Programming Language-II

Lecture # 5

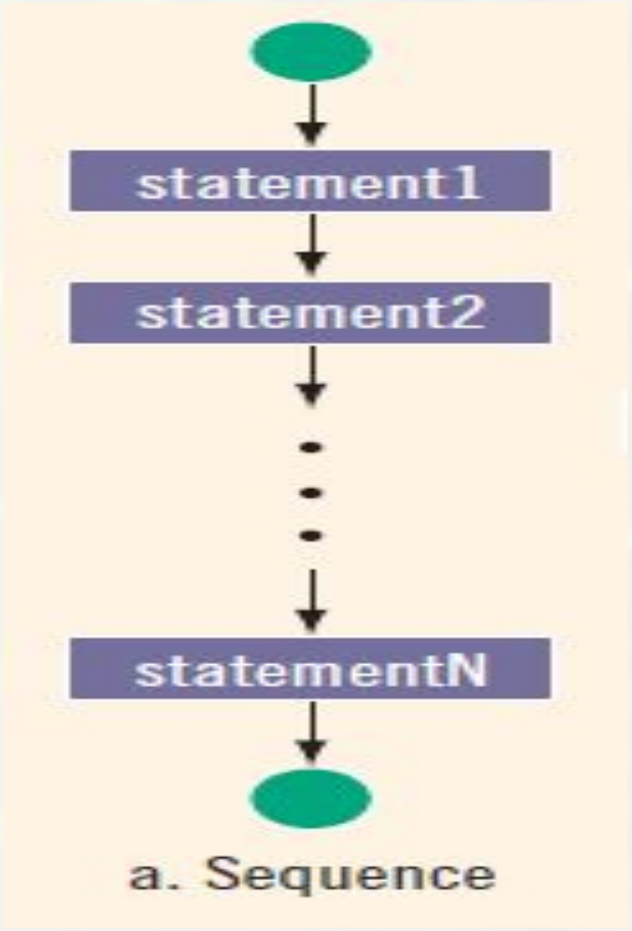
Lecture Content

- Control structures
  - Sequential
  - Selection
  - Repetition
  - Function call

Control structures

- A computer can process a program in one of the following ways: in sequence; selectively, by making a choice, which is also called a branch; repetitively, by executing a statement over and over, using a structure called a loop; or by calling a function.

Program flow(Sequence)



a. Sequence

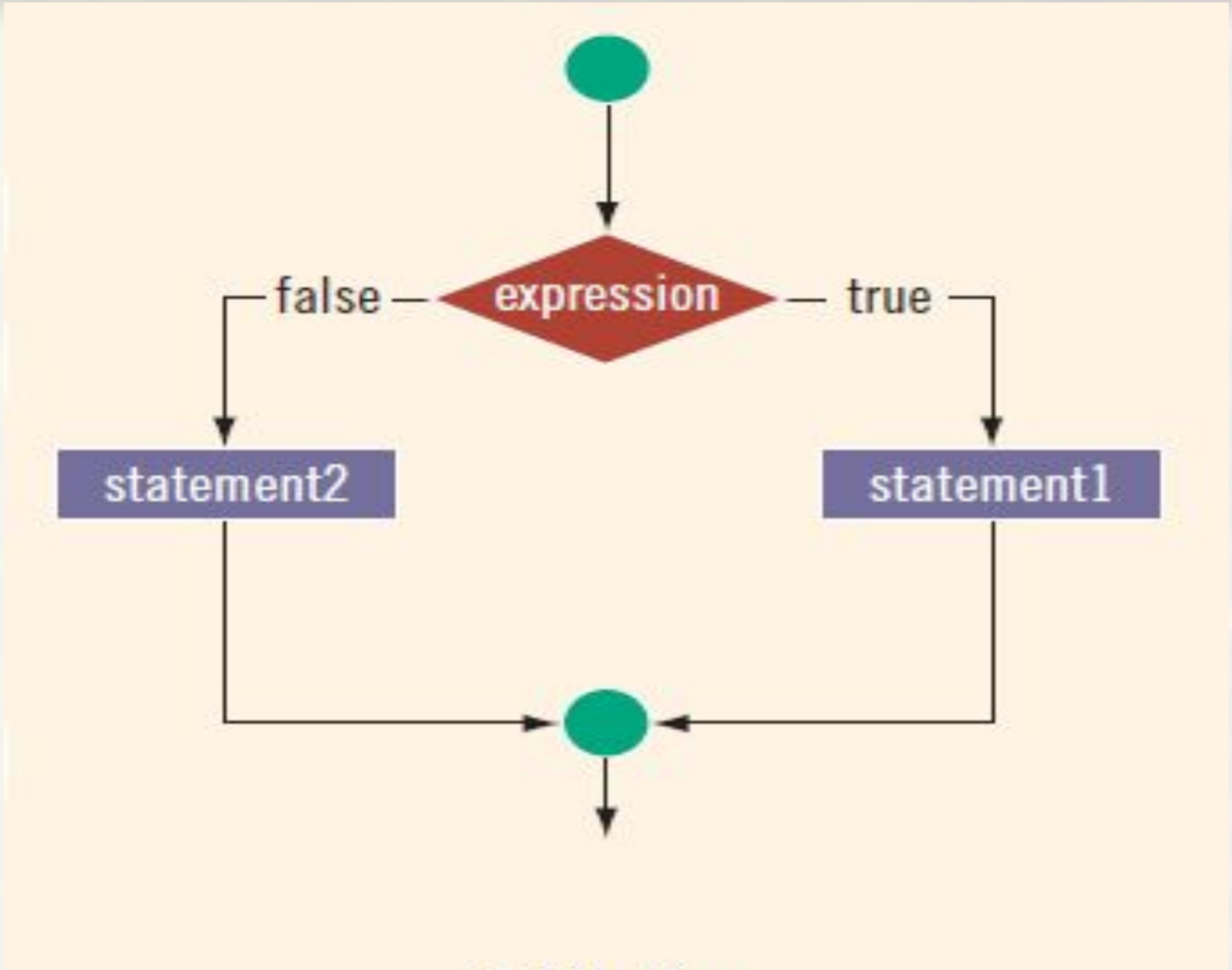
Examples

```
#include <iostream>
using namespace std;
int main()
    {
        int a = 5, b = 10, temp;
        cout << "Before swapping." << endl;
        cout << "a = " << a << ", b = " << b << endl;
        temp = a;
        a = b;
        b = temp;
        cout << "\nAfter swapping." << endl;
        cout << "a = " << a << ", b = " << b << endl;
        return 0;
    }
```

Program flow(Selection)

- A selection structure selects a statement or set of statements to execute on the basis of a **condition**. In this structure, statement or set of statements is executed when a particular condition is true and ignored when the condition is false. There are different types of selection structures in C++. Which are
  - If
  - else if
  - switch

Program flow(Selection)



