;
int EvenSum=0;
while(1)
{
    cin>>n;
    if(n%2==1)
        break;
    EvenSum = EvenSum + n;
}
```

(using break in loops) – Class Exercise 1

- Write a program which reads an integer ***n*** from the user, and prints square value (***n**n***) for that number. Whenever ZERO is entered by the user program should terminate by printing “Invalid Value” message.

continue Statement

- `continue` statement
  - **Only ends the current iteration**
  - **Skips remainder of loop body** (in **current iteration**)
  - **Proceeds** with **next iteration** of **loop**

- **"continue"** can only be inside loops (***for***, ***while***, or ***do-while***). **IT CANNOT BE USED IN *"switch"***

continue Statement - Examples

```
for (int i=1; i<=5; i++)
{
      if (i==3)
            continue;
      cout<<"Hello"<<i;
}
```

```
int n;
  int EvenSum=0;
   while(1)
  {
            cin>>n;
            if(n%2==1)
                  continue;
            EvenSum = EvenSum + n;
  }
```

(Nested Loops)

Nested Repetition Structures

(Nested Loops)

- In a **nested repetition structure**, one loop (**inner loop**) is placed entirely within another loop (**outer loop**)

- In **nested loops** any loop (***for***, ***while***, or ***do*** loop) can be **placed inside another loop** which can be a ***for***, ***while***, or a ***do*** loop

(Nested Loops) - Examples

Outer Loop

Inner Loop

```
for (int i=0; i<2; i++) {
     for (int j=0; j<2;j++) {
          cout<<“\nHello-”<<i<<“:“<<j;
     }
}
```

(Nested Loops) - Examples

```
int main()
{
    int weeks=3, days_in_week=7;

    for (int i = 1; i <= weeks; ++i) {
        cout << "Week: " << i << endl;

        for (int j = 1; j <= days_in_week; ++j) {
            cout << "    Day:" << j << endl;
        }
    }
    return 0;
}
```

(Nested Loops) – Exercise-1

- Write a program to print triangle of starts.

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

(Nested Loops) – Exercise-2

- Write a program to print triangle of starts.

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

(Nested Loops) – Exercise-3

- Write a program to print Rectangle based on two triangles (One using + and other using * symbol).

```
+********
++*******
+++******
++++*****
+++++****
++++++***
+++++++**
++++++++*
+++++++++
```

(Nested Loops) – Exercise-4

- Write a program for a company to calculate total sales for 3 regions. Each region's sales is entered by user which is then summed in total regional sales. The program keep accepting regional sales until it is not 0. The program prints the final regional sum for three regions and exits.

Example Output:

Total Sales for Region 1: 87645

Total Sales for Region 2: 2312

Total Sales for Region 3: 8874

Any Questions!