EECS 211, Winter 2014

Loops

William Hendrix

Lecture 4

Outline

- Loops
  - `while`
  - `do...while`
  - `for`
- Loop control flow
- Introduction to Visual Studio Express
- Minilab

Loops

- Syntax

```
while (condition) //No semicolon!
{
  <stmt(s) to run while condition is true>;
}
```

- `condition` tested at start
  - Statements executed if `true`
  - Test `condition` and repeats until `false`
- Braces not needed for one statement
- **Common error:** the infinite loop
  - At least one variable in `condition` needs to be updated in loop
  - Otherwise, loop runs forever
  - Program seems to "hang" or produce tons of output

Test-last loops

- Syntax

```
do
{
  <statements>;
} while (condition);
```

- `condition` not tested until end
- Always executes at least once

- Example: input checking

```
do
{
  cout << "Please enter a number: ";
  cin.clear();   //Clears error status for cin
  cin >> number;
  cin.sync();
} while (cin.fail());  //Or:  while (!cin);
```

A common construction

- Loop controlled by integer variable

```
int num, count = 0, sum = 0;
while (count < 10)
{
  cout << "Enter #" << count + 1 << ": ";
  cin >> num;
  sum += num;
  count++;  //Infinite loop if you forget!
}
```

- 3 parts: initialize variable, update, test
- Shortcut: `for` loops

The `for` loop

```
int num, sum = 0;
for (int count = 0; count < 10; count++)
{
   cout << “Enter #” << count + 1 << “: ”;
   cin >> num;
   sum += num;
}
```

- Identical execution to previous code (`while` semantics)
- Helps you remember to update variable
- Can define new variable (as above) or use existing one
  - Declared variables disappear after the loop
- Convention: variable `i` often used for `for` loops
- Convention: variables start at 0
- Used mainly with increment or decrement by a constant

Loop control

- `break`
  - Exits a loop immediately
  - Generally used in conjunction with `if`
  - Could always rewrite code to remove `break`

- `continue`
  - Moves directly to next loop iteration
    - Skips rest of statements in loop
  - Generally used in conjunction with `if`
  - Could always rewrite code to remove `continue`

Loop control example

```
int num, sum = 0;
cout << “Enter 10 numbers with total\
 less than 100” << endl;
for (int count = 0; count < 10; count++)
{
  cin >> num;
  sum += num;
  if (sum >= 100)
  {
    cout >> “Sum too large”;
    break;
  }
}
```

- Could change condition to `count < 10 && sum < 100`

Loop design

- What kind of loop should you use?
- What is the first iteration in the loop?
  - Initialize the loop variable
- What is the last iteration?
  - Careful: off-by-one errors are very easy to make
- What code is being repeated?
  - Be sure to update your loop variable