CS161: Introduction to Computer Science I

Week 3

10/2022

Your Algorithm (cont)...

Your algorithm is not:

- I sat down in front of the PC
- I got into the editor
- I entered my program
- I tried to compile it but got errors

I DON'T WANT TO SEE THIS!!!!

Your Algorithm...

First define the major tasks

Then break down these into subtasks

For example, the major tasks might be:

1. Welcome the user
2. Get the loan amount, interest rate, duration
3. Calculate the monthly payment
4. Display the results
5. Sign off Message

Not Detailed Enough!
Not Detailed Enough!
But...a good start...

What is for today?

fit@hcmus

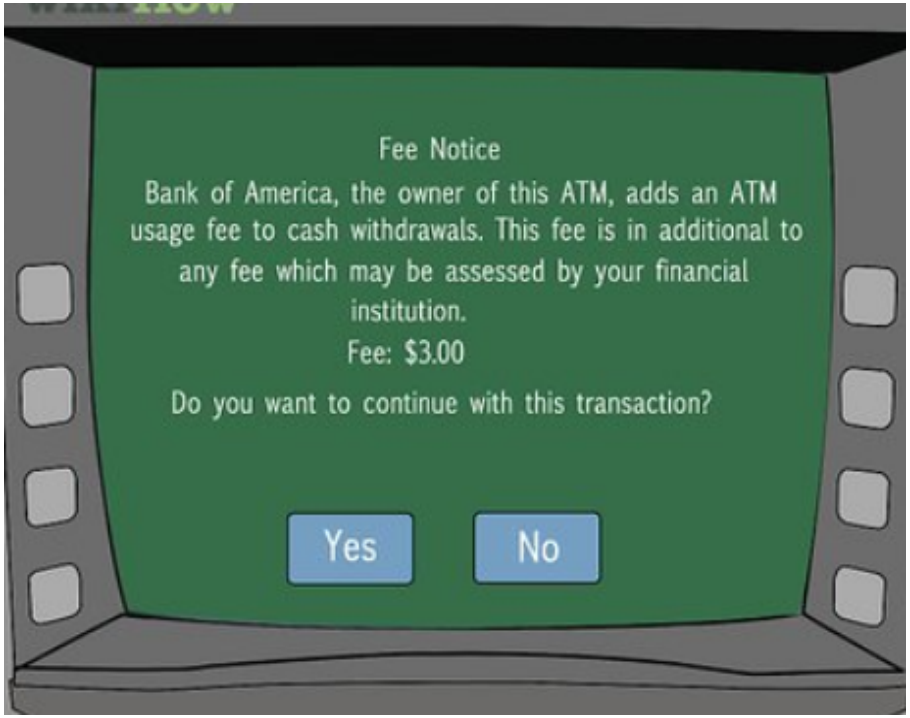
Selective Execution

- `if` statements
- Conditional statements
- switch statement

Selective Execution

Most programs are not as simple as converting inches to mm!
We need to select from alternatives...

- think of the ATM example...
- this can be done using an `if` statement
- an `if` allows us to select between **2** choices
- for example, we can select one thing or another, depending on the user



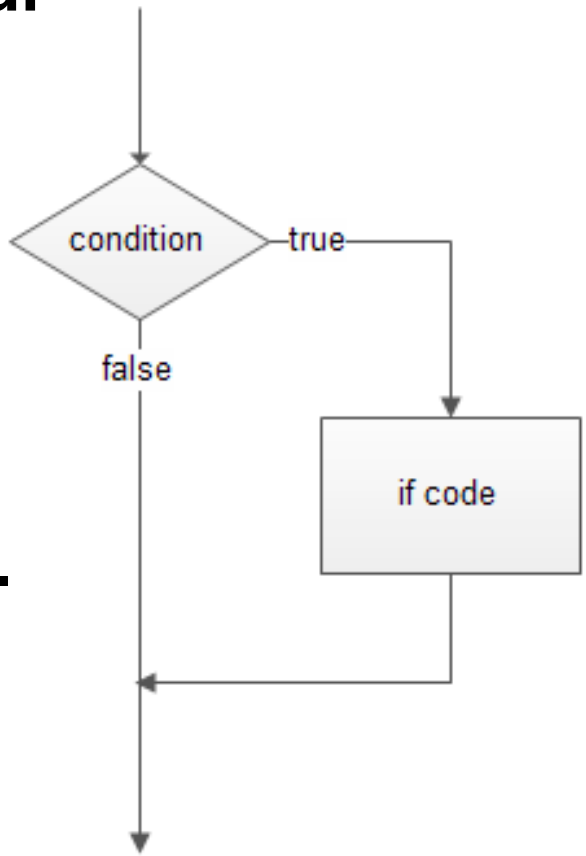
if Statements

For example, we can change our inches to mm conversion program, allowing the user to select whether they want to convert from

