CS2100 - C++ & the command line

Note Title 9/2/2011

Announcements

(on board)

Command line tips - google "unix command line tutorial"

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

- ls
- cp sourcefile targetfile
- mkdir name
- rmdir name
- cd directory name
- mv sourcefile targetfile

- rm ⟵ Careful!

<u>Others</u>

- vi or emacs or pico/nano
- g++
- man - manual pages

  ```
  > man ls
  ```

Tricks

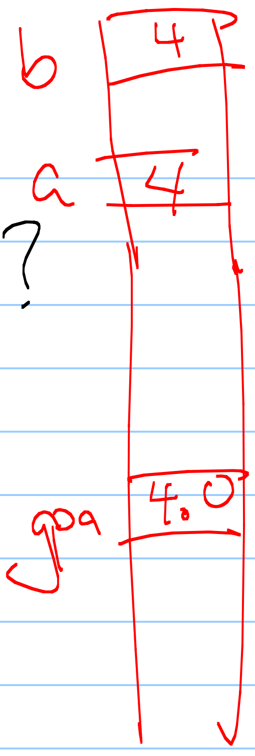
- Hitting the up arrow gives the last thing you typed (& then you can edit)
- Hitting tab will auto complete
- You can use & to get prompt back
  ex: kak file &
- . is current directory, .. is parent
  ex: cd ..
      cp ../file .

Common error

What is wrong?

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

BAD!



$a = \underbrace{b = 4}_{4};$

$c = x + 2 - z;$

$c = x + 6 = 4;$

Defining a function : example

Remember countdown function from ~~150~~ 1300?

no input variables

```
void countdown( ){
  for (int count = 10; count > 0; count--)
    cout << count << endl;
}
```

no return

Optional arguements

```
void countdown(int start=10, int end=1) {
   for (int count = start; count >= end; count--)
      cout << count << endl;
}
```

if start is
- declared
set start's value
to 10

often empty

true ?

```
for ( ; ; )
{
  stuff;
}
```

If statements

```
if (bool) {
    body1;
}
else {
    body 2;
}
```

Ex:

```
if (x < 0)
  x = -x;
```

```
if (groceries.length( ) > 15)
  cout << "Go to the grocery store" << endl;
else if (groceries.contains("milk"))
  cout << "Go to the convenience store" << endl;
```

Note: - Don't need brackets if 1 line
- don't need else
- no elif

So nesting can get ugly!

```
if  ( 1 )
if  ( 2 )
  {code}
else
  {   }
if ( 3 )
if (     )
if (     )
  {   }
else
  {  }
```

```
if  (1)
   if (2)
      {code}
   else
      {
if (3)
      }
```

Booleans & if/whiles

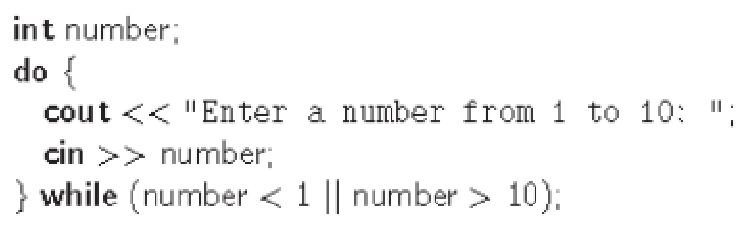
If & while statements can be written with numeric conditions (which are really booleans).

Ex:
```
if (mistakeCount)
    cout << "Error!" << endl;
```

$0 \Longleftrightarrow$ false

anything not zero is true!!

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

The main function

Every program defaults to running a main.

no input variables

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

Arrays

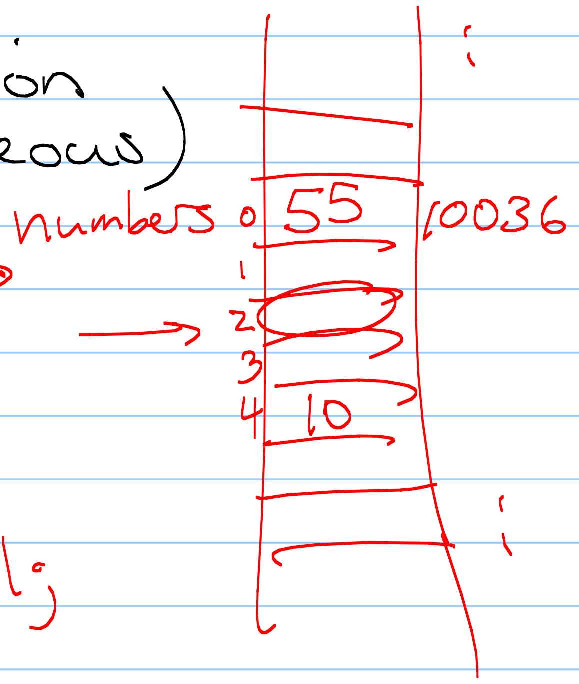
Python has lists, tuples, etc.

