Introduction to Programming

CSC1102 &1103

Lecture-3

American International University Bangladesh
Dept. of Computer Science
Faculty of Science and Information Technology

Lecture 3: Outline

- **Making Decisions**

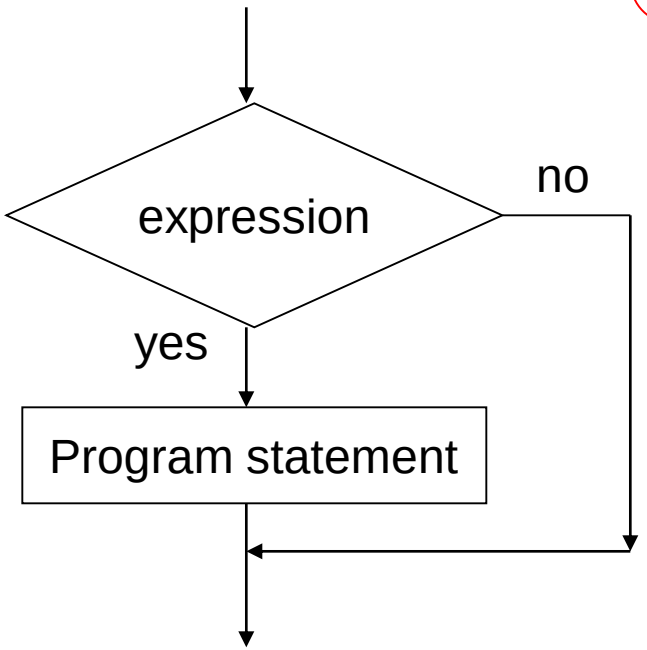
The `if` Statement

  - The `if-else` Construct
  - Logical Operators
  - Boolean Variables
  - Nested `if` Statements
  - The `else if` Construct
  - The `switch` Statement
  - The Conditional Operator

- **Character Input/Output**

The `if` statement

if (*expression*)
program statement



If expression is true (non-zero), executes statement.
If gives you the choice of executing statement or skipping it.

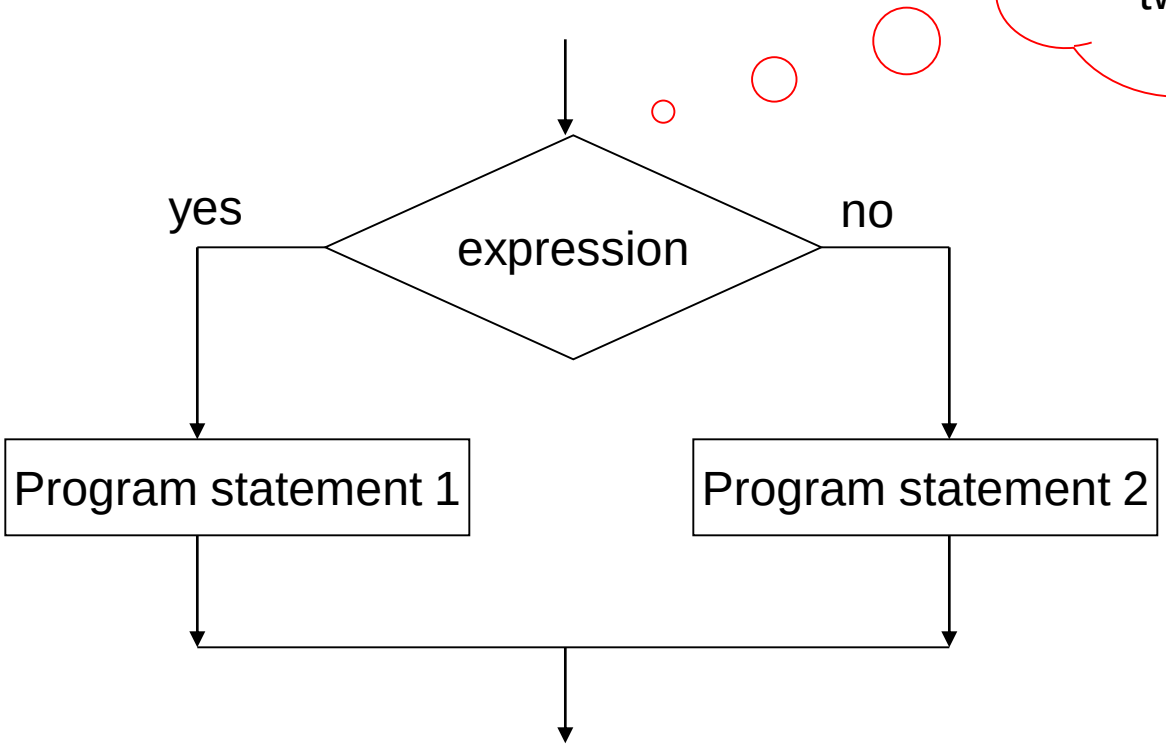
Example - `if`

```
// Program to calculate the absolute value of an integer
int main ()
{
    int number;
    cout<<"Type in your number: “<<endl;
    cin>>number;
    if ( number < 0 )
        number = -number;
    cout<<"The absolute value is “<<number<<endl;
    return 0;
}
```

