Logical Operators and their Precedence Order

Review

Single IF Statement

```
if(condition){

}
```

Multiple IF Statement

```
if(condition){

}
if (condition2){

}
```

IF-Else Statement

```
if(condition){

}
else{

}
```

Nested - IF Statement

```
if(condition){

    if(condition){
    }
    else{
    }
}
else{

}
```

Review: Working Example

Write a C++ program that inputs three numbers from the user and prints "Largest" if the first number is largest and prints "Not Largest" otherwise.

Review: Solution

```
1  #include <iostream>
2  using namespace std;
3  main(){
4      int number1, number2, number3;
5      cout << "Enter First Number: ";
6      cin >> number1;
7      cout << "Enter Second Number: ";
8      cin >> number2;
9      cout << "Enter Third Number: ";
10     cin >> number3;
11     if(number1 > number2){
12         if (number1 > number3){
13             cout << "Largest" << endl;}
14         else{
15             cout << "Not Largest" << endl;}
16     }
17     else{
18         cout << "Not Largest" << endl;}
19 }
```

Single IF

Can we give the solution with single IF statement?

Input

number1 = 4
number2 = 5
number3 = 2

Output

Not Largest

```
1  #include <iostream>
2  using namespace std;
3  main(){
4      int number1, number2, number3;
5      cout << "Enter First Number: ";
6      cin >> number1;
7      cout << "Enter Second Number: ";
8      cin >> number2;
9      cout << "Enter Third Number: ";
10     cin >> number3;
11     if(number1 > number2){
12         if (number1 > number3){
13             cout << "Largest" << endl;}
14         else{
15             cout << "Not Largest" << endl;}
16     }
17     else{
18         cout << "Not Largest" << endl;}
19 }
```

Logical Gates

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

Logical Gates: AND Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)
False	False	
False	True	
True	False	
True	True	

Logical Gates: AND Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)
False	False	False
False	True	False
True	False	False
True	True	True

Logical Gates: OR Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)	X OR Y (X V Y)
False	False	False	
False	True	False	
True	False	False	
True	True	True	

Logical Gates: OR Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)	X OR Y (X V Y)
False	False	False	False
False	True	False	True
True	False	False	True
True	True	True	True

Logical Gates: Not Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)	X OR Y (X V Y)	NOT X (~ X)
False	False	False	False	
False	True	False	True	
True	False	False	True	
True	True	True	True	

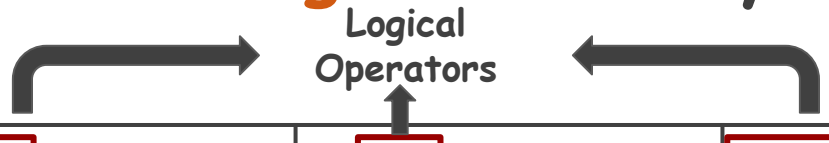
Logical Gates: Not Gate

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.

X	Y	X AND Y (X ^ Y)	X OR Y (X V Y)	NOT X (~ X)
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Logical Gates

Before moving to the solution, lets see the Truth tables of some Logical Gates you have already studied.



X	Y	X AND Y (X ^ Y)	X OR Y (X V Y)	NOT X (~ X)
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Logical Operators:

In C++, we can also use these logical operators.

Operator
AND
OR
Not

Logical Operators:

In C++, we can also use these logical operators.

Operator	In C++
AND	&&
OR	||
Not	!

Logical Operators:

In C++, we can also use these logical operators.

Operator	In C++	Example Suppose x = 3
AND	&&	x < 1 && x < 5
OR	||	x < 1 || x < 5
Not	!	! (x < 1 && x < 5)

Logical Operators:

In C++, we can also use these logical operators.

Operator	In C++	Example Suppose x = 3	Intermediate Result
AND	&&	x < 1 && x < 5	False && True
OR	||	x < 1 || x < 5	False || True
Not	!	! (x < 1 && x < 5)	! (False && True)

Logical Operators:

In C++, we can also use these logical operators.

Operator	In C++	Example Suppose x = 3	Intermediate Result	Final Result
AND	&&	x < 1 && x < 5	False && True	False
OR	||	x < 1 || x < 5	False || True	True
Not	!	! (x < 1 && x < 5)	! (False && True)	True

Single IF

Can we give the solution with single IF statement?

