CSCI 1300

Spring 2022 - Starting Computing

Supriya Naidu and Tom Yeh

Multiple Alternatives

- Homework 2
  - Write solutions in VSCode and paste in Autograder, Homework 2 CodeRunner.
  - Zip your .cpp files and submit on canvas Homework 2. Check the due date! No late submissions!!
- Extra-credit: coderunner (extra credit (3 points))
- Start going through the textbook readings and watch the videos
  - Take Quiz 3.
  - Check the due date! No late submissions!!
- Practice Set 2
- Week 3: 3-2-1

Today

- Good Coding Practices
- Common Errors
- Nested Statements
- `switch` statement

Practicum 1

- Coming up in week 5: Feb 7th @ 7:30 pm
- Covers material from weeks 1 – 3 and H3
  - .cpp programs
  - Variables, arithmetic, cin, cout
  - If-else, nested if-else, switch statements
- Chapters 1, 2 and 3 from the textbook (everything!)
- Two parts - *75 minutes*
  - MCQ – 4 questions
  - CodeRunner – 4 questions
- How do I prepare?
  - Practice questions

Spaces and Parentheses

```
if(a <= 8 && b > 13 && c < 1 && b <= 80 && a > 3 && c > -20)
```

vs.

```
if( (a > 3) && (a <= 8) && (b > 13) && (b <= 80) && (c > -20) && (c < 1) )
```

final else if condition

```
if(num == 0)
{
    cout << num << " is 0" << endl;
}
else if(num > 0)
{
    cout << num << " is greater than 0" << endl;
}
else if(num < 0)
{
    cout << num << " is less than 0" << endl;
}
```

final else if condition

```
if(num == 0)
{
    cout << num << " is 0" << endl;
}
else if(num > 0)
{
    cout << num << " is greater than 0" << endl;
}
else if(num < 0)
{
    cout << num << " is less than 0" << endl;
}
```

Nested conditions

```
if(x > 10)                          if((x > 10) && (y < 15) && (z == 20))
{                                   {
  if(y < 15)                            // code
  {                       vs.       }
    if(z == 20)
    {
      // code
    }
  }
}
```

Order of Conditions

```
if(income < 120000)
{
    tax_rate = 8;
}
else if(income < 100000)
{
    tax _rate = 6;
}
else if(income < 80000)
{
    tax _rate = 4.5;
}
else if(income < 60000)
{
    tax _rate = 3.25;
}
else
{
    tax _rate = 2;
}
```

Order of Conditions

```
if(income >= 100000 && income < 120000)
{
    tax_rate = 8;
}
else if(income >= 80000 && income < 100000)
{
    tax _rate = 6;
}
else if(income >= 60000 && income < 80000)
{
    tax _rate = 4.5;
}
else if(income >= 40000 && income < 60000)
{
    tax _rate = 3.25;
}
else
{
    tax _rate = 2;
}
```

Order of Conditions

```
if(income < 40000)
{
    tax_rate = 2;
}
else if(income < 60000)
{
    tax _rate = 3.25;
}
else if(income < 80000)
{
    tax _rate = 4.5;
}
else if(income < 100000)
{
    tax _rate = 6;
}
else
{
    tax _rate = 8;
}
```

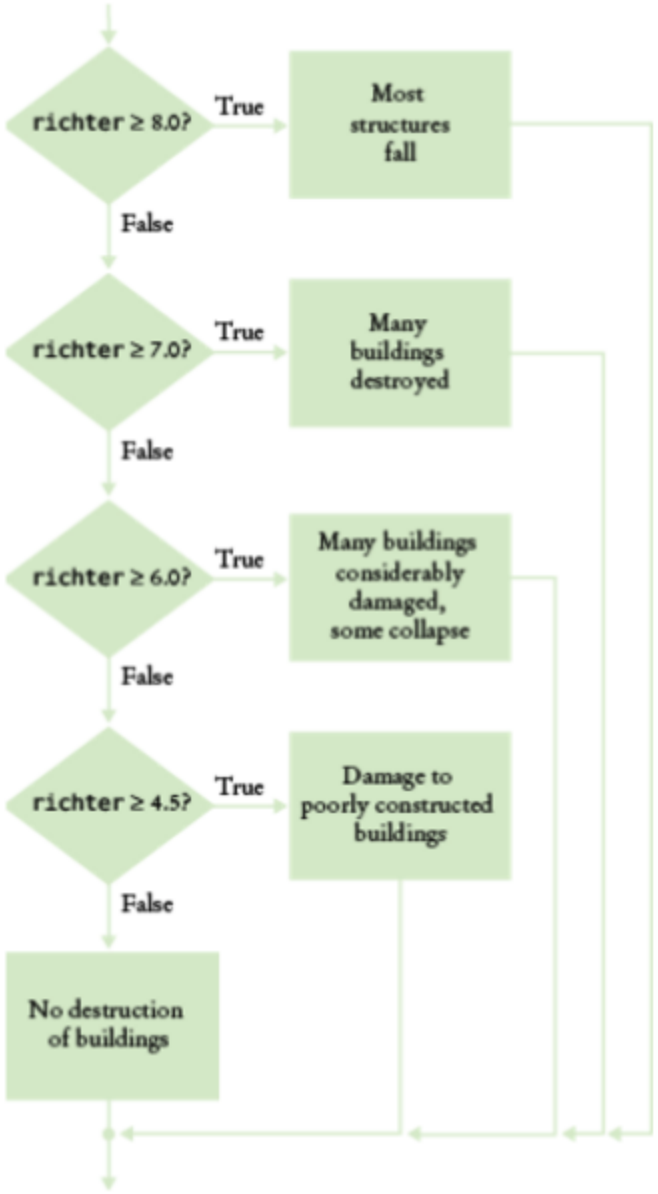
Multiple Alternatives Need Multiple Nested if() Statements

