Ch 4. Making Decisions

CS 1428
Fall 2011

Jill Seaman

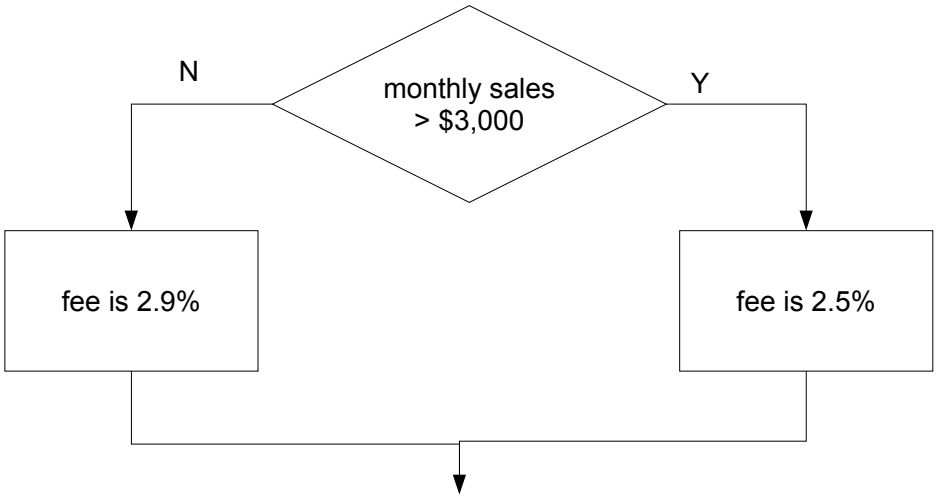
Lecture 9

Straight-line code

- So far all of our programs have followed this basic format:
  - Input some values
  - Do some computations
  - Output the results
- The statements are executed in a sequence, first to last.

Decisions

- Sometimes we want to be able to decide NOT to execute certain statements:



Relational Expressions

- Making decisions require being able to ask "Yes" or "No" questions.
- Relational expressions evaluate to true or false.
- Also called
  - logical expressions
  - conditional expressions
  - boolean expressions

Relational Expressions

- Boolean literals:
  - true
  - false
- Boolean variables:

```
bool isPositive;
bool found;

isPositive = true;
found = false;
```


Relational Operators

- Binary operators used to compare numbers:
  - < Less than
  - <= Less than or equal to
  - > Greater than
  - >= Greater than or equal to
  - == Equals (note: do not use =)
  - != Not Equals

Relational Operators

- Examples

```
int x=6;
int y=10;

a. x == 5
b. 7 <= x + 2
c. y – 3 > x
d. x != y
```

- Can assign relational exprs to variables:

```
bool isPositive, found;
int x;

cin >> x;
isPositive = x > 0;
found = x == 100;
```

- Relational ops have higher precedence than =

Precedence and Relational Operators

- Relational operators are lower than arithmetic operators:

```
int x, y;

...  x < y -10 ...        // minus happens first
...  x * 5 >= y + 10 ... // mult, then plus, then >=
```

- Relational operators are higher than assignment:

```
int x, y;
...
bool t1 = x > 7;             // > then =
bool t2 = x * 5 >= y + 10;   // *, +, >=, =
```

if-else

- if-else statement is used to express decisions

```
if (expression)
   statement1
else
   statement2
```

- expression is evaluated:
  - If it is true, then statement1 is executed. (statement2 is skipped).
  - If it is false, then statement2 is executed (statement1 is skipped).

if-else example

- For example:

```
double rate;
double monthlySales;

cout << "Enter monthly sales last month: " << endl;
cin >> monthlySales;

if (monthlySales > 3000)
   rate = .025;
else
   rate = .029;

double price;
cout << "Enter selling price of item: " << endl;
cin >> price;
double commission = (price + 3.99) * rate;
```

if-else structure

- Notice:

```
if (monthlySales > 3000)
   rate = .025;
else
   rate = .029;
```

- relational expression in parentheses
- NO semi-colon after expression, nor else
- Good style: indent the statements
- The semi-colons belong to the statements, not to the if-else

the block statement

- a block (or a compound statement) is a set of statements inside braces:

```
{
   int x;
   cout << "Enter a value for x: " << endl;
   cin >> x;
   cout << "Thank you." << endl;
}
```

- This allows us to use multiple statements when by rule only one is allowed.

if-else with blocks

- We can use blocks to put more than one statement in the branches of the if-else:

```
int number;
cout << "Enter a number" << endl;
cin >> number;

if (number % 2 == 0)
{
   number = number / 2;
   cout << "0";
}
else
{
   number = (number – 1) / 2;
   cout << "1";
}
```


if statement

- The else part is optional:

```
if (expression)
   statement1
```

- expression is evaluated:
  - If it is true, then statement1 is executed.
  - If it is false, then statement1 is skipped.

if statement example

- Example:

```
cout << "Enter a positive number: ";
cin >> x;
if (x < 0)
{
   cout << "That number is negative. "
        << "Please enter a positive number: ";
   cin >> x;
}

//do something with x here
```


Watch out

- What is output?

```
int x;
x = 13;
if (x==10)
   x = 17;
   cout << x << endl;
cout << "Done!" << endl;
```

- What is output?

```
char babyGender;
cin >> babyGender;
if (babyGender == 'M')
   cout << "It's a boy!" << endl;
cout << "It's a girl!" << endl;
```

Nested If statements

- if-else is a statement. It can occur as a branch of an if-else statement.

```
char bornInUSA;
int age;

if (bornInUSA = 'Y')
   if (age >= 35)
      cout << "You qualify to run for President" << endl;
   else
      cout << "You are too young to run for President" << endl;
else
   cout << "You must have been born in the US in order " <<
           "to run for President" << endl;
```


Dangling Else Problem

- Combining an if with an if-else:

```
if (a > 0)
   if (b > 0)
      cout << "*****" << endl;
   else
      cout << "-----" << endl;
```

- Or is it:

```
if (a > 0)
   if (b > 0)
      cout << "*****" << endl;
else
   cout << "-----" << endl;
```

- It's the first one. The else is paired with the closest if.

To override dangling else convention

- Add braces:

```
if (a > 0)
{
   if (b > 0)
      cout << "*****" << endl;
}
else
   cout << "-----" << endl;
```


Common nested if pattern

- Determine letter grade from testScore:

```
if (testScore < 60)
   grade = 'F';
else {
      if (testScore < 70)
         grade = 'D';
      else {
            if (testScore < 80)
               grade = 'C';
            else {
                  if (testScore < 90)
                     grade = 'B';
                  else
                     grade = 'A';
                }
          }
    }
```


- Note the braces are actually optional here

if-else if (aka else-if)

- Not really a different statement, just a different way of indenting the previous nested if statement:

```
if (testScore < 60)
   grade = 'F';
else if (testScore < 70)
   grade = 'D';
else if (testScore < 80)
   grade = 'C';
else if (testScore < 90)
   grade = 'B';
else
   grade = 'A';
```