The `if-else` statement

```
if ( expression )
        program statement 1
else
        program statement 2
```

if-else statement:
enables you to
choose between
two statements



Example: `if-else`

```
// Program to determine if a number is even or odd
int main ()
{
    int number_to_test, remainder;
    cout<<"Enter your number to be tested: "<<endl;
    cin>>number_to_tes;
    remainder = number_to_test % 2;
    if ( remainder == 0 )
        cout<<"The number is even"<<endl;
    else
        cout<<"The number is odd"<<endl;
    return 0;
}
```

Example: compound relational test

```
// Program to determine if a year is a leap year
int main (void)
{
    int year, rem_4, rem_100, rem_400;
    cout<<"Enter the year to be tested: "<<endl;
    cin>>year;
    rem_4 = year % 4;
    rem_100 = year % 100;
    rem_400 = year % 400;
    if ( (rem_4 == 0 && rem_100 != 0) || rem_400 == 0 )
        cout<<"It's a leap year."<<endl;
    else
        cout<<"It's not a leap year."<<endl;
    return 0;
}
```

Logical operators

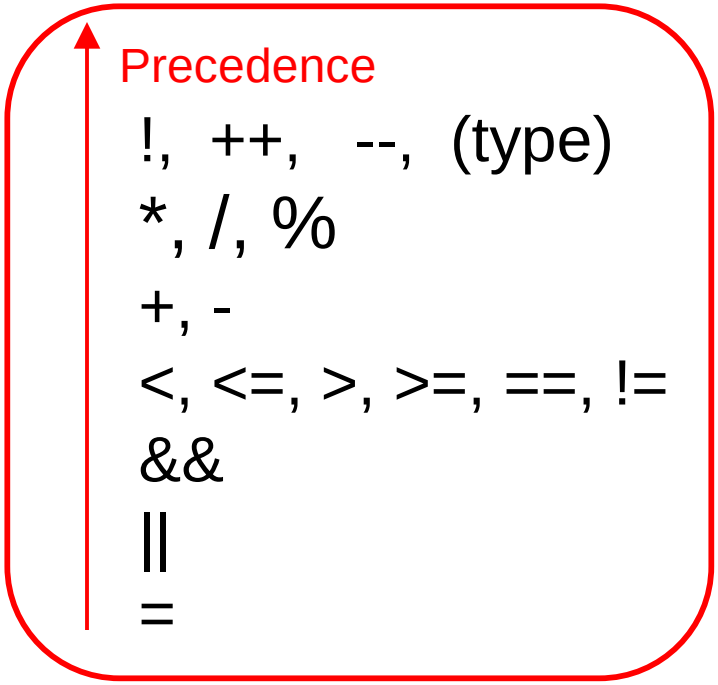
Operator	Symbol	Meaning
AND	`&&`	X && y is true if BOTH x and y are true
OR	`||`	X || y is true if at least one of x and y is true
NOT	!	!x is true if x is false

Logical values as operands or in tests: true = non-zero, false=zero

Logical values returned as results of expressions: true = 1, false=zero

Example: `5 || 0` is `1`

Precedence of operators



Example for operator precedence:

```
a > b && b > c || b > d
```

Is equivalent to:

