Programming with If Statements using Multiple Conditions

Larry Caretto
Computer Science 106

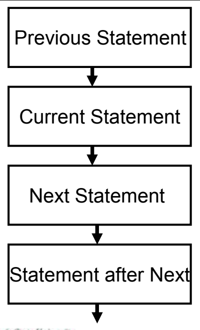
Computing in Engineering and Science

February 23, 2006

Outline

- Review last class
  - Program flow controls
  - if statements
- Exercises with if statements
- Multiple choices
- Exercises with multiple choices
- Sequential if statements versus the if-else-if structure

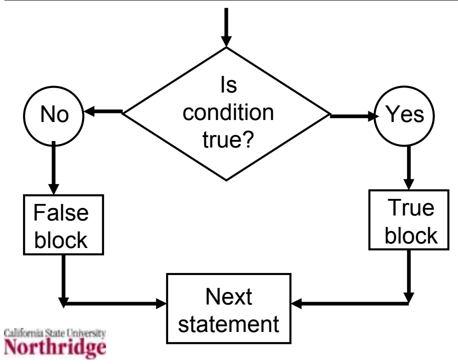
Review Sequential Flow



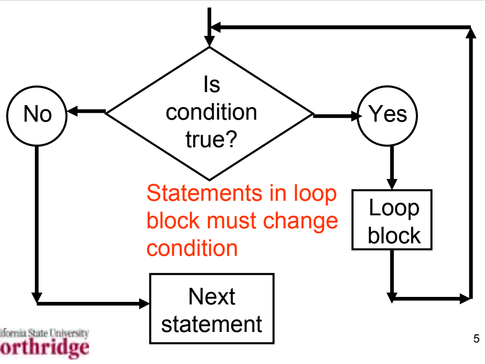
- Statements are normally executed in sequential order but you must have the correct sequence
- What is wrong with

```
y = x * x;
cin >> x;
cout << y;
```

Review Choice Statements



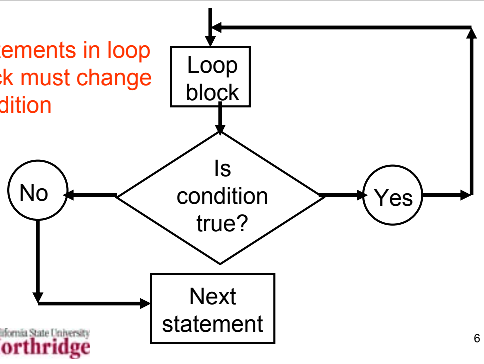
Review Choice Before Loops



Statements in loop block must change condition

Review Choice After Loops

Statements in loop block must change condition

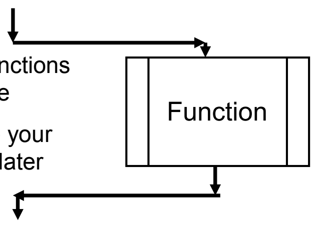


Review Function

- Transfer control and data to separate part of program
- Return control and data

Math library functions are an example

You will create your own functions later



Northridge 7

Review Conditions

- A condition is an expression that evaluates to a **bool**ean value of true or false
  - Use relational operators: greater than >, equal to ==, less than <, not equal to !=, greater than or equal to >=, less than or equal to <=
  - Logical operators: not !, and &&, or ||
  - Examples: hours > 40, wind > 20 && temperature < 30

Northridge 8

Exercise on Conditions

- Use relational (<, >, <=, >=, ==, !=) and logical (!, &&, ||) operators to write conditions for the following:
- An integer variable year is ***not*** evenly divisible by four
- year % 4 != 0 or !(year % 4 == 0)
- A string variable status equals “single” and an integer variable dependents is 0
- status == “single” && dependents == 0

Northridge 9

Review if Statements

- Implementation of choice statements in most high-level languages uses an if statement
- The C++ format is

if (***<condition>***)

{

<statements done if condition true>

}

Northridge 10

Review if-else Statements

- Executes different statement blocks if condition is true or false

if (***<condition>***)
{
<statements done if condition true>
}
else
{
<statements done if condition false>
}
<Next statement after one block done>

Northridge 11

Writing if Statements Exercise

- Define variable inc for “income”, deduct for “deductions” and ti “taxable income”
- Taxable income is income minus deductions, but is never less than zero
- Write code to compute taxable income

```
double ti = inc – deduct;
if ( ti < 0 )
    {  ti = 0; }
```

Northridge 12

Exercise

- Write a program that delcares and reads a type double variable x, and determines if it is greater than zero
  - If x > 0 compute and print the natural logarithm using the log() function
    - Also print the value of x input by the user
  - Otherwise print an error message that you cannot compute log of a negative number

Exercise Solution

```
double x;
cout << “Enter a value for x: “
cin << x;
if ( x > 0 )
    cout << “The natural log of “
         << x << “ is “ << ln(x);
else
    cout << “Cannot compute log for “
         << “negative input x = “
         << x;
```

