Announcements

- gcc should now be set on turing

- Labs are due Sunday
  prereqs are due before class
generally, you can work in pairs
  (tell me your partner)

- HW1 is due 1 week from Sat.
Command line tips

In general, you'll use 5-6 commands the most.

- `ls` - list files in current directory
- `cp sourcefile targetfile`
- `mkdir name`
- `rmdir name`
- `cd directory name`
- `mv sourcefile targetfile`
- `rm`
Others

- vi or emacs or pico (nano)
- g++
- man
Tricks

- Hitting the up arrow gives the last thing you typed (so then you can edit)
- Hitting tab will auto-complete
- You can use & to get prompt back
  ex: kake file &
- `...` is current directory, `..` is parent
  ex: cd ..
  cp ../*.file .
Last time

- More C++:
  
  while loops
  
  if statements
  
  functions
Common error
What is wrong?

\[ a = b = c = 0 \]

```cpp
double gpa;
cout << "Enter your gpa: ";
cin >> gpa;
if (gpa == 4.0)
    cout << "Wow!" << endl;
```

==

In Python: compiler complaining

In C++ gpa is
result:
(+ it is true)
Do-while loops

```cpp
int number;
do {
    cout << "Enter a number from 1 to 10: ";
    cin >> number;
} while (number < 1 || number > 10);
```

- Executes body before checking the boolean
For loops

In python, iterator based.

Example:

```python
for (int count = 10; count > 0; count--):
    cout << count << endl;
    cout << "Blast off" << endl;
```

Note: int declaration isn't required.
The main function
Every program defaults to running a

```cpp
int main() {
    body;
    return 0;
}
```
Arrays

Python has lists, tuples, etc.
In C++, only have arrays.

- Size is fixed at declaration
- Type is fixed (4 homogeneous)

Ex: int numbers[5];
    numbers[0] = 55;
    numbers[4] = 10;

?? numbers[5] = 5;
Creating Arrays:

Allowed: `int daysInMonth = [31, 28, 31, 30, 31, 30, 31, 30, 31, 30, 31, 31];`

Error: `int daysInMonth [ ];`

Allowed: `char greeting[] = "Hello";`

Strings are char arrays.
Multidimensional arrays

```c
int table [8][10];
```

```c
for (int i = 0; i < 8; i++)
for (int j = 0; j < 10; j++)
    table[i][j] = 0;
```

\[
\sum_{j=1}^{m} \sum_{i=1}^{n} A_{ij} \geq 1
\]
### C++ has several predefined classes.

<table>
<thead>
<tr>
<th>Class</th>
<th>Purpose</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>istream</td>
<td>Parent class for all input streams</td>
<td>&lt;iostream&gt;</td>
</tr>
<tr>
<td>ostream</td>
<td>Parent class for all output streams</td>
<td>&lt;iostream&gt;</td>
</tr>
<tr>
<td>iostream</td>
<td>Parent class for streams that can process input and output</td>
<td>&lt;iostream&gt;</td>
</tr>
<tr>
<td>ifstream</td>
<td>Input file stream</td>
<td>&lt;fstream&gt;</td>
</tr>
<tr>
<td>ofstream</td>
<td>Output file stream</td>
<td>&lt;fstream&gt;</td>
</tr>
<tr>
<td>fstream</td>
<td>Input/output file stream</td>
<td>&lt;fstream&gt;</td>
</tr>
<tr>
<td>istream</td>
<td>String stream for input</td>
<td>&lt;sstream&gt;</td>
</tr>
<tr>
<td>ostream</td>
<td>String stream for output</td>
<td>&lt;sstream&gt;</td>
</tr>
<tr>
<td>stringstream</td>
<td>String stream for input and output</td>
<td>&lt;sstream&gt;</td>
</tr>
</tbody>
</table>
Using `iostream`
```
#include <iostream>
using namespace std; // omit this
```

Notes:
- can now use `cin` (for input) and `cout` (for output)
- separate distinct variables by `>` or `<` in `cin` and `cout`
- use `endl` for end of a line
  ```
Example

Python

```python
print "Hello"
print "Hello," first
print first, last  # automatic space
print total
print str(total) + "."  # no space
print "Wait...",     # space; no newline
print "Done"
```

C++

```cpp
1  cout << "Hello" << endl;
2  cout << endl;        // blank line
3  cout << "Hello, " << first << endl;
4  cout << first << " " << last << endl;
5  cout << total << endl;
6  cout << total << "." << endl;
7  cout << "Wait... ";   // no newline
8  cout << "Done" << endl;
```
Formatting output

cout << team << ": ranked " << rank << " of " << total << " teams" << endl;

- No '%d' here to easily format

Can set precision:

cout << "pi is " << fixed << setprecision(3) << pi << endl;

- Note that precision stays set to 3
Using `cin`

```cpp
int number;
cout << "Enter a number:" << number;
cin >> number;
```

**Note:**
- Inputs are separated by any white space.
- Type of input must match type of variable (not all strings).

```
10 20 2
10 20 2
```
One possible problem:

```cpp
string person;
cout << "What is your name?"; cin >> person;
```

I type "Erin Chambers".

What happens?

```
person = "Erin"
```
Getline

- getline is a function which saves the string up to (but not including) the next newline.

Ex:
```cpp
string person;
cout << "What is your name?";
getline (cin, person);
```
Another tricky example

```cpp
int age;
string food;
cout << "How old are you? ";
cin >> age;
cout << "What would you like to eat? ";
getline(cin, food);
```

I type: 15

Problem: age = 15
food = ""
Using File Streams - fstream

```c++
#include <fstream>
using namespace std;

if file is known:

    ifstream mydata("scores.txt");

if not:

    ifstream mydata;
    string filename;
    cout << "What file? ";
    cin >> filename;
    mydata.open(filename.c_str()); // parameter to open must be a C-style string
    mydata >> var;
```
ofstream

By default, writing to a file overwrites the file. (Think ‘w’ in Python.)

To append:

```cpp
ofstream datastream("scores.txt", ios::app);
```
Reading and writing

There is also an fstream object which allows reading and writing to a single file.

Much more complex.
String Streams

Ex: Cashing between numbers and strings.

```c++
int age(42);
string displayedAge;
stringstream ss;
ss << age;
ss >> displayedAge;
```