Input

number1 = 4
number2 = 5
number3 = 2

Output

Not Largest

```
1  #include <iostream>
2  using namespace std;
3  main(){
4      int number1, number2, number3;
5      cout << "Enter First Number: ";
6      cin >> number1;
7      cout << "Enter Second Number: ";
8      cin >> number2;
9      cout << "Enter Third Number: ";
10     cin >> number3;
11     if(number1 > number2){
12         if (number1 > number3){
13             cout << "Largest" << endl;}
14         else{
15             cout << "Not Largest" << endl;}
16     }
17     else{
18         cout << "Not Largest" << endl;}
19 }
```

Single IF

Yes, We can!!

Input

number1 = 4
number2 = 5
number3 = 2

Output

Not Largest

```
1   #include <iostream>
2   using namespace std;
3   main(){
4       int number1, number2, number3;
5       cout << "Enter First Number: ";
6       cin >> number1;
7       cout << "Enter Second Number: ";
8       cin >> number2;
9       cout << "Enter Third Number: ";
10      cin >> number3;
11      if(number1 > number2 && number1 > number3){
12              cout << "Largest" << endl;
13      }
14      else{
15          cout << "Not Largest" << endl;
16      }
17  }
```

Working Example

Now, Write a C++ program to give a 10% raise in salary if the experience exceeds 15 years, or the employee's position is AP. The program should take salary, experience, and employee's position as input, and the program should return the updated salary.

Working Example

The user entered the employee position as **AP**, and the experience is greater than **15 years**; therefore, the program gave a raise of 10% in the salary

```
C:\C++>c++ example.cpp -o example.exe

C:\C++>example.exe
Enter the Salary: 50000
Enter the Position: AP
Enter The Experience: 19
Increased Salary: 55000

C:\C++>
```

The user entered the employee position as **Lecturer**, and the experience is less than **15 years**; therefore, the program did not give a raise of 10% in the salary.

```
C:\C++>c++ example.cpp -o example.exe

C:\C++>example.exe
Enter the Salary: 30000
Enter the Position: Lecturer
Enter The Experience: 10
No increase in Salary

C:\C++>
```

Solution

```
1   #include <iostream>
2   using namespace std;
3   main(){
4       float salary, experience;
5       string position;
6       float newSalary;
7       cout << "Enter the Salary: ";
8       cin >> salary;
9       cout << "Enter the Position: ";
10      cin >> position;
11      cout << "Enter The Experience: ";
12      cin >> experience;
13      if(experience > 15 || position == "AP"){
14          newSalary = salary + (salary * 10/100);
15          cout << "Increased Salary: " << newSalary << endl;
16      }
17      else{
18          cout << "No increase in Salary" << endl;
19      }
20  }
```

Working Example

Write a C++ program to give a 10% raise in the salary if the employee's position is not AP. The program should take the salary, the experience and the employee's position as input. The program should return the updated salary if the employee position is not AP; otherwise, there will not be any raise in the salary.

Working Example

the user entered the employee position as AP; therefore, the program did not give a 10% raise in salary.

```
C:\C++>c++ example.cpp -o example.exe

C:\C++>example.exe
Enter the Salary: 60000
Enter the Position: AP
Enter the Experience: 15
No increase in Salary

C:\C++>
```

The user entered the employee position as **Lecturer**, and the experience is less than **15 years**; therefore, the program did not give a raise of 10% in the salary.

```
C:\C++>c++ example.cpp -o example.exe

C:\C++>example.exe
Enter the Salary: 60000
Enter the Position: LE
Enter the Experience: 15
Increased Salary: 66000

C:\C++>
```

Solution

```
1   #include <iostream>
2   using namespace std;
3   main(){
4       float salary, experience;
5       string position;
6       float newSalary;
7       cout << "Enter the Salary: ";
8       cin >> salary;
9       cout << "Enter the Position: ";
10      cin >> position;
11      cout << "Enter the Experience: ";
12      cin >> experience;
13      if(!(position == "AP")){
14          newSalary = salary + (salary * 10/100);
15          cout << "Increased Salary: " << newSalary << endl;
16      }
17      else{
18          cout << "No increase in Salary" << endl;
19      }
20  }
```

What will be the Output?

```
1   #include <iostream>
2   using namespace std;
3
4   main(){
5       int money = 0;
6       string meal = "fruit";
7                       1       1       0       0       0
8       if(meal == "fruit" || meal == "sandwich" && money >= 2){
9           cout << "Lunch being delivered" << endl;
10      }
11      else{
12          cout << "Cannot deliver Lunch" << endl;
13      }
    }
```

Logical Operators

What will be the Output?

```
1   #include <iostream>
2   using namespace std;
3
4   main(){
5       int money = 0;
6       string meal = "fruit";
7                       1       1       0       0       0
8       if(meal == "fruit" || meal == "sandwich" && money >= 2)
9           cout << "Lunch being delivered" << endl;
10      }
11      else{
12          cout << "Cannot deliver Lunch" << endl;
13      }
    }
```

Logical Operators

Which one is correct?

```
1   #include <iostream>
2   using namespace std;
3
4   main(){
5       int money = 0;
6       string meal = "fruit";
7
8       if(meal == "fruit" || meal == "sandwich" && money >= 2){
9           cout << "Lunch being delivered" << endl;
10      }
11      else{
12          cout << "Cannot deliver Lunch" << endl;
13      }
    }
```

Precedence Order

Before answering Which one is correct, we must know about the precedence of logical operators.