b. Selection

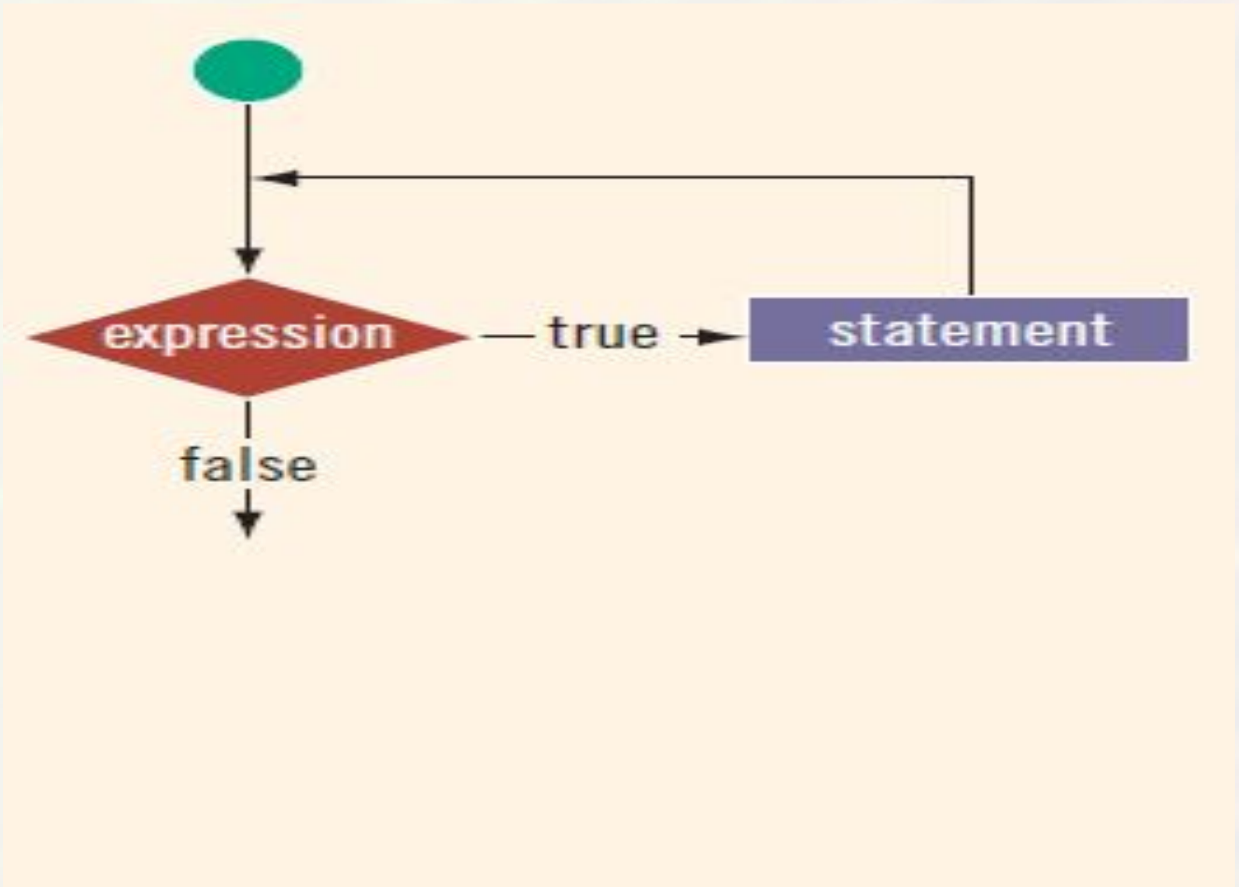
Examples

- Suppose a program inputs the marks of a student and displays a message on screen whether the student is pass or fail. It displays Pass if the student gets 40 or more than 40 marks. It displays Fail when the marks are below 40. The program checks the marks before displaying the message. It is an example of selection structure.

Program flow(Repetition)

- A repetition structure executes a statement or set of statements repeatedly. It is also known as iteration structure or- loop. There are different types of repetition structures in C++. Which are,
    - while
    - do while
    - for

Program flow(Repetition)



c. Repetition

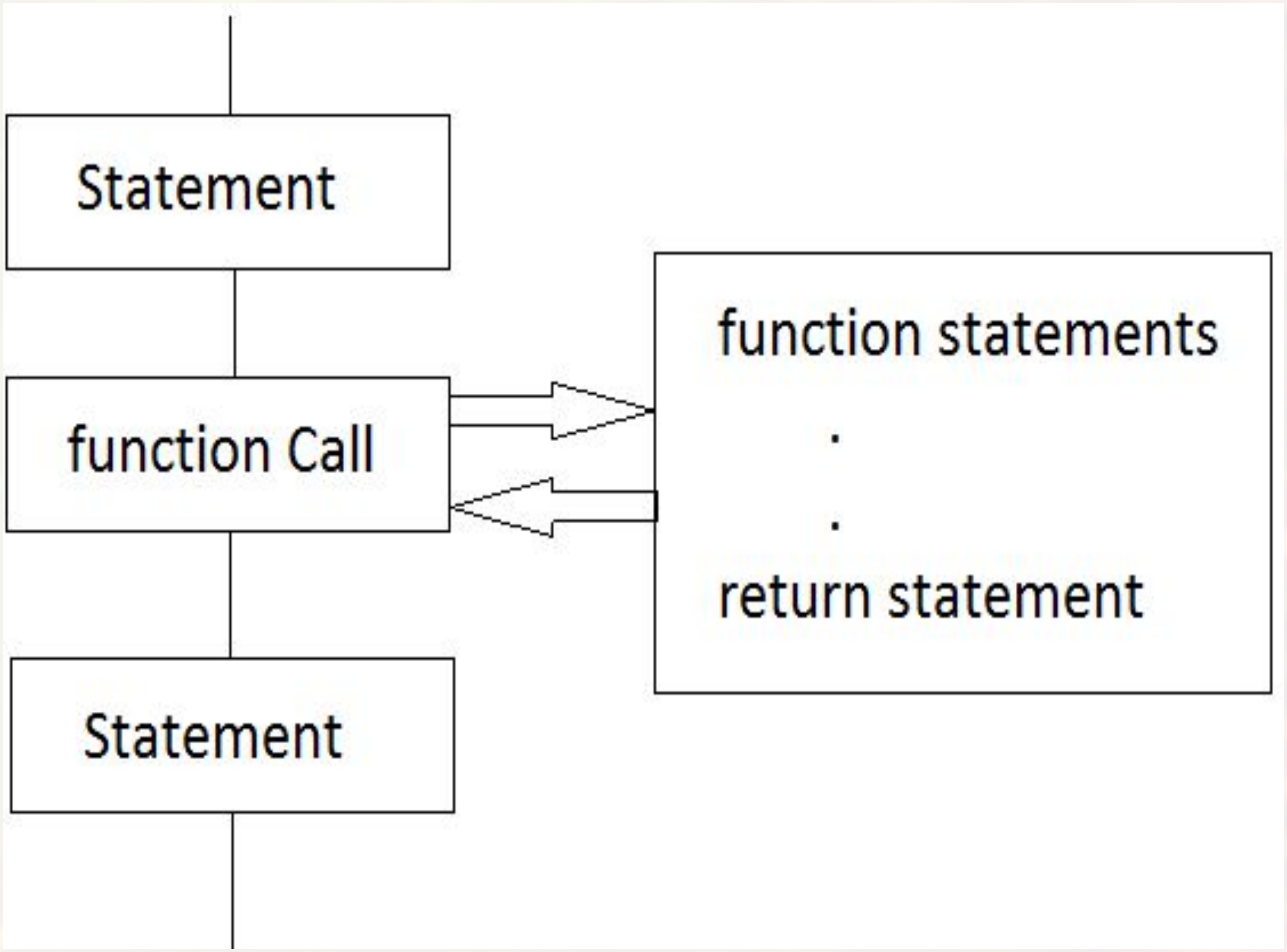
Examples

- Suppose we want to display a message “I love Pakistan" on screen for 100 times. It is time consuming to perform this task using control sequence structure. However, it is very easy to perform this task using repetition structure. A single loop statement can display the message for 100 times.

Program flow(Function call)

- Function call is a type of statement that moves the control to another block of code. The control returns back after executing statements in the block. The remaining statements are executed immediately after the function call when the control is returned.

Program flow(Function call)



Selection Control Structure

If-else statement

- This is the most simple form of the branching statements.
- It takes an expression in parenthesis and an statement or block of statements. if the expression is true then the statement or block of statements gets executed otherwise these statements are skipped.
- **NOTE:** Expression will be assumed to be true if its evaluated values is non-zero.

If-else statement(Syntax)

```
if (expression)
        statement;
OR
if (expression)
       {
        Block of statements;
       }
OR
if (expression)
       {
        Block of statements;
       }
else
       {
               Block of statements;
       }
```

If Example

```
// Program to print positive number entered by the user
// If user enters negative number, it is skipped
#include <iostream>
using namespace std;
int main()
{
    int number;
    cout << "Enter an integer: ";
    cin >> number;
    if ( number > 0)
        cout << "You entered a positive integer: " << number << endl;

    cout << "This statement is always executed.";
    system(“pause”);
    return 0;
}
```

If-else Example

```
// Program to check whether an integer is positive or negative
// This program considers 0 as positive number
#include <iostream>
using namespace std;
int main() {
    int number;
    cout << "Enter an integer: ";
    cin >> number;

    if ( number >= 0)
        cout << "You entered a positive integer: " << number << endl;
    else
        cout << "You entered a negative integer: " << number << endl;

    cout << "This line is always printed.";
    return 0;
}
```

Another If Example

```
#include<iostream>
using namespace std;
int main()
    {
        int marks;
        cout << "Enter marks = ";
        cin >> marks;

        if (marks >= 60)
              cout << "Pass";

        system("pause");
        return 0;
    }
```

Another If-else Example

```
#include<iostream>
using namespace std;
int main()
    {
        int marks;
        cout << "Enter marks = ";
        cin >> marks;

        if (marks >= 60)
              cout << "Pass";
        else
              cout << “Fail";

        system("pause");
        return 0;
    }
```

Nested if

- An if statement within an If statement is called nested if statement. In nested structure, the control enters into the inner if only when the outer condition is true. Only one block of statements are executed and the remaining blocks are skipped automatically.
- The user can use as many if statements inside another if statement as required.
- increase in the level of nesting increases the complexity of nested if statement.