In C++, only have arrays.

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

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

cout << number[2] <<endl;
```

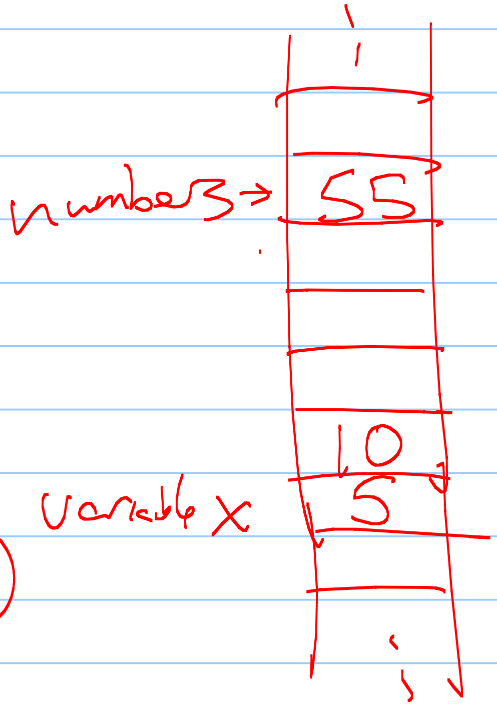


Careful of segfaults!

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

numbers[5] = 5;
```

```
for (int i=0; i<size; i++)
    numbers[i] = 0;
```



Creating Arrays:

__Allowed:__

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

__Error:__

```
int daysInMonth [ ];
```

must specify size

__Allowed:__

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

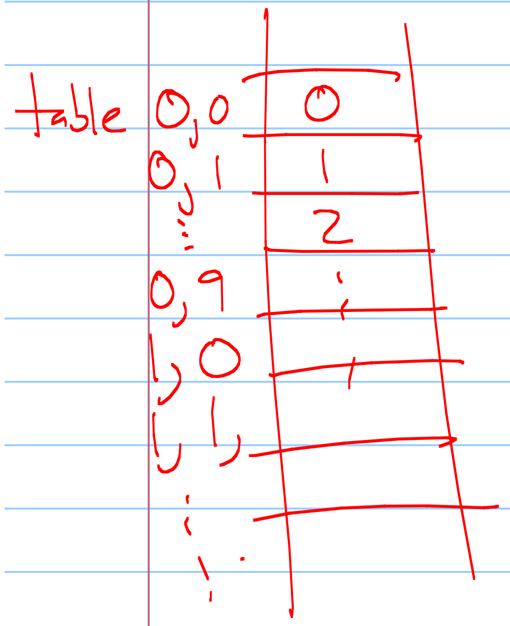
one exception

really same as

Multi-dimensional arrays

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

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



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

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>

Using iostream

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

optional, but say:
std::cin
+ std::cout

Notes:

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

Example

```
x = y = z = 1;
cout << x << y << z << endl;
```

```
>111
```

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++

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

<u>Using cin</u>

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

<u>Note</u>: - inputs are separated by any white space

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

```
> 10 11 2
```

```
> 10 2
> 11 2
```

- type of input must match type of variable (<u>not</u> all strings)

One possible problem:

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

I type "Erin Chambers".
What happens?

```
person = "Erin"
```

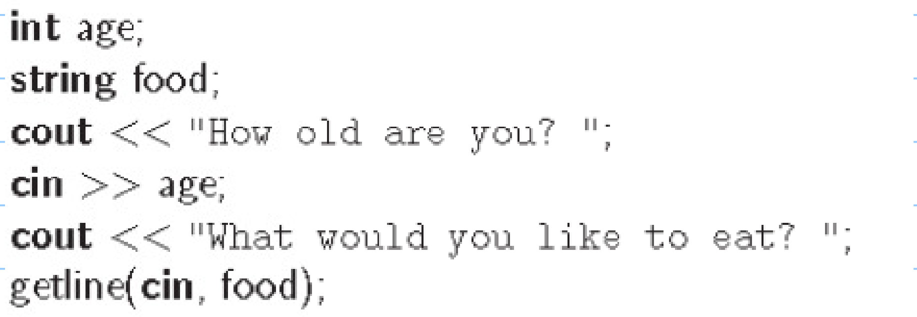
Getline

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

<u>Ex</u>:

```
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:

Using File Streams - fstream

```
# 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
```

ofstream

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

To append:

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

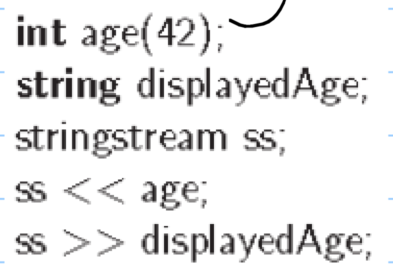
Reading and writing

There is also an fstream object, 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;
}
```