CS 142
More C++

Rhodes College

Announcements

- Reminder
  - Program 4 due 3/17 by 11:55pm
- Quiz on C++ basics on Wednesday

Getting Input

- Use cin keyword

```
cin >> x >> y;
```

- You need to write your own prompt with cout.
- Cin can input multiple things at the same time.
- With strings, only reads one word at a time, not a whole line.
- Like Python, your program MIGHT crash/MIGHT keep running with bad data if you try to cin an int, but type a float.

C++ Demo Program

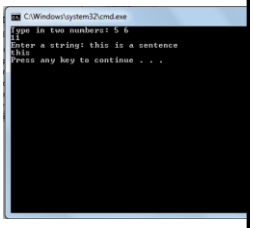
```
#include <iostream>
#include <string>

using namespace std;

int main()
{
    int x, y;
    string s;
    cout << "Type in two numbers: ";
    cin >> x >> y;
    cout << x + y << endl;


    cout << "Enter a string: ";
    cin >> s;
    cout << s << endl;

    return 0;
}
```



Type Casting

- While you can type-cast like Python with:
  `a = float(b);`
- Newer compilers support a different syntax, so we will use the new, preferred syntax for C++ type conversion

```
int b, e;
double c, d; //a double is a big float
b = 3;
c = 2.5;
c = b + c; //automatically upgrades b to a double, c holds 5.5
d = static_cast <double> (b); //d holds 3.0
e = static_cast <int> (c); //e holds 5
```


Integer Division

- Remember that if you divide 2 integers, you will get an integer, even if you declare the variable as a float

```
int a = 9, b = 4;
float div;

div = a / b; //div = 2
//must upgrade at least 1 of a or b to a float

div = static_cast <float> (a) / b;      //div = 2.25
```


Comments in C++

- In-line comments or 1 line comments

```
// this is a 1-line comment
```

- Multiple-line comments

```
/* This is a multi-line
Comment. I can keep typing and all this will
Be treated like a comment until I end this
block
*/
```


Practice

- Calculate the total cost of a restaurant per person (assume dividing equally)
- Prompt the user for the total bill, the percent tip (as a decimal) and the number of people in the group
- Output the total cost per person

Decision Statements (if, else if, else)

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

int main()
{
  int x = 5, y = 3;

  if (x < y)
    cout << "x is less ";
    cout << "than y\n";
  }
  cout << "the end\n";
  return 0;
}
```

( ) are required around outside of condition

Decision Statements (if, else if, else)

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

int main()
{
  int x = 5, y = 3;

  if (x < y) {
    cout << "x is less ";
    cout << "than y\n";
  }
  cout << "the end\n";
  return 0;
}
```

Required if more than 1 line of code is inside the if block

Recall that { } braces are used to mark blocks of code and are used to group lines of code together


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

int main()
{
  double grade;

  cout << "Enter your grade average (i.e., 93.2): ";
  cin >> grade;

  if (grade >= 90.0) {
    if (grade > 92.0) {
      cout << "Your grade is an A\n";
    }
    else {
      cout << "Your grade is an A-\n";
    }
  }
  else if (grade >= 80.0) {
    if (grade > 87.0) {
      cout << "Your grade is a B+\n";
    }
    else if (grade > 82.0) {
      cout << "Your grade is a B\n";
    }
    else {
      cout << "Your grade is a B-";
    }
  }
  return 0;
}
```

No more elif, must write out else if

Python	C++
and	&&
or	||
not	!

Python

```
if  (x < 3) and not (y > 4):
   print(x, y)
```

C++

```
if ((x < 3) && !(y > 4)){
   cout << x << " " << y << endl;
}
```

All relational operators (<=, >=, <, >, !=, ==) are the same.

Just like in Python, C++ will not mark `if (x = 0)` as an error – it will allow it, so be aware what you are really saying is `x = 0; if(0)`

While Loops

- Same differences as if statements
  - Parenthesis around the boolean expression (condition)
  - Boolean operators are &&, ||, and ! (and, or, not)
  - Curly braces { } are used to denote which lines of code are part of the loop body (not required if loop body is only 1 line of code long)

```
int i = 0;
while (i < 10) {
  cout << i << endl;
  i += 1;            //i++; is equivalent to this (postfix)
                     //can also use ++i; (prefix)
}
```

Do-While Loops

New – does not have a Python equivalent!

A while loop is classified as a ***pretest loop***, meaning that you check the condition before ever executing the loop

A do-while loop is a ***posttest loop***, in that you execute the body of the loop at least once and then you check the condition

```
int i = 0;
do {                        //loop body executed at least once
  cout << i << endl;        //loop body
  i += 1;                   //loop body
} while (i < 10);           //boolean expression
```


For Loops

No more range() or iterating over items in a list or lines in a file

Increment (step value), you can change this to i+=5 to count by 5s, or some other number

```
int i;
for (i=0; i<10; ++i) {
    cout << i << endl;
}
//by declaring i before the loop, I have access to the
value of i here
//However, I can also do this
for (int i=0; i<10; ++i) {
    cout << i << endl;
}
//Now I do not have access to the value of i here (outside
the scope) (scope is determined by { })
```


For Loops

You can use floating point numbers in for loop

```
float i;
for(i = 0; i < 10; i+=0.1){
   cout << i << endl;
}
```

```
Output:   1.2     2.5     3.8     5.1     6.4     7.7     9
0         1.3     2.6     3.9     5.2     6.5     7.79999 9.1
0.1       1.4     2.7     4       5.3     6.6     7.89999 9.2
0.2       1.5     2.8     4.1     5.4     6.7     7.99999 9.3
0.3       1.6     2.9     4.2     5.5     6.8     8.09999 9.4
0.4       1.7     3       4.3     5.6     6.9     8.2     9.5
0.5       1.8     3.1     4.4     5.7     7       8.3     9.6
0.6       1.9     3.2     4.5     5.8     7.1     8.4     9.7
0.7       2       3.3     4.6     5.9     7.2     8.5     9.8
0.8       2.1     3.4     4.7     6       7.3     8.6     9.9
0.9       2.2     3.5     4.8     6.1     7.4     8.7
1         2.3     3.6     4.9     6.2     7.5     8.8
1.1       2.4     3.7     5       6.3     7.6     8.9
```

Practice

Write a while loop to let the user enter numbers from the keyboard until they type zero. Print the average of all the numbers at the end. Ignore all negative numbers.

To ignore all negative numbers, use the continue command

In C++ it is:

```
continue;
```

When you've finished that, add a for loop at the end that prints out all numbers 2, 4, 6, ….up to 50. 50 should be printed. Put each number on a separate line.

*You need to create a New Project like we did on Friday. You cannot have 2 cpp files in the same project with main functions