Nested If statement(Syntax)

```
if (expression)
        {
        if (expression)
                {
                Block of statements;
                }
        else
                {
                        Block of statements;
                }
        }
else
        {
                Block of statements;
        }
```

Nested If statement(Example)

```
#include<iostream>
using namespace std;

int main()
      {
      int a, b, c;
      cout << "Enter three number:";
      cin >> a >> b >> c;
      if (a < b)
            if (a < c)
                  cout << a << " is smallest number.";
            else
                  cout << c << " is smallest number.";
      else if (b < c)
            cout << b << " is smallest number.";
      else
            cout << c << " is smallest number.";

system("pause");
return 0;
}
```

Nested If statement(Example)

```
#include<iostream>
using namespace std;
int main(){
    int a, b, c;
    cout << "Enter three number:";
    cin >> a >> b >> c;
    if (a == b)
     if (a == c)
         cout <<" All numbers are equal.";
     else
         cout << "All numbers are not equal.";
    else
     cout<< "All numbers are not equal.";
    system("pause");
    return 0;
}
```

Practice questions

1) Write a C++ program to find maximum between two numbers.
2) Write a C++ program to find maximum between three numbers.
3) Write a C++ program to check whether a number is negative, positive or zero.
4) Write a C++ program to check whether a number is divisible by 5 and 11 or not.
5) Write a C++ program to check whether a number is even or odd.

Practice questions

6) Write a C++ program to check whether a year is leap year or not.
7) Write a C++ program to check whether a character is alphabet or not.
8) Write a C++ program to input any alphabet and check whether it is vowel or consonant.
9) Write a C++ program to input any character and check whether it is alphabet, digit or special character.
10) Write a C++ program to check whether a character is uppercase or lowercase alphabet.

Practice questions

11) Write a C++ program to input all sides of a triangle and check whether triangle is valid or not.

12) Write a C++ program to input all sides of a triangle and check whether triangle is right angle or not.

```
if(a*a==b*b+c*c ||b*b==c*c+a*a || c*c==a*a+b*b)
```

13) Write a C++ program to check whether the triangle is equilateral, isosceles or scalene triangle.

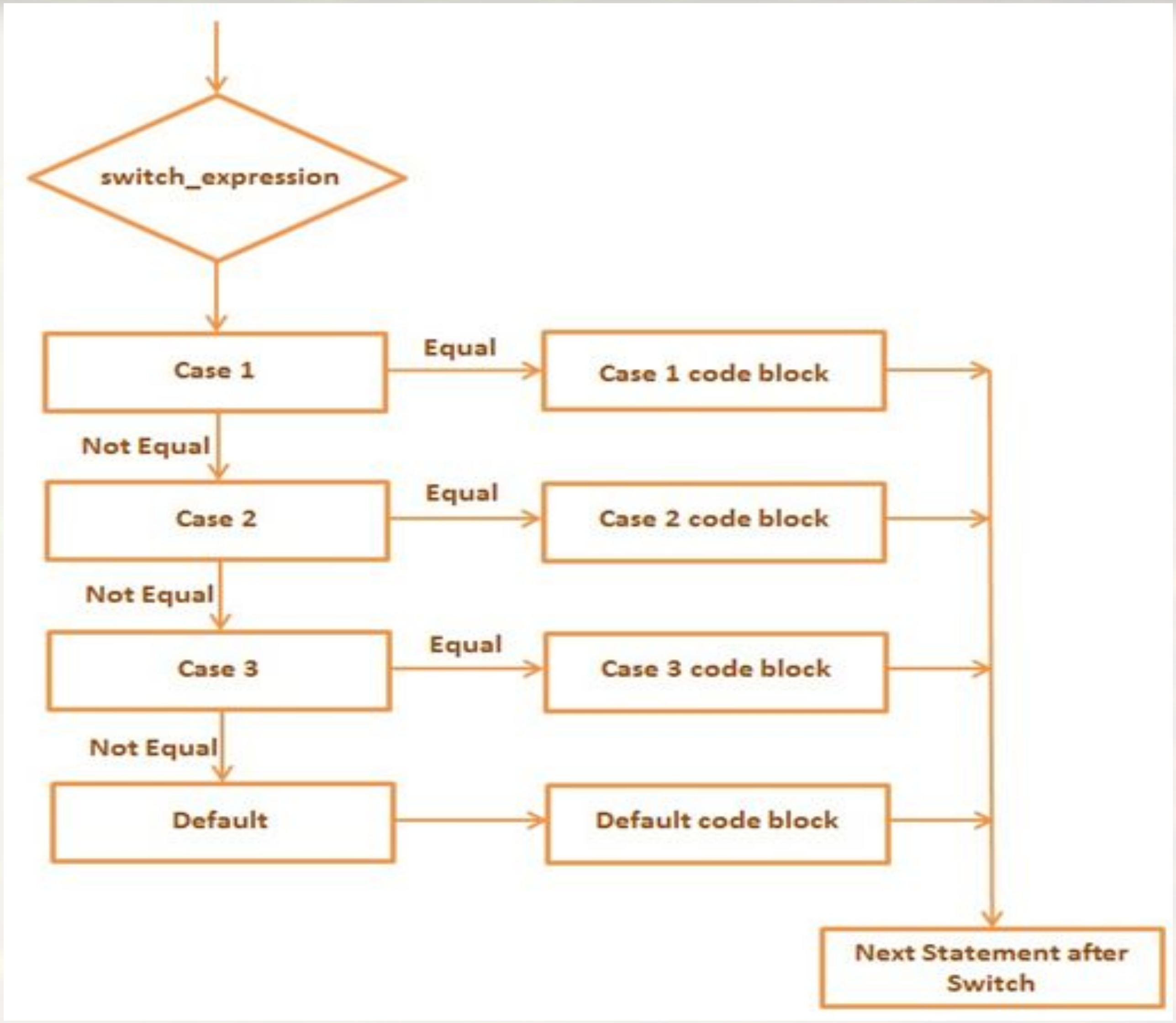
14) Write a C++ program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate percentage and grade according to following:

Percentage >= 90% : Grade A
Percentage >= 80% : Grade B
Percentage >= 70% : Grade C
Percentage >= 60% : Grade D
Percentage >= 40% : Grade E
Percentage < 40% : Grade F

Switch statement

- The switch statement is another conditional structure. It is a good alternative of nested if-else. It can be used easily when there are many choices available and only one should be executed. Nested if becomes very difficult in such situation.

Switch statement flow diagram



Switch statement syntax

```
switch(expression)
    {
            case constant-expression :
             statement(s);
             break;
            case constant-expression :
             statement(s);
             break;
            /* you can have any number of case statements */
            default : /* Optional */
             statement(s);
        }
```

Switch example 1

```
#include <iostream>
using namespace std;
int main ()
{
char grade = 'D';
switch(grade)
    {
            case 'A' :
             cout << "Excellent!" << endl;
             break;
            case 'B' :
            case 'C' :
             cout << "Well done" << endl;
             break;
```

Switch example 1(Cont…)

```
        case 'D' :
                cout << "You passed" << endl;
                break;
        case 'F' :
                cout << "Better try again" << endl;
                break;
        default :
                cout << "Invalid grade" << endl;
        }
cout << "Your grade is " << grade << endl;
return 0;
}
```

Switch example 1 variation

```
#include <iostream>
using namespace std;
int main()
{
char grade=‘D’;
switch (grade)
    {
           case 'A':
           case 'a':
        cout << "Excellent!" << endl;
        break;
           case 'B':
           case 'b':
           case 'C':
           case 'c':
        cout << "Well done" << endl;
        break;
```

