CIS 130 - Intro to Programming - Fall 2005
Homework Assignment #2

Homework #1 DUE**: 12:00 NOON**, Friday, September 16, 2005
Week "3" Lab Exercise - due at end of your registered lab section
on either 9-9 or 9-12

WEEK "3" LAB EXERCISE

1. Follow these steps to obtain and set up a copy of **funct_play0** in your directory:

a. Use ssh to connect to cs-server, and log onto your cs-server account (remember to use your NHW 244 password).

b. While in your home directory (where you are when you first log in), do the following commands, all at the cs-server prompt (explanation is in Times New Roman font, commands you are to type are in `Courier New`):

Copy **funct_play0** from my account into your **bin** directory:
```
cp  ~st10/bin/funct_play0  bin
```

Make your new copy of **funct_play0** executable:
```
chmod  700  bin/funct_play0
```

c. Now try typing **funct_play0** at the cs-server prompt, followed by typing the enter key. If you see the tool's greeting, you are set!

(If you don't --- try logging off, then on again, and try again. If that doesn't work, ask for help!)

Remember to answer **n** (for no) when asked if you are creating a function that doesn't use other functions, and answer **y** (for yes) when asked if you are creating a function that does use other functions.

2. Demonstrate that you are able to use your new **funct_play0** --- use it to enter a function **disk_area**, which returns the area of a circular disk given its radius.

3. And, try a function that calls another function --- use **funct_play0** to enter a function **ring_area**, which returns the area of a ring given the radius of the ring and the radius of the ring's hole, with the additional requirement that your new function must do this by calling the function **disk_area**.

4. Evaluate the following expressions in **expr_play**, one at a time:

```
10 . 20
10 20 +
+ +
```

Read the resulting error messages; note that, sadly, compilers do not always produce the most human-readable error messages!

5. Enter the following functions, one by one, into **funct_play0**:

```
halve (int number)
{
    return number / 2.0;
}

double halve (int number)
    return number / 2.0;
}

double halve {int number}
{
    return number / 2.0;
}

double halve (number)
{
    return number / 2.0;
}
```

Read the resulting error messages, then fix **halve** so that it is correct and compiles, and use **funct_play0** or **expr_play** (your choice!) to run it with at least two different arguments.

When you have completed the above, put your name on the **Next:** list so that I can check your work. The above must be completed, and your work checked, by the end of your registered lab section.

HOMEWORK #2

You are to work **individually** on this assignment.

We are in the unusual part of the semester in which you are using tools that automatically send me information, so running **funct_play0** successfully from cs-server will automatically "turn in" your work. Enjoy this while it lasts... 8-)

PLEASE NOTE: You will be reading about a **program design recipe** in the reading for next week --- you are **not** required to use this for this homework (HW #2), but you **will** be required to do so for HW #3 (and beyond).

(Of course, if you WANT to use the program design recipe while developing these, that's great!)

Problems **1-7** make use of the CIS 130 - Reading Packet, pp. 9-11. Note that when these problems say a function "consumes" some value, they mean that the function being described expects that value as a parameter.

1. Do Exercise **2.2.1** in the CIS 130 - Reading Packet Use **funct_play0** to enter your function, and use **funct_play0** or **expr_play** (your choice) to test your function by calling it with at least the following values:

212 (which should result in a Celsius temperature of 100)
32 (which should result in a Celsius temperature of 0)
68 (which should result in a Celsius temp of around 20)
and at least one additional, different Fahrenheit temperature of your choice.

2. Do Exercise **2.2.2** in the CIS 130 - Reading Packet. (Note that you could also look up the current exchange rate on the Web, also.) Use **funct_play0** to enter your function, and use **funct_play0** or **expr_play** (your choice) to test your function by calling it with at least two different dollar amounts.

3. Do Exercise **2.2.4** in the CIS 130 - Reading Packet. Use **funct_play0** to enter your function, and use **funct_play0** or **expr_play** (your choice) to test your function by calling it with at least three different sets of values, one of which must be (1, 2, 3) (and thus should result in 321)

4. Do Exercise **2.2.5** in the CIS 130 - Reading Packet, using **funct_play0**. (If you end up doing some of the required calls using **expr_play**, that is all right, too.)

5. Do Exercise **2.3.1** in the CIS 130 - Reading Packet, using **funct_play0** to enter your resulting functions and **funct_play0**/**expr_play** to test them on at least two different sets of arguments. (Be sure to call other functions as appropriate within function definitions.)

Note: "gross pay" means the pay before taxes have been deducted. The function **tax** expects the gross pay as its parameter; the function **netpay** expects the number of hours that an employee has worked as its parameter.

6. Do Exercise **2.3.2** in the CIS 130 - Reading Packet, using **funct_play0** to enter your resulting functions and **funct_play0**/**expr_play** to test them on at least two different sets of arguments. (Be sure to call other functions as appropriate within function definitions.)

7. Do Exercise **2.3.3** in the CIS 130 - Reading Packet, using **funct_play0** to enter your resulting functions and **funct_play0**/**expr_play** to test them on at least two different sets of arguments. (Be sure to call other functions as appropriate within function definitions.)

8. Develop a function **minutes_to_hours**, that takes a number of minutes and converts it to a number of hours, using **funct_play0** to enter your resulting function and **funct_play0/expr_play** to call it with at least the following arguments:

with at least one argument less than 60,
with at least one argument between 60 and 120,
with at least one argument greater than 120.