CS 161

Introduction to CS I

Lecture 7

- More practice with loops
- How can we track down bugs in our programs?

Oregon State University
College of Engineering

Updates

- Class calendar now goes through the end of the term
  - Remember: Calendar is subject to change
  - Midterm #1 review: Thurs. Jan. 30 6-7 p.m., KEC 1001
- Assignment 2 Design Document – please follow instructions carefully
  - Include name, date, assignment
  - Submit on **Canvas**, not TEACH (**why?**)
- Assignment 2 Peer Reviews – do not fill in the rubric, but instead add comments
  - See the list of 8 items you should provide to get full credit

Updates

- Hourly file backups available!
  - A very sad tale – has this happened to you?
    - `g++ -o assignment2.cpp assignment2`
  - If so, check the backup (snapshot), created each hour
    - https://it.engineering.oregonstate.edu/restore-using-snapshots

Loop summary

- `for` loop: repeat for a **specific number of times**
  - Brush teeth with 30 strokes
- `while` loop: repeat while a condition is true **(might be never)**
  - While teeth are dirty, brush them
- `do-while` loop: **always do once**, then repeat while condition is true
  - Brush teeth… while they are dirty

Oregon State University
College of Engineering

What kind of loop would you use?

A. Given a student’s grade on each assignment, calculate final grade
B. Query user to generate a grocery list
C. Search a file for the first “k” and report its location
D. Play checkers until there is a winner
E. Scrape ice off the windshield

Challenge: Re-write this for loop as a while loop

Condition

Initialize

Update

```
for (int x = 0; x < 3; x++)
  {
    dice_roll = rand()%6 + 1;
    cout << x << ") You rolled "
         << dice_roll << endl;
  }
```

Initialize

```
int x = 0;
while (x < 3)
  {
    dice_roll = rand()%6 + 1;
    cout << x << ") You rolled "
         << dice_roll << endl;
    x++;
  }
```

Condition

Update

Oregon State University
College of Engineering

Loop tricks: prefix and postfix updates

A. Postfix

```
for (int x = 1; x <= 5; x++)
{
  cout << x << endl;
}
```

B. Prefix

```
for (int x = 1; x <= 5; ++x)
{
  cout << x << endl;
}
```

Oregon State University
College of Engineering

Loop tricks: modify the loop counter

A. Postfix

```
for (int x = 1; x <= 5; x++)
{
  cout << x++ << endl;
}
```

B. Prefix

```
for (int x = 1; x <= 5; x++)
{
  cout << ++x << endl;
}
```

Oregon State University
College of Engineering

Loop tricks: characters

A

```
for (char c = 'a'; c < 'e'; c++)
{
  cout << c << endl;
}
```

B

```
for (char c = 'a'; c <= 'f'; c+=2)
{
  cout << c << endl;
}
```

Oregon State University
College of Engineering

Loop tricks: skip an iteration

```
for (char c = 'a'; c < 'e'; c++)
{
  if (c == 'c')
  {
    continue;
  }
  cout << c << endl;
}
```

Loop tricks: stop the loop

```
for (char c = 'a'; c < 'e'; c++)
{
  if (c == 'c')
  {
    break;
  }
  cout << c << endl;
}
```

Loop tricks: nested loops

- What does this print?

A

```
for (int x = 0; x < 10; x++)
{
   for (int y = 0; y < 5; y++)
   {
      cout << "CS 161!" << endl;
   }
}
```

B

```
for (int x = 0; x < 2; x++)
{
   for (int y = 0; y < 3; y++)
   {
      cout << "CS 161!";
   }
   cout << endl;
}
```

Variable reuse

- What does this print?

A

```
int x;
for (x = 0; x < 5; x++)
{
   cout << "x is: " << x << endl;
}
for (x = 0; x < 5; x++)
{
   cout << "x is: " << x << endl;
}
```

B

```
int x;
for (x = 0; x < 5; x++)
{
  for (x = 0; x < 5; x++)
  {
     cout << "x is: " << x << endl;
  }
}
```

Oregon State University
College of Engineering

Variable reuse

- What does this print?

A

```
int x;
for (x = 0; x < 5; x++)
{
   cout << "x is: " << x << endl;
}
for (x = 0; x < 5; x++)
{
   cout << "x is: " << x << endl;
}
```

C

```
int x;
for (x = 0; x < 5; x++)
{
  for (x = 0; x < 2; x++)
  {
     cout << "x is: " << x << endl;
  }
}
```

Infinite loop!

Ctrl-c to kill the program

Oregon State University
College of Engineering

Scope and "shadowing"

A

```
int x;
for (x = 0; x < 5; x++)
{
  for (x = 0; x < 2; x++)
  {
    cout << x << endl;
  }
}
```

Infinite loop!