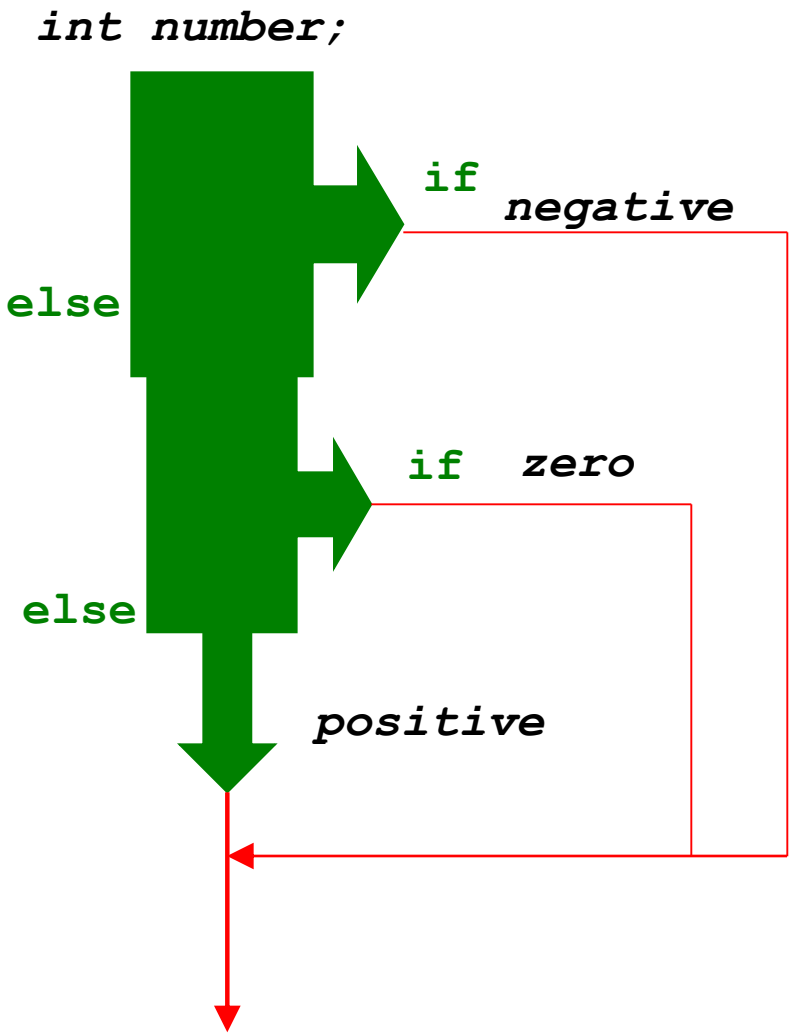
```
((a > b) && (b > c)) || (b > d)
```

Nested `if` statements

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

```
void main()
{
      int a,b,c;

      cout << "\nEnter value of A : ";
      cin >> a;

      cout << "\nEnter value of B : ";
      cin >> b;

      cout << "\nEnter value of C++ : ";
      cin >> c;

      if (a>b)
        {
              if (a>c)
                cout << "\n\nA is Greatest";
               else
                 cout << "\n\nC is Greatest";
        }
      else
        {
               if (b>c)
                  cout << "\n\nB is Greatest";
                else
                   cout << "\n\nC is Greatest";
         }
}
```

Multiple choices – `else-if`




```
if ( expression 1)
    program statement 1
else if ( expression 2)
    program statement 2
else
    program statement 3
```

Program style: this unindented formatting improves the readability of the statement and makes it clearer that a three-way decision is being made.

Example – multiple choices

```
/* Program to evaluate simple expressions of the form
number operator number */
int main (void) {
    float value1, value2;
    char operator;
    cout<<“Type in your expression”<<endl;
    cin>>value1>>operator>>value2;
    if ( operator == '+' )
        cout<<value1 + value2<<endl;
    else if ( operator == '-' )
        cout<<value1 - value2<<endl;
    else if ( operator == '*' )
        cout<<value1 * value2<<endl;
    else if ( operator == '/' )
        cout<<value1 / value2<<endl;
    else cout<<"Unknown operator.";
    return 0;
}
```