CIS 130 - Intro to Programming
Week 11 Lab - Wednesday, 04-04-07
Week 11 Lab Exercise

Purpose: get some practice with simple functions in C++ (and the ~st10/funct_play2 and ~st10/expr_play tools)

YOU MAY WORK IN PAIRS FOR THIS LAB. Make sure you understand all the concepts!

NAME(S) ____

1. Goal #1: see how you can use the **~st10/funct_play2** and **~st10/expr_play** tools on cs-server to write and test simple functions written in C++.

As was demonstrated in C++, if you type the command **~st10/funct_play2** at the cs-server prompt, it walks you through the steps of the usual design recipe, except that you are expected to type your code using C++ syntax instead of Python syntax.

As a reminder, consider the following versions of **circ_area**, one in Python and one in C++:

```
# contract: circ_area: number -> number
# purpose: compute and return the area for a circle whose radius is <radius>
# examples: circ_area(10) == 314.159
#           circ_area(5) == 78.53975

PI = 3.14159

def circ_area(radius):
    return PI * (radius * radius)

/*-----
  contract: circ_area: double -> double
  purpose: compute and return the area of a circle whose radius is <radius>
  examples: circ_area(10) == 314.159
            circ_area(5) == 78.53975
-----*/

const double PI = 3.14159;

double circ_area(double radius)
{
    return PI * (radius * radius);
}
```

Try typing in and testing the C++ version of **circ_area** using **~st10/funct_play2**

2. You don't have to re-type a function if you need to debug it or otherwise modify it. If you now type the following at the cs-server command line, you'll see that you have three **circ_area** files:

```
ls circ_area*        # show names of files that begin with circ_area

circ_area.cpp
circ_area.h
circ_area.o
```

...you can simply use **pico circ_area.cpp** or **pico circ_area.h** and make whatever changes you'd like. If you call **~st10/funct_play2** again, and answer the "new function" questions so that you say the file already exists, you can edit it further as desired and re-compile the changed function.

(You can also re-compile your function at the cs-server prompt, IF you prefer, by typing:

g++ -c circ_area.cpp

...and then you can just use ~st10/expr_play to run the changed function, if you prefer that to **~st10/funct_play2**)

Using the method of your choice, CHANGE the precision of **PI** for **circ_area**. (Which file is that declaration in, **circ_area.cpp** or **circ_area.h**?) Re-run **circ_area** and verify that you see the new precision. When it is your turn, I will ask you to run your **circ_area .**

3. What if you are writing a function that USES another function? Then, in the **~st10/funct_play2** tool, you specify what functions the new function uses before writing the new function --- the tool then inserts the needed code to make this possible (which we'll discuss a bit later).

To see this, write a C++ version of **ring_area** that uses the C++ version of **circ_area** you have developed and modified. (Remember? The area of a ring is the area of the outer circle minus the area of the ring's "hole"... 8-))

Use either **~st10/expr_play** or **~st10/funct_play2** to test **ring_area**, but BEWARE of the following QUIRK: expr_play needs you to list ALL the functions involved with a function, to be able to run it (for reasons we will discuss later on). So, you need to enter the names of both **circ_area** and **ring_area** to be able to test **ring_area**.

When it is your turn, I will ask you to run your **ring_area**.

NOW write your name(s) on the **NEXT:** list on the board. (You write your name on this list if you have questions along the way, as well as when you are done; I'll then work my way down the list.) You need to complete the above and have it checked by the end of lab.