Announcements

- HW is up - due Saturday by midnight
- Lab 2 is posted - pre-lab before class tomorrow
- Presentation about on-campus opportunity
- Math/CS club meets Wed at 4pm
Example from last time

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

gpa = 4.0 returns true, not "true".

Why?

\[ a = (b = 0) \]

\[ y = a = a + 1; \]
Arrays

Python has lists, tuples, etc.

In C++, only have arrays.

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

Ex:
```c
int numbers[10];
numbers[0] = 55;
numbers[9] = 10;
numbers[10] = 5;  // In C++, seg fault
```

```
+---+---+---+---+---+---+---+---+---+---+
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
+---+---+---+---+---+---+---+---+---+---+
| 55 | 2659 | 2660 |
| 6261 |
```

Register address

```c
numbers[5] = 155;
```

+---+---+---+---+---+---+---+---+---+---+
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
+---+---+---+---+---+---+---+---+---+---+
| 55 |
| 2659 |
| 2660 |
| 6261 |
```
Creating Arrays:

Allowed:

\[ \text{int} \quad \text{daysInMonth} = \{ 31, 28, 31, 30, 31, 30, 31, 30, 31 \} \]

Error: \[ \text{int} \quad \text{daysInMonth} \]

no size

Allowed:

\[ \text{char} \quad \text{greeting} = \text{"Hello"} \]

must be a char!

reason: 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] = i + j;
```
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>istringstream</td>
<td>String stream for input</td>
<td>&lt;sstream&gt;</td>
</tr>
<tr>
<td>ostringstream</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 10 streams

```cpp
#include <iostream>
using namespace std;
```

(useful for debugging)

```cpp
std::cin
```

(just like printf or raw-input)

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

Python

```python
print "Hello"
print
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

\begin{verbatim}
  cout << team << " : ranked " << rank << " of " << total << " teams" << endl;
\end{verbatim}

- No `%d` here to easily format
Can set precision:

\begin{verbatim}
  cout << "pi is " << fixed << setprecision(3) << pi << endl;
\end{verbatim}

π is 3.141

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

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

**Note:**
- Inputs are separated by any white space
- `cin >> a >> b` (not `cin >> a, b`)

- Type of input must match type of variable (not type of string)
One possible problem:

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

I type "Erin Chambers".
What happens?

```cpp
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);
Erin _ Chambers
```
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, p1za9
```

Problem:

```
age = 15
food = ""
```
Using File Streams - fstream

```cpp
#include <iostream>
#include <fstream>
using namespace std;

if file is known:
    ifstream mydata("scores.txt");
    mydata >> variable;  // mydata >> variable

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
```

Converts to C-style string