Switch example 1 variation(Cont)

```
        case 'D':
        case 'd':
         cout << "You passed" <<endl;
         break;
    case 'F':
    case 'f':
         cout << "Better try again" << endl;
         break;
    default:
         cout << "Invalid grade" << endl;
    }
cout << "Your grade is " << grade << endl;
system("pause");
return 0;
}
```

Switch example 2

```
#include <iostream>
using namespace std;

int main()
{
    char oper;
    float num1, num2;
    cout << "Enter an operator (+, -, *, /): ";
    cin >> oper;
    cout << "Enter two operands: ";
    cin >> num1 >> num2;
    switch (oper)
    {
        case '+':
            cout << num1 << " + " << num2 << " = "          <<
num1 + num2;
            break;
```

Switch example 2(Cont…)

```
    case '-':
        cout << num1 << " - " << num2 << " = " << num1      -
num2;
        break;
    case '*':
        cout << num1 << " * " << num2 << " = " <<
num1*num2;
        break;
    case '/':
        cout << num1 << " / " << num2 << " = " << num1      /
    num2;
        break;
    default:
        cout << "Error! operator is not correct";
        break;}

return 0;}
```

switch statement(Practice questions)

1) Write a program that inputs number of week's day and displays the name of the day. For example if user enters 1, it displays “Monday" and so on.
2) Write a program that inputs a character from the user and checks whether it is a vowel or consonant.
3) Write a program that converts ASCII number to character and vice versa. The program should display the

switch statement(Practice questions)

following menu:

1. Convert ASCII value to Character
2. Convert Character to ASCII value

4) Write C++ code to display no of days in a month.

5) Write a C program to check whether a number is even or odd using switch case.

Home work

- Solve exercise of chapter 4 of D.S Malik book(Page=241 to 245)

Repetition Control Structure

Repetition Control Structure

- Also called loops
- C++ provide following Repetition Control Structure
  - **For loop**
  - **while loop**
  - **do-while loop**
- In addition of that we also need to study the concept of
  - **break statement**
  - **continue statement**

The general format for a for loop

```
for (Initialization_action;
Condition; Condition_update)
{
    statement_list;
}
```

Factorial of n is $n \times (n-1) \times (n-2) \times ... 2 \times 1$

```
int n,f=1;

cin >> n;

for (i=2; i<=n; i++)
{
   f *= i;
}
cout << “The factorial of ” << n << “ is ” << f << “.”;
```

for loop(Practice questions)

1) Write C++ code to print “I love Pakistan” ten time on console.
2) Write a C++ program to print all odd number between 1 to 100.
3) Write a C++ program to find sum of all even numbers between 1 to n.
4) Write a for statement to add all the multiples of 3 between 1 and 100.

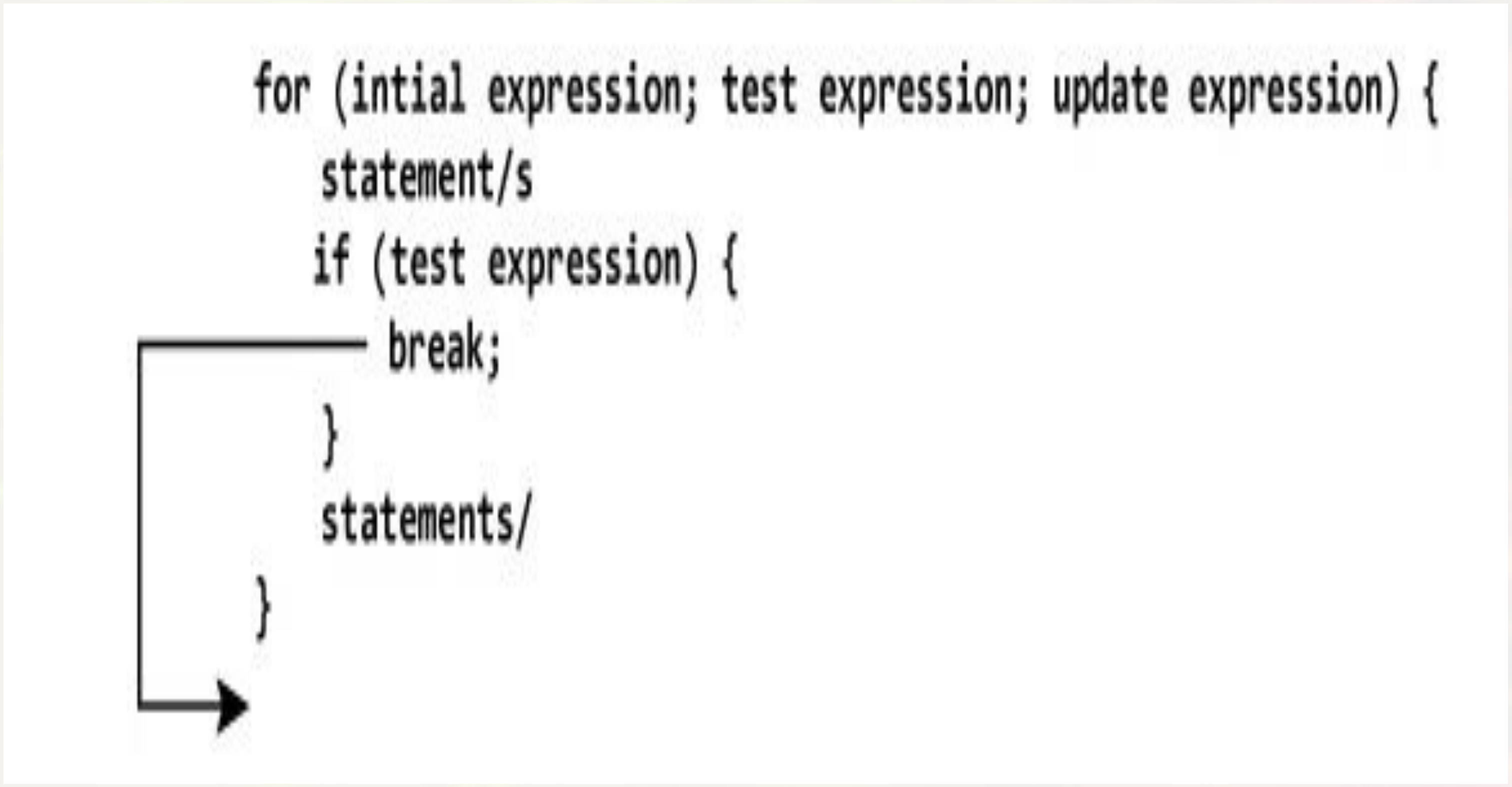
for loop(Practice questions)

5) Write a C++ program to find sum of all even and odd numbers between 1 to n.
6) Write a C++ program to count number of digits in a number.
7) Write a C++ program to enter a number and print its reverse.
8) Write a C program to enter a number and find its palindrome or not.

for loop(Practice questions)

9) Write a program that prompts the user to input an integer and then outputs both the individual digits of the number and the sum of the digits. For example, it should output the individual digits of 3456 as 3 4 5 6, output the individual digits of 8030 as 8 0 3 0, output the individual digits of 2345526 as 2 3 4 5 5 2 6, output the individual digits of 4000 as 4 0 0 0, and output the individual digits of -2345 as 2 3 4 5.

