CS 161
Intro to CS I

Conditions
Odds and Ends

• Assignment 2 and Back Evals due Sunday
• No school Monday
• Questions?
Extra Credit Exercise

• Get into groups of 4-5.
• Write your names on a piece of paper.

• How are you adding an element of chance to your Assignment #2?
### Additional Operators

- **Common operation: fetch/store same variable**
  
  ```
  var = var + 2;  // increment variable contents
  var = var * 2;  // double variable contents
  ```

  - operator/assignment combination (all ops supported):
    ```
    var += 2;
    var *= 2;
    ```

- **Pre/Post increment/decrement: ++ and --**
  
  - Example: `age++` vs. `++age`
```cpp
#include <iostream>    //library for input (cin) and output (cout)

using namespace std;

int main() {
    int age = 21;

    cout << age++ << endl;  //
    cout << age << endl;
    cout << ++age << endl;
    cout << age << endl;
    age++;  
    cout << age << endl;
    ++age;
    cout << age << endl;

    return 0;
}
```
Decisions in Life

• What is a decision?
• When do we make decisions?
• How do we make decisions?
  if it is sunny today
  \( \rightarrow \) then I’ll go to the beach and fly a kite
 else if it is raining today
  then I’ll stay inside and read a book
 else if it is snowing
  then I’ll go to the mountains to ski
Decisions within Decisions

• What happens if there is no wind at the beach?
• How does this change our decisions?

  if it is sunny today
  then I’ll go to the beach
  if it is windy at the beach
  then I’ll fly a kite
  else if it is not windy at the beach
  then I’ll walk on the shore
Flow chart for decisions

Start

Is it sunny?

Yes → Go to beach

No → Is it raining?

Yes → Read book

No → Go outside

Is it windy?

Yes → Fly kite

No → Walk on beach
Decisions in our programs

- Use an if/else
  
  if (<expression>) {
    <statement>;
    ...
    <statement>;
  }
  else {
    <statement>;
    ...
  }
What is the `<expression>`?

Could be a relational expression:

\[ <\text{expression}> \ <\text{relational op}> \ <\text{expression}> \]

- **Relational Ops**
  - `==` - equal to
  - `!=` - not equal to
  - `<` - less than
  - `>` - greater than
  - `<=` - less than or equal to
  - `>=` - greater than or equal to
C++ If/Else Syntax...

```cpp
if( x > y) {
    std::cout << "X is greater than Y";
}
else {
    std::cout << "X is less than Y";
}

• When does this logic fail?
C++ If/Else...

```cpp
if( x > y) {
    std::cout << "X is greater than Y";
}
else if( x < y) {
    std::cout << "X is less than Y";
}
else {
    std::cout << "X is equal to Y";
}
```
What are the curly braces for?

```cpp
if( x > y)
    std::cout << "X is greater than Y";
else if( x < y)
    std::cout << "X is less than Y";
else
    std::cout << "X is equal to Y";
```
What if we are testing for ==?

```cpp
if( x == 0) {
    std::cout << "X is zero";
}
else if( x == 1) {
    std::cout << "X is one";
}
else if( x == 2) {
    std::cout << "X is two";
}
else {
    std::cout << "You have entered an invalid number!";
}
exit(0);
```
```cpp
#include <iostream>  // library for input (cin) and output (cout)

using namespace std;

int main() {
    int age=21;
    cout << age++ << endl;  //
    cout << age << endl;
    cout << ++age << endl;
    cout << age << endl;
    age++;
    cout << age << endl;
    ++age;
    cout << age << endl;

    if(age==21) { // make sure you don't use an assignment op, =
        cout << "you can have a beer!" << endl;
    }
    else {
        cout << "wait a few more years!" << endl;
    }
    cout << age << endl;

    return 0;
}
```

-- INSERT --
```cpp
#include <iostream>  // library for input (cin) and output (cout)

using namespace std;

int main() {
    int age = 21;

    cout << age++ << endl;  //
    cout << age << endl;
    cout << ++age << endl;
    cout << age << endl;
    age++;
    cout << age << endl;
    ++age;
    cout << age << endl;

    if (age >= 21) {  // better to see if age is 21 or older
        cout << "you can have a beer!" << endl;
    }
    else {
        cout << "wait a few more years!" << endl;
    }

    cout << age << endl;

    return 0;
}
```
Logical Operators

• AND: if((1>2) && (2<5))
• OR: if((1>2) || (2<5))
• NOT: if(!(1>2) && (2<5))