Precedence Order	Operator	In C++
1	Not	!
2	AND	&&
3	OR	||

What will be the Output?

```
1   #include <iostream>
2   using namespace std;
3
4   main(){
5       int money = 0;
6       string meal = "fruit";
7
8       if(meal == "fruit" || meal == "sandwich" && money >= 2){
9           cout << "Lunch being delivered" << endl;
10      }
11      else{
12          cout << "Cannot deliver Lunch" << endl;
13      }
    }
```

Learning Outcome

In this lecture, we learnt how to write a C++ program for complex conditional statements with multiple Boolean expressions using AND, OR and NOT logical operators while considering the Precedence Rules.

Conclusion

- C++ supports three types of logical operators (AND, OR and NOT).
- Logical operators are used to combine multiple conditions so that these conditions can be applied in a single if statement. The result of the operation of a logical operator is a boolean value either true or false.
- The AND operator is used to combine multiple conditional statements and it returns true only when the conditions around it are true.
- OR operator returns true when any one or both of the conditions are true.
- NOT operator reverses the result.
- The order of precedence between logical operators is
  NOT
  AND
  OR

Self Assessment

Solve Following Programs

Ali is a teacher, he needs a program which helps him to compile his class results. He has 5 subjects (English, Math, Chemistry, Social Science and Biology) marked in detail. Program asks the user to enter 5 subjects' marks including student name and displays the total marks, percentage, grade (by percentage) and student name. Every subject has a total 100 marks. Grading policy details are mentioned below in table

90-100 percentage	A+
80-90 percentage	A
70-80 percentage	B+
60-70 percentage	B
50-60 percentage	C
40-50 percentage	D
Below 40 percentage	F

Self Assessment

Solve Following Programs

Write a program that asks the user for 3 different integers. If one of those integers is equal to or greater than 50, print out "One of Value is too large."

Write a program that asks the user which province they live in. If the province isn't "Sindh", print out "You should come visit Sindh sometime!".

Self Assessment

Solve Following Question

Insert parentheses into the following expression to show how operator precedence groups operands :

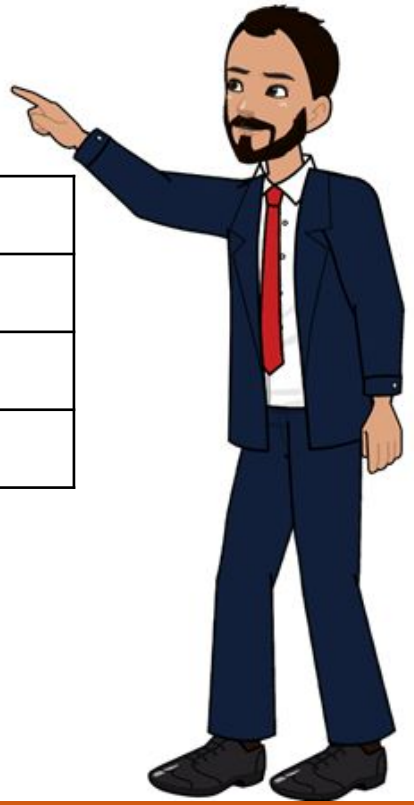
```
a > b && 45 <= sum || sum < a + b && d > 90
```

Don't change the meaning of the expression; use parentheses to make the order of evaluation clear.

Self Assessment

Fill Following Table

A || B && C	means	A || (B && C)
A && B || C && D	means	
A && B && C || D	means	
!A && B || C	means	



Self Assessment

Solve Following Programs

Write the code which asks for a login.
If the visitor enters "Admin", then prompt for a password. If the input is an empty line – show “Canceled”. If it is another string, then show “I don’t know you”.
The password is checked as follows:
If it equals “TheMaster”, then show “Welcome!”,
Another string – show “Wrong password”
For an empty string or cancelled input, show “Canceled”
as shown in the diagram in the next slide

Self Assessment

Solve Following Programs