- inches to mm, or mm to inches!

We will give the user a choice...

- Type '**m**' to convert to mm
- Type '**i**' to convert to inches



if Statements have the form...

fit@hcmus

1) Syntax in C++, one alternative:

```
if (logical expression)
        single C++ statement;
```

```
char selection;
cout << "Enter a selection (m or i): " ;
cin >> selection;
if (selection == 'q')
   cout << "Your selection was incorrect"
        << endl;
```

if Statements have the form...

fit@hcmus

2) Syntax in C++, 2 alternatives:

```
if (logical expression)
        single C++ statement;
else
        single C++ statement;
```

```
char selection;
cout << "Enter a selection (m or i): " ;
cin >> selection;
if (selection == 'm')
   cout <<"Converting inches -> mm"<<endl;
else
   cout <<"Converting mm -> inches"<<endl;
```

if Statements have the form...

This means that either the first statement is executed when running your program OR the second statement is executed. BOTH sets of statements are NEVER used.

If the comparison is true - the first set is used;

If the comparison is false - the second set is used;

if Statements have the form...

When an `if` is encountered, the logical expression is TRUE if it is **non zero.** In this case, the statement following the expression is executed.

Otherwise, if the logical expression evaluates to **zero** it means it is FALSE. In this case, if there is an else the statement following the else is executed.

If there is no else then nothing is done if the logical expression evaluates to **zero** (FALSE).

if Statements have the form...

3) Syntax in C++, two or more alternatives:

```
if (logical expression)
        single C++ statement;
else if (logical expression)
        single C++ statement;
```

```
char selection;
cout << "Enter a selection (m or i): " ;
cin >> selection;
if (selection == 'm')
   cout <<"Converting inches -> mm"<<endl;
else if (selection == 'i')
   cout <<"Converting mm -> inches"<<endl;
```

if Statements have the form...

fit@hcmus

You might want more than a single statement to be executed given an alternative...so instead of a single statement, you can use a **compound statement.**

```
if (logical expression)
{
      Many C++ statements;
}
else //optional
```

Example of `if` Statements

fit@hcmus

```
#include <iostream>
using namespace std;
int main() {
   char selection;   //the user's answer
   float inches, mm;

   //prompt for input from the user
   cout << "Enter i to convert to inches"
        << " and m to convert to mm: ";
   cin >> selection; //get the response
```

Example of `if` Statements

fit@hcmus

```
if (selection == 'm') //notice expression!
{
    cout << "Enter the # inches: ";
    cin >> inches;
    mm = 25.4 * inches;
    cout << inches << "in converts to "
         << mm << " millimeters" << endl;
}
```

Example of `if` Statements

fit@hcmus

```
else //selection is not an 'm'
{
     cout << "Enter the # millimeters: ";
     cin >> mm;
     inches = mm / 25.4;
     cout << mm << "mm converts to "
          << inches << " inches" << endl;
}
```

Or, use the `else if` sequence…

fit@hcmus

```
else if (selection == 'i') //selection is not an 'm'
{
    cout << "Enter the # millimeters: ";
    cin >> mm;
    inches = mm / 25.4;
    cout << mm << "mm converts to "
         << inches << " inches" << endl;
}
else   //selection is not an 'm' nor 'i'
   cout << "Neither i nor m were selected"
        << endl;
```

What is next?

More Selective Execution

- Logical Operations: && and ||
- Truth Tables
- Applying logicals to if statements

Logical Operators

Logical operators evaluate the expression(s) to obtain a boolean result (TRUE/FALSE)

Operator	Decription
!	Operator NOT
&&	Operator AND
||	Operator OR

When using the logical operators, C++ only evaluates what is necessary from **left to right** to come up with the combined relational result, ignoring the rest.

