CS 161
Intro to CS I

Finish Conditionals/Begin Loops
Odds and Ends

• Assignment 3 Design due Sunday
• Back Evaluations due Sunday
Fun Friday...

- What if we didn’t have a break in the case of a switch statement?
- How could you write an if/else to match not having breaks in this switch?

```cpp
switch( x ) {
    case 0:
    case 1:
        std::cout << "X is zero or one\n";
    case 2:
        std::cout << "X is two\n";
    default:
        std::cout << "You have entered an invalid number!!!\n";
}
```

- What do you think this does in C++?

```cpp
<expression1> ? <expression2> : <expression3>
(x == 0 || x == 1) ? cout + 3 : cout + 3

if (x == 0 || x == 1) { cout + 3; }
else if (x == 2) { cout + 3; }
else { cout + 3; }
```
The for Loop Examples

```cpp
for(x=1; x <= 1; x++) {
    cout << "hello world\n";
}

for(x=1; x < 1; x++) {
    cout << "hello world\n";
}
```

0 or more times
The **while loop**

for(x=1; x <= 100; x++)
    cout << "hello world\n";

**VS.**

int x=1;
while(x<=100) {
    cout << "hello world\n";
    x++;
}

---

**Oregon State University**

**College of Engineering**
Common Mistakes

```cpp
int x = 1;
while(x <= 100) {
    cout << "hello world\n";
    x++;
}
```

What if we forget this?

What if we forget this?

Ctrl + C
The do/while loop

```java
int x=1;
do {
    cout << "hello world\n";
    x++;
} while(x<=100);
```

- Difference b/w while and do/while?
Nested for Loops

for(x = 0; x < 10; x++) {
    for(y = 0; y < 10; y++) {
        cout << "hello world\n";
    }
}

- How many times is Hello World printed?

10 times the outer loop is executed.

10 times the inner loop executed.

160
Reuse Variables

```cpp
for(x = 0; x < 10; x++) {
    cout << "The value of x is: " << x << endl;
}
```

Variables with same name

```c++
int x;
for(x = 0; x < 10; x++) {
    for(x = 0; x < 10; x++) {
        cout << "The value of x is: " << x << endl;
    }
}
```

- What is the output from this nested loop?
Infinite Loops

```cpp
int x;
for(x = 0; x < 10; x++) {
    for(x = 0; x < 5; x++) {
        cout << "The value of x is: " << x << endl;
    }
}
```
Infinite Loops

```cpp
int x, y;
for(x = 0; x < 10; x++) {
    for(y = 0; y < 5; x++) {
        cout << "The value of x is: " << x << endl;
    }
}
```
Infinite Loops

```cpp
int x, y;
for(x = 0; x < 10; x++) {
    for(y = 0; x < 5; y++) {
        cout << "The value of x is: " << x << endl;
    }
}
```
Infinite Loops

```cpp
int x;
for(x = 1; x <= 10; x++) {
    cout << "The value of x is: " << --x << endl;
}
```

14
Why is this good/bad?

```cpp
int x, y;  // declarative statement
for(int x = 0; x < 10; x++) {
    for(int y = 0; y < 10; y++) {
        cout << "hello world" << endl;
    }
}
```

- Where can we access x and y?

Old C can't have declarative statements after non-declarative statement.
Looping Recap...

• for loops
  – Repeat for specific number of times
  – Example?

• while loops
  – Repeat while a condition is being met
  – Example?

• do while loops
  – Always do once, and repeat while condition is met
  – Example?
More about break, exit, and return

- **break** – used with switch and loops, breaking out of the closest associated case or loop (for, while, or do while). **This statement can only occur in a loop or case**, otherwise the compiler yells!

- **return** – leave the current function, which exits the program when in the main() function. You can put this **anywhere inside any function**, otherwise the compiler yells!

- **exit()** – exit the entire program, no matter where this is encountered. You can put this **anywhere inside any function**, as long as you include `<cstdlib>`, otherwise the compiler yells!
How do we read a string of chars?

- User-defined type in string library
  ```
  #include <string>
  ```
- Declare/Create type
  ```
  string mssg;
  ```
- Read with cin or getline
  ```
  cin >> mssg;  //get a word
  getline(cin,mssg);  //get a line of txt
  ```