B

```
for (int x = 0; x < 5; x++)
{
  for (int x = 0; x < 2; x++)
  {
    cout << x << endl;
  }
}
```

Outer x is "shadowed"

Not infinite loop

C

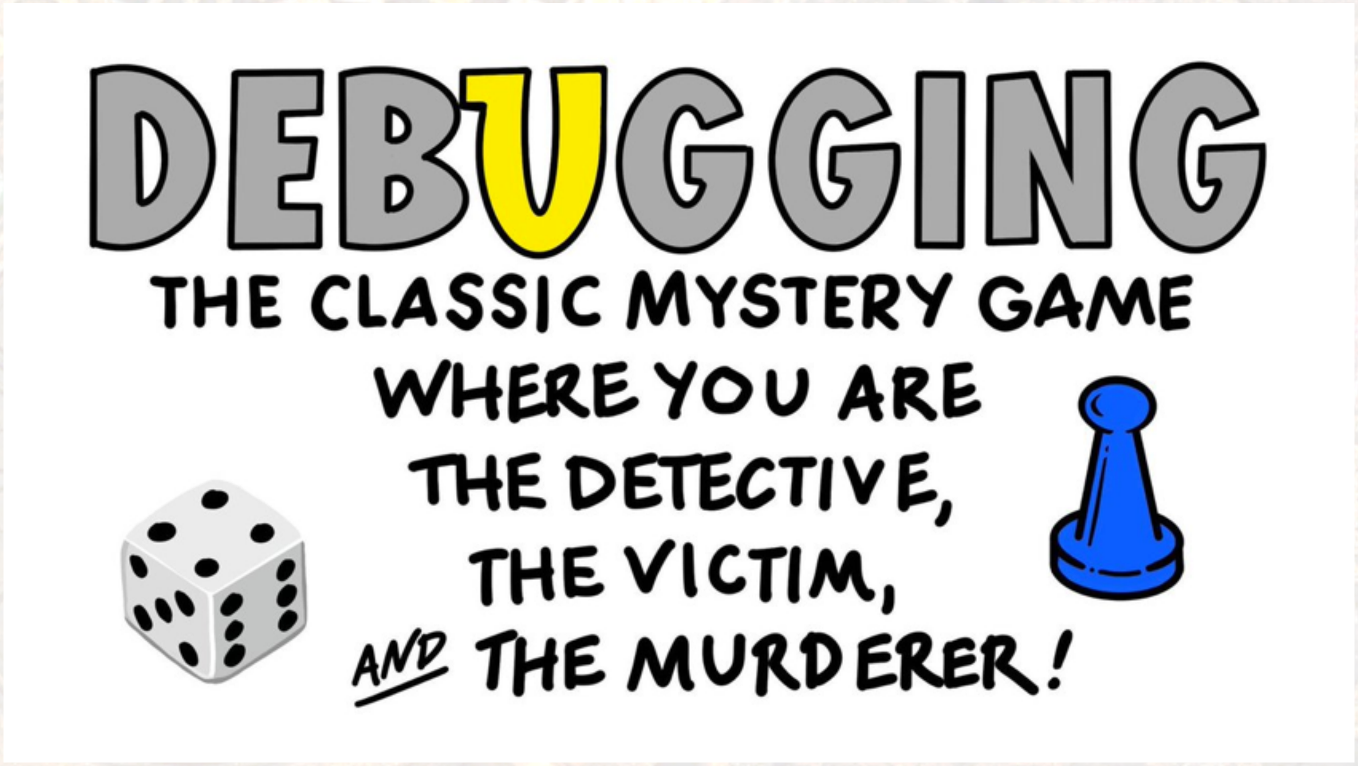
```
for (int x = 0; x < 5; x++)
{
  for (int x = 0; x < 2; x++)
  {
    int x;
    cout << x << endl;
  }
}
```

Not allowed!

{} define variable <u>scope</u> (visibility)

Oregon State University
College of Engineering

Tracking down bugs in your program



Oregon State University
College of Engineering

Bug detection tools: Is something wrong?

- Visual inspection
- Read and interpret compiler messages
  - Search the web for the exact error
- Create test cases and check that output matches input
- Trace through the code (read it out loud)

Bug localization tools: Where is it?

- Look at line numbers identified by the compiler
- Inspect program state
  - Print variables out to see what is happening during execution
  - Use `cin` to pause the program
- Check your assumptions explicitly with `assert(<expr>)`
- Trace through the code (read it out loud)
- Comment out problematic code to isolate it

Oregon State University
College of Engineering

What vocabulary did we learn today?

- Loop control: `continue, break`
- Variable scope (visibility)
- Shadowing

Oregon State University
College of Engineering

What ideas and skills did we learn today?

- Nested loops
  - Cautions for variable reuse
- How to choose the type of loop to use
- Be aware of variable scope (visibility)
- Strategies for bug detection (is something wrong?)
- Strategies for bug localization (where is it?)

Oregon State University
College of Engineering

Week 3 – it’s a short one!

- ❑ Attend lab (laptop required)
- ❑ Read **Rao Lesson 6** (pp. 128-142) – loops
  and **Miller Lecture 5** – a good summary/review
- ❑ Finish **Assignment 2 design peer review (due tonight)**
- ❑ Continue working on **Assignment 2 implementation**
  (due **Sunday, Jan. 26**)

See you Friday!