Logical Operators

&& evaluates to true if both of its operands are true;

- otherwise it is false.

|| evaluates to true if one or the other of its operands are true;

- it evaluates to false only if both of its operands are false.

! gives the boolean complement of the operand.

- If the operand was true, it results in false.

Logical Expressions

Expression	Logical	T/F
5 > 100	0	False
100 > 5	1	True
5 == 200	0	False
10 == 10	1	True
10 = 10	INVALID	

BE CAREFUL when checking for equality to use == and not use the assignment operator =)

Logical Operators

fit@hcmus

Conditional Expression Logical value True/False

(5 == 10) && (30 < 88)

(5 == 10) || (30 < 88)

!(5==10) && (30 < 88)

40 != 44

!(40 != 44)

AND Truth Table

op1 && op2 results in:

op1	op2	residual value	
true	true	true	1
true	false	false	0
false	true	false	0
false	false	false	0

OR Truth Table

fit@hcmus

op1 || op2 results in:

op1	op2	residual value	
true	true	true	1
true	false	true	1
false	true	true	1
false	false	false	0

NOT Truth Table

fit@hcmus

!op1 results in:

op1	residual value	
true	false	0
false	true	1

Logicals in `if` Statements

Now let's apply this to the if statements.

For example, to check if our input is only an 'm' or an 'i'

```
char selection;
cin >> selection
if ( ('m' != selection) &&
     ('i' != selection) )
    cout << "Error! Try again" << endl;
```

Logicals in `if` Statements

fit@hcmus

Why would the following be incorrect?

```
char selection;
cin >> selection
if ( ('m' != selection) ||
     ('i' != selection) )
   cout << "Error! Try again!" << endl;
```

→ *Because no mater what you type in (m, i, p, q)*

- it will never be both an m and an i!
- If an m is entered, it won't be an i!!!!!

Logicals in `if` Statements

Let's change this to check if they entered in either an m or an i: (this is correct)

```
char selection;
cin >> selection
if ( ('m' == selection) ||
     ('i' == selection) )
   cout << "Correct!" << endl;
else
   cout << "Error! Try again!" << endl;
```

Logicals in `if` Statements

Now, let's slightly change this....

```
char selection;
cin >> selection
if ( !((‘m’ == selection) ||
       (‘i’ == selection)) )
   cout << “Error! Try again!” << endl;
```

Notice the parens...you must have a set of parens around the logical expression

Exercise

Write an algorithm to classify a GPA to rank A, B, C, D, F using if-else statement:

- GPA ≥90: rank A
- 80 ≤GPA < 90: B
- 70 ≤GPA < 80: C
- 65 ≤GPA < 70: D
- GPA < 65: F

Nested Statements

fit@hcmus

```
char rank;
if ( gpa >= 80)
{
        if (gpa < 90) // 80≤gpa < 90
                rank = 'B';
        else // gpa ≥90
                rank = 'A';
}
```

Multiway if-else Statement

fit@hcmus

```
char rank;
if ( gpa >= 90)
        rank = 'A';
else if (gpa >= 80) // 80≤gpa < 90
        rank = 'B';
else if (gpa >= 70) // 70≤gpa < 80
        rank = 'C';
else if (gpa >= 65) // 65≤gpa < 70
        rank = 'D';
else                          // gpa < 65
        rank = 'F';
```

The Conditional Operator

```
if (n1 > n2)
        max = n1;
else
        max = n2;
```

This can be expressed using the conditional operator:

```
max = (n1 > n2) ? n1 : n2;
```