Break statement

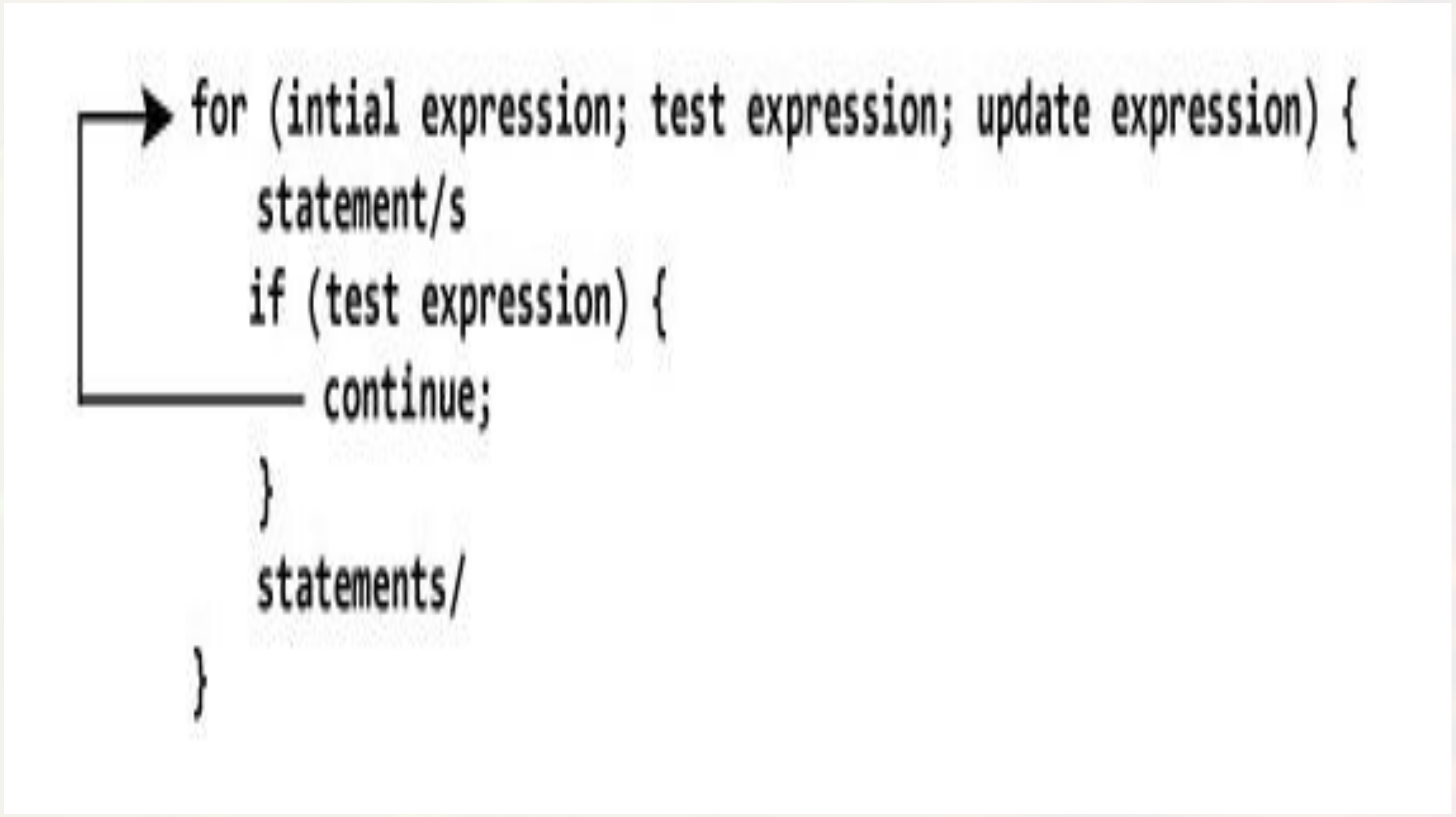


```
for (intial expression; test expression; update expression) {
    statement/s
    if (test expression) {
        break;
    }
    statements/
}
```

Break statement(Example)

```
#include <iostream>
using namespace std;

int main()
    {
    int count = 1;
    for (; ;)
     {
         if (count > 10)
              break;
     cout << "I Love Pakistan"<<endl;
     count++;
     }
    system("pause");
    return 0;
    }
```

Continue statement



```
for (intial expression; test expression; update expression) {
    statement/s
    if (test expression) {
        continue;
    }
    statements/
}
```

Continue statement(Example)

```
#include <iostream>
using namespace std;

int main()
   {
      for (int i = 1; i <= 10; ++i)
      {
          if (i == 6 || i == 9)
              continue;
      cout << i << "\t";
      }

      return 0;
   }
```

Nested for loop(syntax)

```
for ( init; condition; increment )
{
    for ( init; condition; increment )
    {
            statement(s);
    }
   statement(s);
}
```

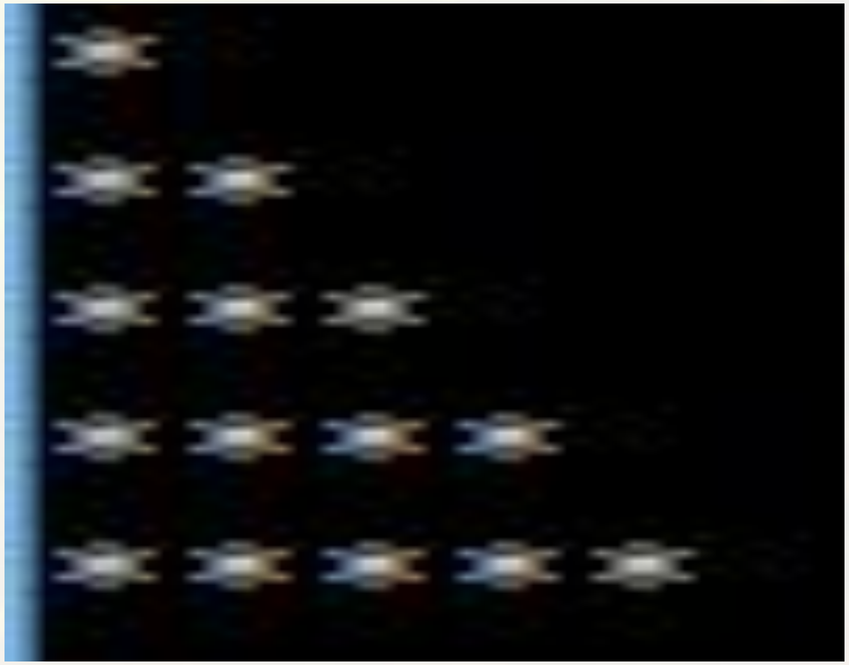
Nested for loop(Example 1)

```
#include <iostream>
using namespace std;

int main()
{
for (int i = 1; i <= 5; i++)
{
    for (int j = 1; j <= 5; j++)
        cout << "* ";
    cout << endl;
}
system("pause");
return 0;
}
```

```cpp
#include<iostream>
using namespace std;
int main()
{
for (int i = 1; i<6; ++i)     // Outer Loop
   {
      for (int j = 1; j <= i; ++j)  // Inner Loop
           cout << "*";
      cout << endl;
   }
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

1) Write C++ code to print following pattern on console using nested for loop.



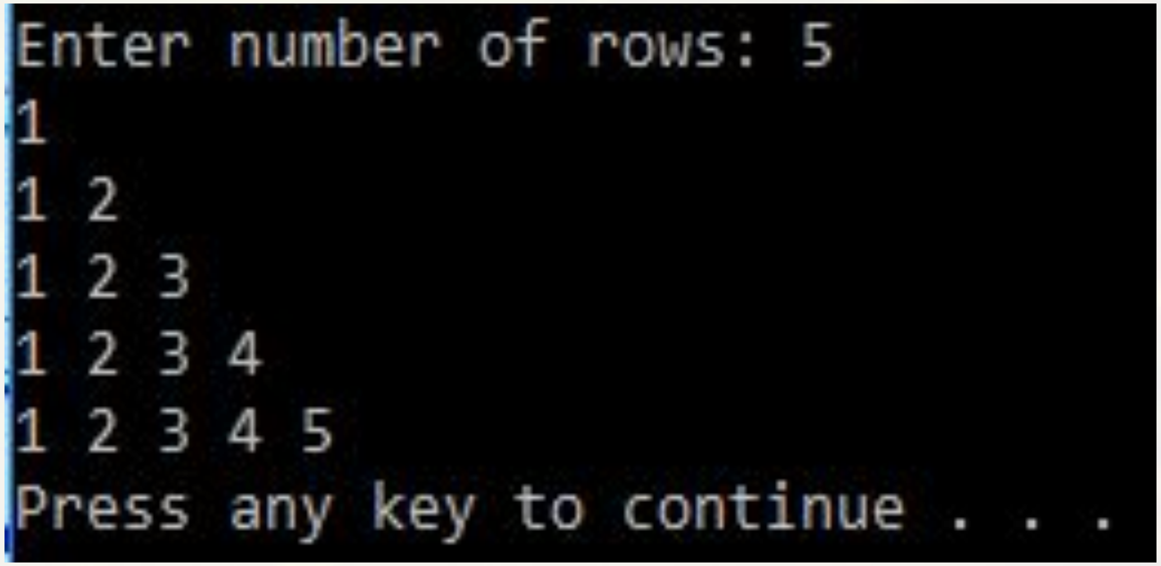
```
*
* *
* * *
* * * *
* * * * *
```

Solution

```
#include<iostream>
using namespace std;
int main()
{
for (int i = 1; i < 6; i++)
{
   for (int j = 1; j <= i; j++)
      cout << "*";
   cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

2) Write C++ code to print following pattern on console using nested for loop.