Multiple Conditions

- Can have several choices
  - Example is an empirical function for y(x) with different equations for y used in different ranges of x
- Structure to handle this is called if-else-if block
- Allows initial if (and associated code) to be followed by several other statements like else if (***<new condition>***)

if – else – if Structure

if (***<condition1>***)
{
<statements done if condition1 true>
}
else if (***<condition2>***)
{
<statements done if condition2 true>
}
// Place additional conditions here
// Continue on next chart

if – else – if Structure

// Continued from previous chart
else if (***<conditionN>***)
{
<statements done if conditionN true>
}
else // optional to have this final else
{
<statements done if all conditions false>
}
<Next statement after any block done>

if – else – if Operation

- In this structure only one block of code – the code associated with the first true condition – is executed
- Conditions are scanned from top to bot-tom until the first true condition is found
- The code associated with that condition is executed and control is transferred to the first statement after the final block in the if – else – if structure

if – else – if Operation II

- Because only one block of code – the code associated with the first true condition – is executed we have information at else-if conditions
- Example, what do we know about x at the else-if statement in the following?

```
if ( x < 0 )
     y = 0;
else if ( x …
```

If $x < 0$, we would set y = 0 and exit the if-else-if structure. If we get to the else-if statement we know $x \geq 0$

Northridge 19

Example/Exercise

- How do you program the following definition of an empirical function y(x)?
- If $x < 0$, then $y = 0$.
- If $0 \leq x < 1$, then $y = 0.1 x$
- If $1 \leq x < 10$, then $y = (x – 0.8) / 2$
- If $10 \leq x < 100$, then $y = 4.6 + 0.2(x – 10)^3$
- If $x >= 100$, then $y = 1624.6$

Northridge 20

Answer to Exercise

- If $x < 0$, then $y = 0$.
- If $0 \leq x < 1$, then $y = 0.1 x$
- If $1 \leq x < 10$, then $y = (x – 0.8) / 2$

```
if ( x < 0 )
{     y = 0;  }
else if ( x < 1 )   // (  x >= 0 && x < 1 )??
{     y = 0.1 * x; }
else if ( x < 10 )  // (  x >= 1 && x < 10 )??
{  y = ( x – 0.8 ) / 2; }
```

Northridge 21

Answer to Exercise II

- If $10 \leq x < 100$, then $y = 4.6 + 0.2(x – 10)^3$
- If $x >= 100$, then $y = 1624.6$

```
else if ( x < 100 )  //(  x >= 10 && x < 100 )
{     y = 4.6 + 0.2 * pow( x – 10, 3);  }
else   // else if (  x >= 100)??
{     y = 1624.6; }
```

Northridge 22

Another Exercise

- A diagnostic test has the following result
  - Score ≥ 75 – take first course
  - 65 ≤ score < 75 take two-week prep course
  - Score < 65 take four-week prep course
- Complete the following code, using if statements to print out the correct result

```
int score;
cout << “Enter your score: “;
cin >> score;
```

Northridge 23

Another Exercise Solution

```
if ( score >= 75 )  {
   cout << “Take college course”;
}
else if ( score >= 65 )  {
    cout << “Take two-week prep course”;
}
else  {
    cout << “Take four-week prep course”;
}
```

Northridge 24

Use of if versus if-else-if

```
if ( a == 0 && b == 0 && c == 0 )
    x = 0;
if ( a == 0 && b == 0 )
    x = 1;
```

- What is difference between code above and code below?

```
if ( a == 0 && b == 0 && c == 0 )
    {   x = 0;  }
else if ( a == 0 && b == 0 )
    {   x = 1;  }
```

25

Use of if versus if-else-if II

```
if ( a == 0 && b == 0 && c == 0 )
    x = 0;
if ( a == 0 && b == 0 )
    x = 1;
```

- Code above has two separate ifs
  - Braces not needed since there is only one statement for each if
  - Second if is always executed and, in fact, is only one that matters – if a, b, and c are all zero we will get x = 1 with this code

26

Use of if versus if-else-if III

- Code below is a single if-else-if
  - Braces not necessary here
  - If a, b, and c are all zero we set x= 0 and exit the if-else-if structure
  - If a and b are zero and c is nonzero we set x = 1 and exit the structure

```
if ( a == 0 && b == 0 && c == 0 )
    {   x = 0;  }
else if ( a == 0 && b == 0 )
    {   x = 1;  }
```

27

Another Exercise

- Credit, no-credit grading rules
  - Graduate students: B- or better is credit
  - Undergraduates: C- or better is credit
- Data system has string variable status (grad or ugrd) and grade variable as type double (1.7/2.7 for C-/B-)
- Write code to examine these variables and print correct value of credit or no credit

28

Another Exercise Solution

```
if ( status == “grad” && grade >= 2.7 )
{     cout << “Grade is credit.”; }
else if ( status == “ugrd” && grade >= 1.7 )
{     cout << “Grade is credit.”; }
else
{     cout << “Grade is not credit.”; }
```

29