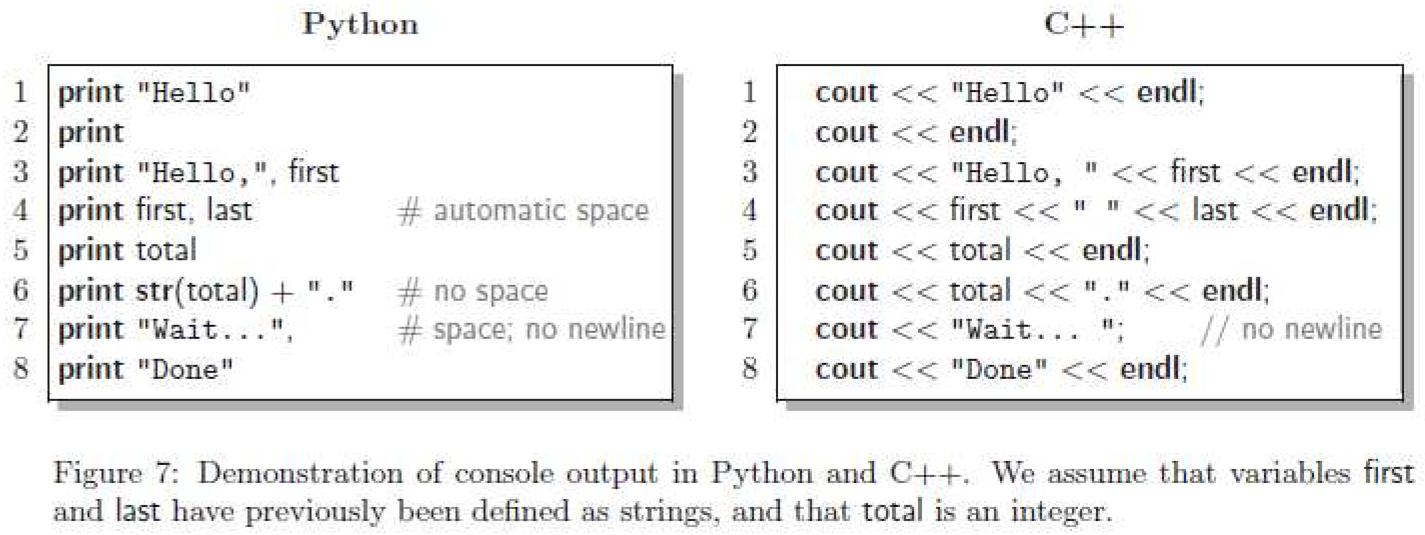
CS180 - More C++

Note Title 1/19/2011

Announcements

- First lab is tomorrow
  (Prelab is due before 10am)
- Office hours today 1:30 - 3:30

Examples



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++

```
cout << "Hello" << endl;
cout << endl;
cout << "Hello, " << first << endl;
cout << first << " " << last << endl;
cout << total << endl;
cout << total << "." << endl;
cout << "Wait... ";        // no newline
cout << "Done" << endl;
```

Figure 7: Demonstration of console output in Python and C++. We assume that variables first and last have previously been defined as strings, and that total is an integer.

Formatting output

Unfortunately '%d' output is not really available
↖ # of digits

(Inherited from C, so there, but can't be used with C++ objects like strings.)

Python

```
print '%s: ranked %d of %d teams' % (team, rank, total)
```

C++

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

Setting precision is harder:

```
print 'pi is %0.3f' % pi
```

output?

```
pi is 3.141
```

In C++:

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

Note: Precision stays set to 3.