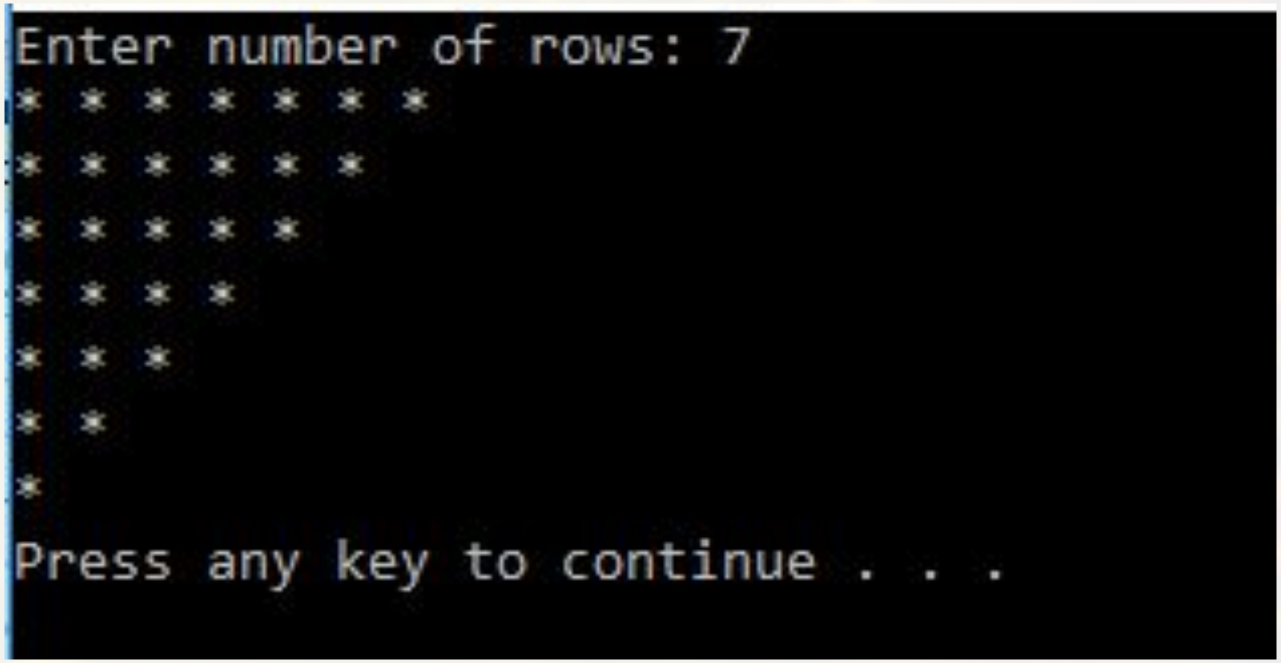
```
Enter number of rows: 5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main()
{
for (int i = 1; i < 6; i++)
{
   for (int j = 1; j <= i; j++)
      cout << j;
   cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

3) Write C++ code to print following pattern on console using nested for loop.



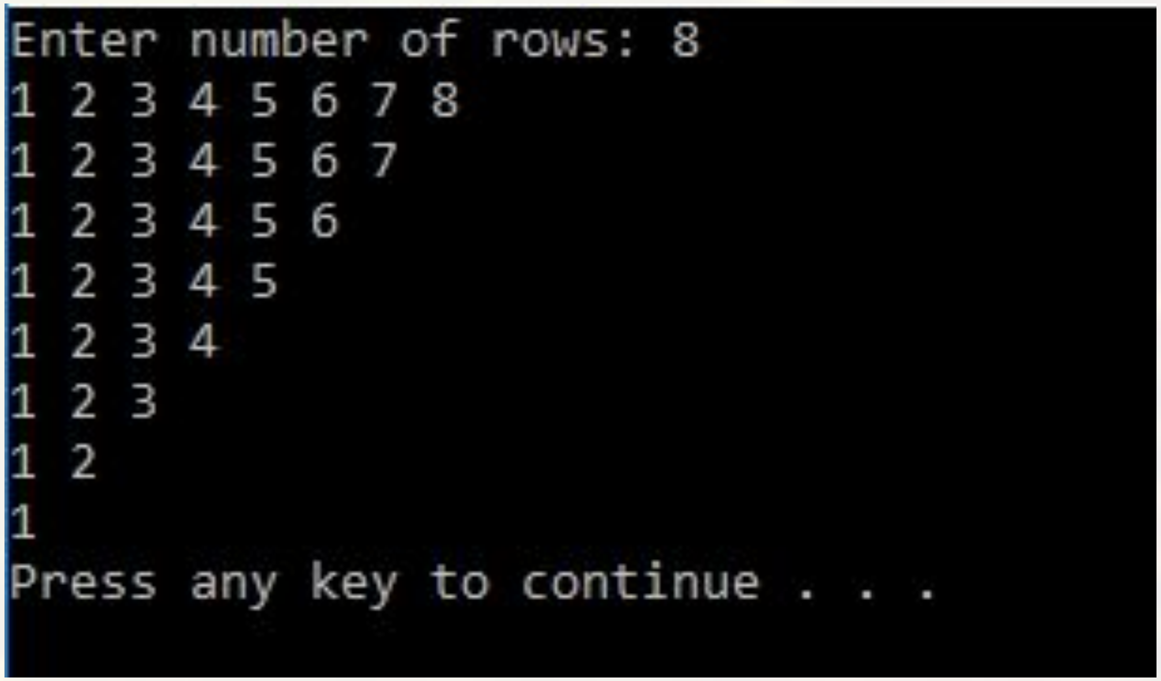
```
Enter number of rows: 7
* * * * * * *

* * * * * *

* * * * *

* * * *

* * *

* *

*

Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main(){
unsigned short int rows;
cout << "Enter no of rows=";
cin >> rows;
for (int i =rows ; i >0; i--){
   for (int j = 1; j <= i; j++)
    cout << "*";
   cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

4) Write C++ code to print following pattern on console using nested for loop.



