CS180 – C++ & the command line

Note Title 9/2/2011

Announcements

- HW1 is posted – due next Sat.
  (individual this time)
- Lab tomorrow
  (& prelab due before class tomorrow)

Command line tips – google UNIX tutorial/tips

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

- ls – list
- cp sourcefile targetfile
- mkdir name
- rmdir name
- cd directoryname (change directory)
- mv sourcefile targetfile
- rm file ← careful!

Others

- vi or emacs or ~~pico~~ nano
- g++
- man

Tricks

- Hitting the up arrow gives the last thing you typed
  (& then you can edit)
- Hitting tab will autocomplete
- You can use & to get prompt back
  ex: kak file &
- . is current directory, .. is parent

  ex: cd ..
  cp ../file .

Last time

- loops
- if
- functions

Today: input / output

Common error

What is wrong?

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

==

$x = y = (z = 2);$

(gpa = 4.0)

↓

4.0

Do-while loops

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

- Executes body before checking the boolean

```
while ( ) {

}
```

The main function

Every program defaults to running a main.

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

Arrays

Python has lists, tuples, etc.

In C++, only have arrays.

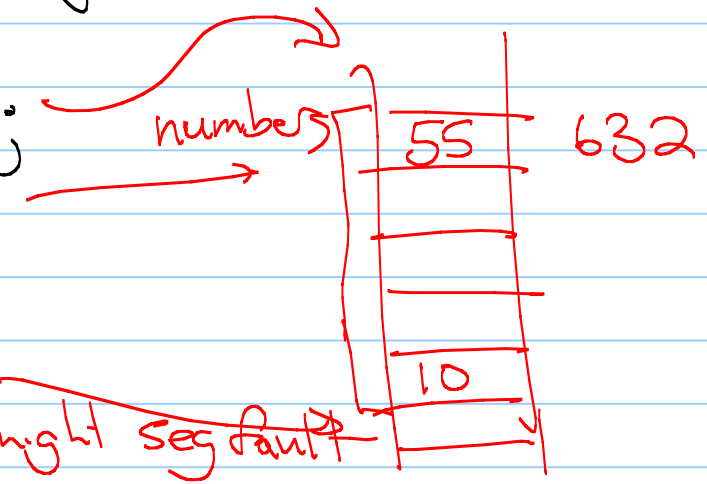
- Size is fixed at declaration
- type is fixed (& homogeneous)

Ex:

```
int numbers[5];
numbers[0] = 55;
numbers[4] = 10;

numbers[5] = 5;
```

↳ might work, might seg fault



Creating Arrays:

Allowed:

```
int daysInMonth = {31, 28, 31, 30, 31, 30,
                   31, 31, 30, 31, 30, 31};
```

Error:

```
int daysInMonth[];
```

must specify a size

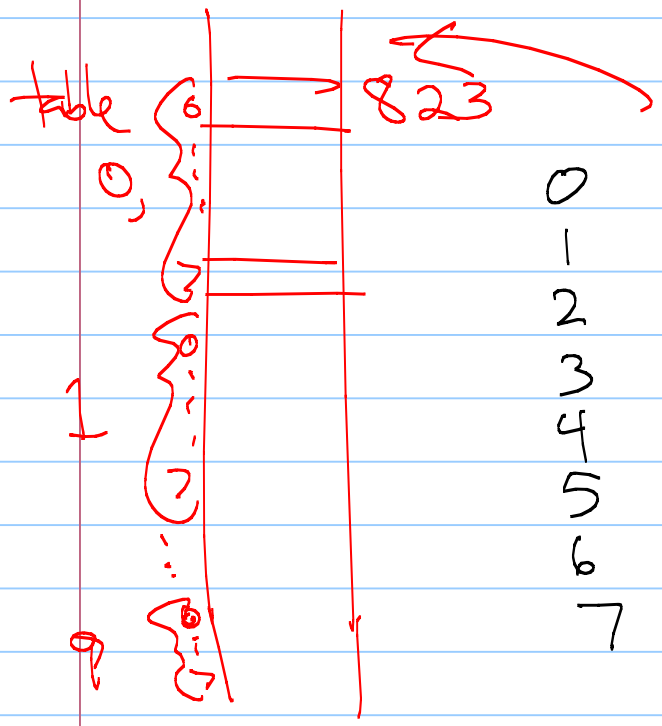
Allowed:

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

Multi-dimensional arrays

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

int table [8] [10];



	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4					
1	1	2	3	4	5					
2	2	3	4	5						
3										
4										
5										
6										
7										

#include <iostream>

Input & Output

C++ has several predefined classes.

Class	Purpose	Library
istream	Parent class for all input streams	<iostream>
ostream	Parent class for all output streams	<iostream>
iostream	Parent class for streams that can process input and output	<iostream>
ifstream	Input file stream	<fstream>
ofstream	Output file stream	<fstream>
fstream	Input/output file stream	<fstream>
istringstream	String stream for input	<sstream>
ostringstream	String stream for output	<sstream>
stringstream	String stream for input and output	<sstream>

(new versions)

Using iostream

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

X

std::cin

Notes:
- can now use cin (for input) + cout (for output)
- separate distinct variables by >> or <<
- use endl for end of a line
- "using namespace std" is (sort of) optional

Example

Python

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

C++

```
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

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

Note: - inputs are separated by any white space

```
cin >> a >> b;
```

5 → 6 ↵

5↵

6↵

- type of input must match type of variable (not all strings)

One possible problem:

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

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:

```
string person;
cout << "What is your name?";
getline (cin, person);
```

Another tricky example

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

I type: 15 ↵
hot dogs ↵

Problem: age = 15
food = \n or ""

Using File Streams - fstream

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

if file is known:

```
int score;
ifstream mydata("scores.txt");
mydata >> score;
```

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

ofstream

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

To append:

```
ofstream datastream("scores.txt", ios::app);
```

Reading and writing

There is also an fstream abject which allows reading & writing to a single file.

Much more complex.

String Streams

Ex: Casting between numbers & strings.

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

A note on variable scopes:

```
int main () {
    int a;
    if (a > 0)
        int b = 12;
    else
        int b = 16;

    cout << "a is " << a << endl;
    cout << "b is " << b << endl;

}
```