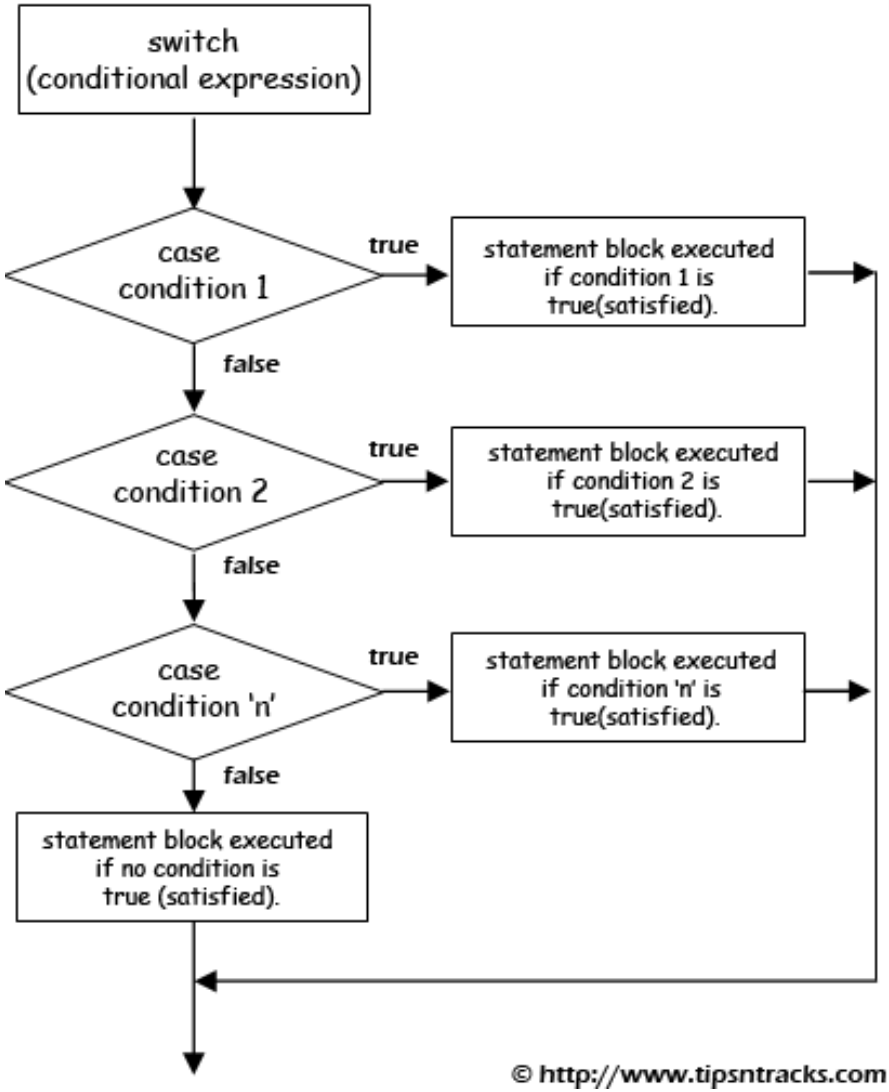
→ *It is an older programming style, we do not advise using it!!!*

Exercises

1. Write a program to find maximum between three numbers
2. Write a program to determine if the inputted year a leap year.
3. Write a program to determine if 3 numbers a, b, c are sides of a triangle.

`switch` Statements

In a menu-driven program, you can extend an `if else if` else sequence to handle many alternatives, but the C++ `switch` statement can easily handle selecting a choice from an extended list.



`switch` Statements

`switch` statement acts as a routing device that tells the computer which line of code to execute next.

On reaching a switch statement, a program jumps to the line labeled with the value corresponding to the value of ***integer-expression***.

- It must be an expression that reduces to an integer value. (for example: 2, 4, 100, ‘m’, ‘i’, …)
- Also, each label must be an integer constant expression.

`switch` Statements

If integer-expression doesn't match any of the labels, the program jumps to the line labeled **`default`**.

- **`default`** label is optional.

To make execution stop at the end of a particular group of statements, you must use the **`break`** statement.

- This causes execution to jump to the statement following the switch

switch Statements

fit@hcmus

```
switch(num)
{
   case 1:  statement 1; //if num is 1
            break;
   case 2:  statement 2; //if num is 2
            break;
   case 3:  statement 3; //if num is 3
            break;
   default: statement 4; //if num is 5
}
```

Exercises

Write a menu-driven program. The program should display a menu offering four choices, each labeled with a letter. If the user responds with a letter in one of the four valid choices, the program should use a switch to select a simple action based on the user's selection.

A program run could look something like this:

Please enter one of the following choices:
c) Carnivore p) Pianist t) Tree g) Game
User input: t
Output: Do you know maple tree?