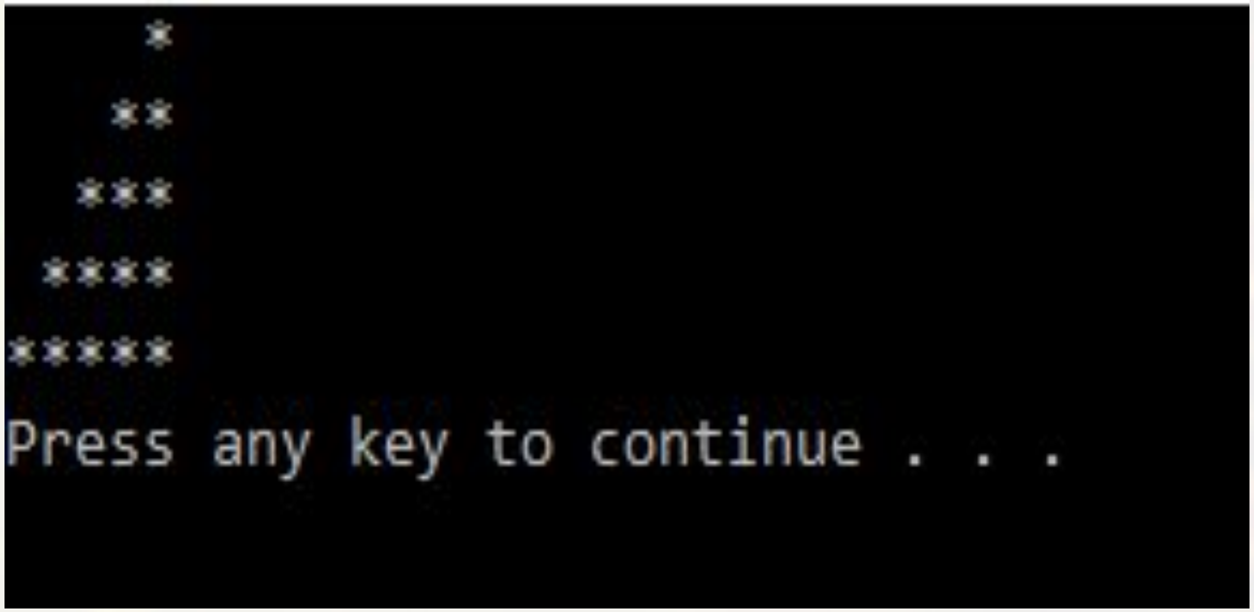
```
Enter number of rows: 8
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main(){
unsigned short int rows;
cout << "Enter no of rows=";
cin >> rows;
for (int i =rows ; i >0; i--){
   for (int j = 1; j <= i; j++)
    cout << j;
   cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

5) Write C++ code to print following pattern on console using nested for loop.



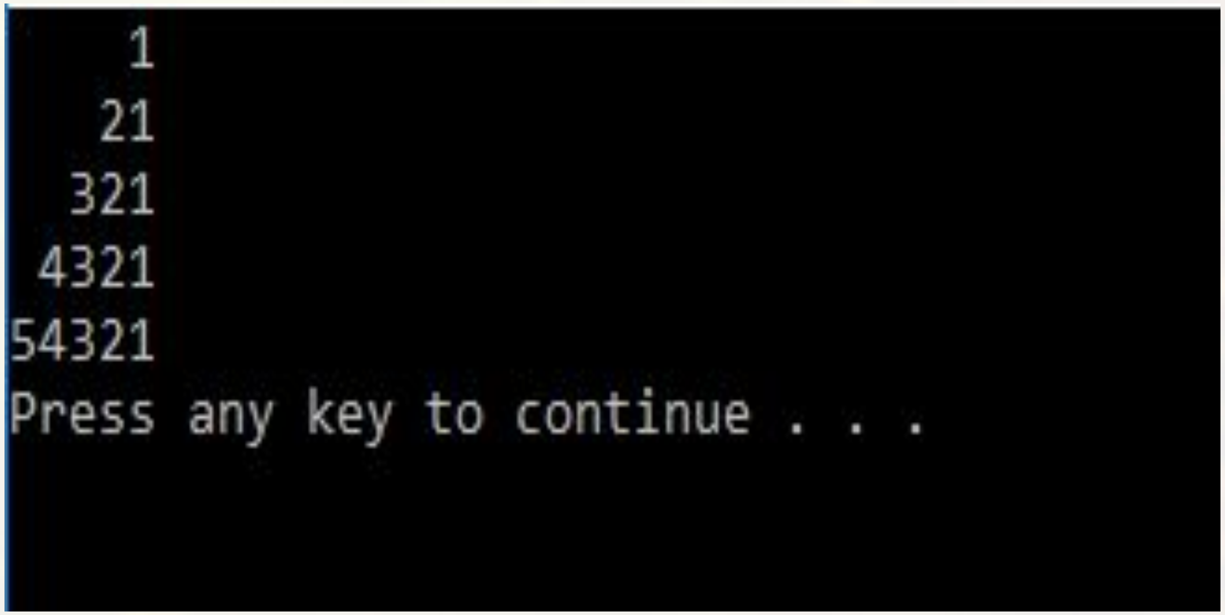
```
    *
   **
  ***
 ****
*****
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main(){
for (int i = 1; i < 6; i++){
   for (int j = 5; j > 0; j--)
      if (i < j)
          cout << " ";
      else
          cout << "*";
      cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

5) Write C++ code to print following pattern on console using nested for loop.



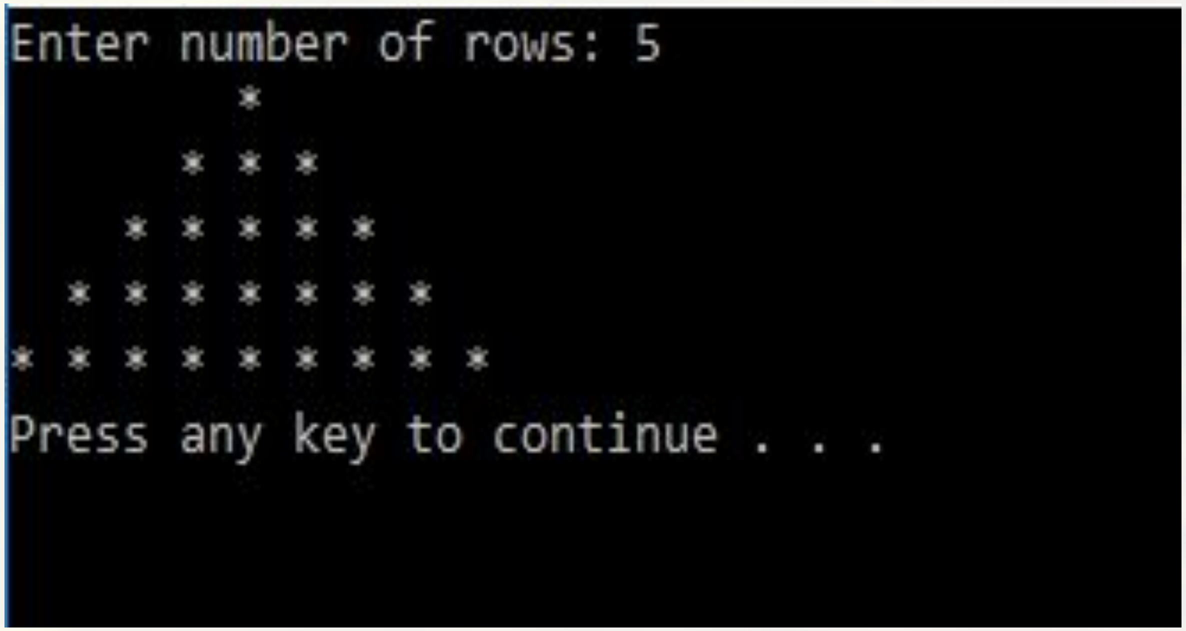
```
    1
   21
  321
 4321
54321
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main(){
for (int i = 1; i < 6; i++){
   for (int j = 5; j > 0; j--)
      if (i < j)
          cout << " ";
      else
          cout << j;
      cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

5) Write C++ code to print following pattern on console using nested for loop.



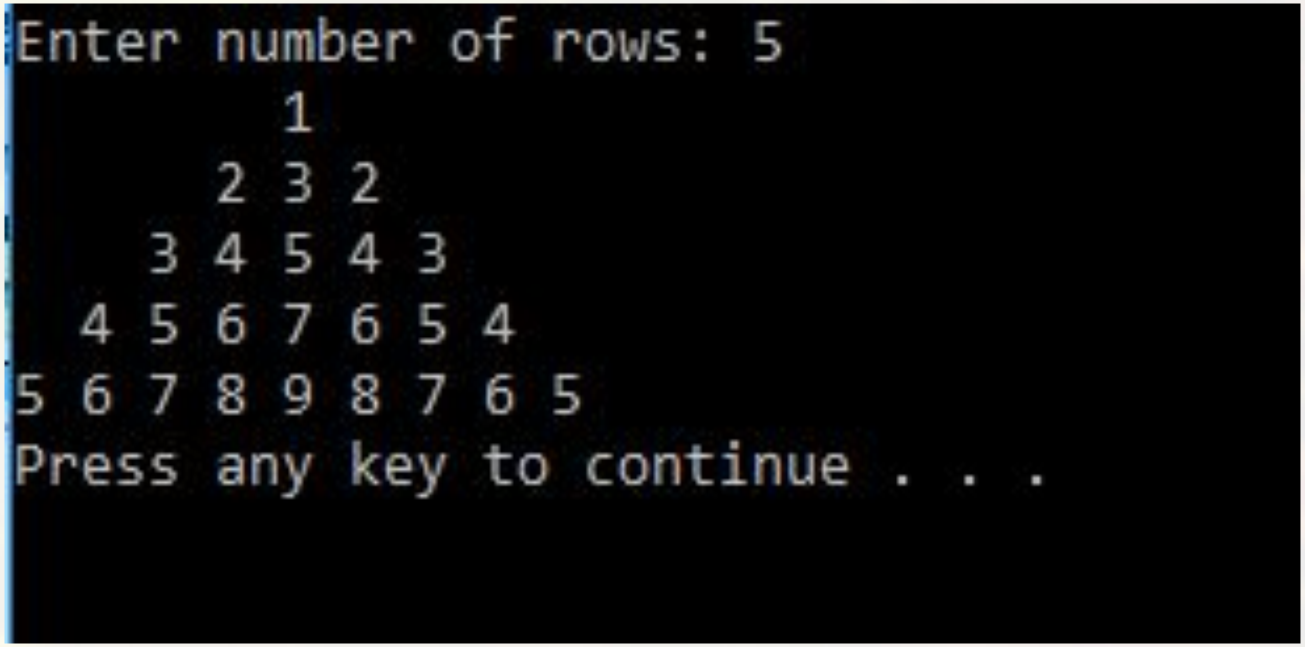
```
Enter number of rows: 5
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main(){
for (int i = 1; i < 6; i++){
   for (int j = 5; j > 0; j--)
      if (i < j)
          cout << " ";
      else
          cout << “* ”;
      cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