Cin : Other data types (not strings)

Python :

```
number = int(raw_input('Enter a number from 1 to 10: '))
```

C++ : cin >> number;

```
int number;
cout << "Enter a number from 1 to 10: ";
cin >> number;
```

Note : cin looks for whitespace

cin << a << b;

```
10   20
-----------  same
10
20
```

Input: Strings

Python: raw_input

✓
```
person = raw_input('What is your name?)
```

C++: getline

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

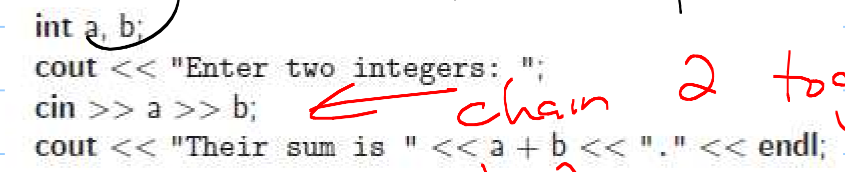
↑ does not work for anything except strings

Note (for getline):

- inputs a <u>string</u>
- stores up to the newline, but strips the newline off

Some other diffences with cin :

Chaining multiple inputs



```
int a, b;
cout << "Enter two integers: ";
cin >> a >> b;
cout << "Their sum is " << a + b << "." << endl;
```

← chain 2 together

do arithmetic operations

<u>Note:</u> - different types are allowed (but must match the variable)

- separated by <u>any</u> whitespace!

A word of caution:

Ex:

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

I type "Erin Wolf Chambers /n".

What happens?

person = "Erin"

use getline!

Another caution:

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

30 40 50 60 ↵

30
pepperoni␣pizza

age = 30
food = " "

30␣pepperoni␣pizza

age = 30
food = "␣pep... "

File Streams: Input

If file name is known:

```
ifstream mydata("scores.txt");
```

← declares + opens an input file

If file name is unknown:

```
ifstream mydata;
string filename;
cout << "What file? ";
cin >> filename;
mydata.open(filename.c_str( ));
```

↑ converts to a C-type string (historical legacy)

Output:

By default, opening ofstream overwrites an existing file!

(just like "w" option in Python)

To append:

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

normally get deleted

"a" in Python

fstream

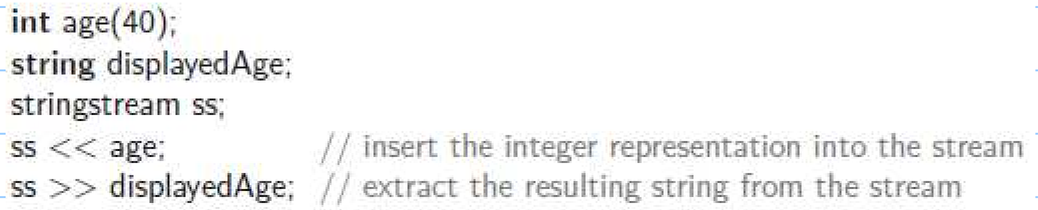
There is also an "fstream" object, which allows both input & output.

Much more confusing.

(Whenever possible, much safer to keep input & output separate.)

String Streams

Casting from numbers to strings is not straightforward.



```
int age(40);
string displayedAge;
stringstream ss;
ss << age;          // insert the integer representation into the stream
ss >> displayedAge; // extract the resulting string from the stream
```

Classes

What is a class?

- Way to store information in your own objects

Ex: Credit Card
related collection of data

- to make life easy
- limit (& define) functionality
- access control

Classes

Creating an instance of a class

```
string s;
string greeting("Hello");
```

input parameters to constructor

NEVER: `string s( );`

Why? Create a function called s that returns a string

NEVER: `string("Hello") greeting;`

Why? Compile error

Defining a class: Remember the Point class?

```
class Point {
  private:
    double _x;                    // explicit declaration of data members
    double _y;

  public:
    Point( ) : _x(0), _y(0) { }   // constructor

    double getX( ) const {        // accessor
      return _x;
    }

    ~~void setX(double val) {~~       // mutator
      ~~_x = val;~~
    ~~}~~

    double getY( ) const {        // accessor
      return _y;
    }

    void setY(double val) {       // mutator
      _y = val;
    }

};                                // end of Point class (semicolon is required)
```

data in the class

Point mypoint;

cout << mypoint._x << endl;

ERROR

mypoint.getX()