- In the case of the Richter Scale for earthquake magnitude, there are five branches:
    - one each for the four descriptions of damage, and a "default" fifth one for no destruction (not shown).

Table 3 Richter Scale

Value	Effect
8	Most structures fall
7	Many buildings destroyed

Flowchart for Richter Scale Code



Multiple Alternatives (Richter Scale Code)

```
if (richter >= 8.0)
{
  cout << "Most structures fall";
}
else if (richter >= 7.0)
{
  cout << "Many buildings destroyed";
}
else if (richter >= 6.0)
{
  cout << "Many buildings considerably damaged, some collapse";
}
else if (richter >= 4.5)
{
  cout << "Damage to poorly constructed buildings";
}
else
{
  cout << "No destruction of buildings";
}
```

Multiple Alternatives – Order of Tests

- Because of this execution order, when using multiple if statements, pay attention to the order of the conditions.

Multiple Alternatives – Wrong Order of Tests

```
if (richter >= 4.5) // Tests in wrong order
{
  cout << "Damage to poorly constructed buildings";
}
else if (richter >= 6.0)
{
  cout << "Many buildings considerably damaged, some collapse";
}
else if (richter >= 7.0)
{
  cout << "Many buildings destroyed";
}
else if (richter >= 8.0)
{
  cout << "Most structures fall";
}

// Suppose the value of richteris 7.1. Because we tested small first with a >=, the first statement is (wrongly) printed.
```

The switch Statement vs. the if statement

- Below is a complicated if() statement to choose a text string to assign based on the value of an int variable:

```
int digit;
//digit variable gets set here by some code
if (digit == 1) { digit_name = "one"; }
else if (digit == 2) { digit_name = "two"; }
else if (digit == 3) { digit_name = "three"; }
else if (digit == 4) { digit_name = "four"; }
else if (digit == 5) { digit_name = "five"; }
else if (digit == 6) { digit_name = "six"; }
else if (digit == 7) { digit_name = "seven"; }
else if (digit == 8) { digit_name = "eight"; }
else if (digit == 9) { digit_name = "nine"; }
else { digit_name = ""; }
```

The switch Statement

- The switch statement is an alternative to nested if() else statements. But switch is at least as awkward to code as nested if() else:

```
int digit; //switch can only test int and char types
... //digit variable gets set here by some code
switch(digit)
{
  case 1: digit_name = "one"; break;
  case 2: digit_name = "two"; break;
  case 3: digit_name = "three"; break;
  case 4: digit_name = "four"; break;
  case 5: digit_name = "five"; break;
  case 6: digit_name = "six"; break;
  case 7: digit_name = "seven"; break;
  case 8: digit_name = "eight"; break;
  case 9: digit_name = "nine"; break;
  default: digit_name = ""; break; //taken if none of the above
}
```

break statements in the switch statement

- Every branch of the switch must be terminated by a break statement. And each branch must terminate with a semicolon.
- break tells the machine to skip down to the end of the switch statement, because a match was found.
- If the break is missing, execution falls through to the next branch, and so on until finally a break or the end of the switch is reached.
- In practice, this fall-through behavior is rarely useful, and ***it is a common cause of errors***.
- If you accidentally forget the break statement, your program compiles but executes unwanted code. Try it and see!