6) Write C++ code to print following pattern on console using nested for loop.



```
Enter number of rows: 5
        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
Press any key to continue . . .
```

Solution

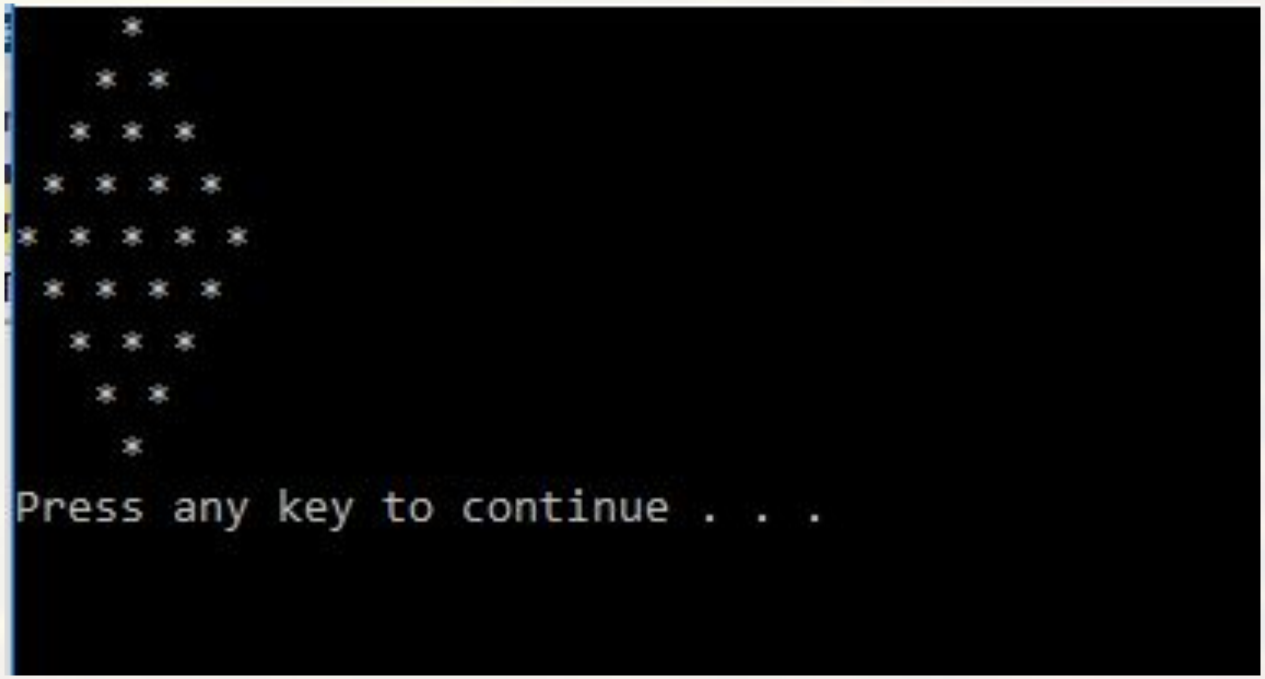
```
#include <iostream>
using namespace std;

int main(){
int rows, count = 0, count1 = 0, k = 0;

cout << "Enter number of rows: ";
cin >> rows;

for (int i = 1; i <= rows; ++i){
     for (int space = 1; space <= rows - i; ++space)
     {
          cout << "  ";
          ++count;
     }

while (k != 2 * i - 1)
{
     if (count <= rows - 1)
     {
          cout << i + k << " ";
          ++count;
     }
```

Solution(Cont…)

```
else
{
        ++count1;
        cout << i + k - 2 * count1 << " ";
}
++k;
}
                count1 = count = k = 0;
                cout << endl;
}
system("pause");
return 0;
}
```

Nested for loop(Practice questions)

6) Write C++ code to print following pattern on console using nested for loop.



```
    *
   * *
  * * *
 * * * *
* * * * *
 * * * *
  * * *
   * *
    *
Press any key to continue . . .
```

Solution

```
#include<iostream>
using namespace std;
int main()
{
	for (int i = 1; i <= 5; i++)
		{
			for (int j = 5; j >= 1; j--)
				if (j > i)
				cout << " ";
				else
				cout << "* ";
			cout << endl;
			}
	for (int i = 1; i <5 ; i++)
		{
			for (int j = 1; j <= 5; j++)
				if (j <= i)
				cout << " ";
				else
				cout << "* ";
			cout << endl;
		}
system("pause");
return 0;
}
```

The general format for a while loop

```
While(Condition)
    Statement;
```

The syntax for compound statements is as follows:

```
While(Condition)
{
    Statement 1;
    Statement 2;
    ............
    Statement N;
}
```

while loop(example 1)

```
#include <iostream>
using namespace std;
int main()
{
int num=1;
while (num <= 5)
    {
        cout << "I Love Pakistan!" << endl;
        num++;
    }
system("pause");
return 0;
}
```

while loop(example 2)

```
#include <iostream>
using namespace std;
int main()
{
int num=1;
while (num <= 10)
    {
        cout << num << endl;
        num++;
    }

system("pause");
return 0;
}
```

while loop(example 3)

Infinite Loop

```
#include <iostream>
using namespace std;
int main()
{
int num=1;
while (1)
   cout << "Infinite while loop!" << endl;

system("pause");
return 0;
}
```

while loop(Practice questions)

1) Write C++ code to print “I love Pakistan” ten time on console.
2) Write a C++ program to print all odd number between 1 to 100.
3) Write a C++ program to find sum of all even numbers between 1 to n.
4) Write a for statement to add all the multiples of 3 between 1 and 100.

while loop(Practice questions)

5) Write a C++ program to find sum of all even and odd numbers between 1 to n.
6) Write a C++ program to count number of digits in a number.
7) Write a C++ program to enter a number and print its reverse.
8) Write a C program to enter a number and find its palindrome or not.

while loop(Practice questions)

9) Write a program that prompts the user to input an integer and then outputs both the individual digits of the number and the sum of the digits. For example, it should output the individual digits of 3456 as 3 4 5 6 and sum of these is 18, and output the individual digits of 2345 as 2 3 4 5 and sum of these is 14.

while loop(Practice questions)

10) Write a program that take input a number and checks whether it is an Armstrong number or not, A number is an Armstrong if the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since 3pow3 + 7pow3 + 1pow3 = 371.

The general format for a do-while loop

```
do{
Statement
}While(Condition);
```

The syntax for compound statements is as follows:

```
do{
   Statement 1;
   Statement 2;
   …………
   Statement N;
} While(Condition);
```

do-while(Example 1)

```
#include <iostream>
using namespace std;

int main() {
// Local variable declaration:
int a = 10;

// do loop execution
do {
cout << "value of a: " << a << endl;
a = a + 1;
} while (a < 20);

return 0;
}
```

do-while(Example 2)

```
#include<iostream>
using namespace std;

#define MAX  10
int main()
{
int sum = 0, num = 1;
do
{
    sum = sum + num;
    num++;
} while (num <= MAX);
cout << "SUM = " << sum;
system("pause");
return 0;
}
```

Comparison between while and do-while loop

BASIS FOR COMPARISON	WHILE	DO-WHILE

General Form	while (condition) {
	statements; //body of loop
	}
		do{
		.
		statements; // body of loop.
		.
		} while(Condition);

Controlling Condition	In 'while' loop the controlling condition appears at the start of the loop.	In 'do-while' loop the controlling condition appears at the end of the loop.
Iterations	The iterations do not occur if, the condition at the first iteration, appears false.	The iteration occurs at least once even if the condition is false at the first iteration.