LAB DIRECTIONS for LAB 3/5/2015

Based on Lesson Set 6.1

1. Create a folder named **Lesson Set 6.1**. Put all files you create for this lab in this folder.

2. Download the **Lesson Set 6.1 source files** from ilearn and extract them into your Lesson Set 6.1 folder on your computer or jump drive.

3. Bring in program **`newproverb.cpp`** from the Lesson Set 6.1 source files folder.
   a. Some people know this proverb as "Now is the time for all good men to come to the aid of their country" while others heard it as "Now is the time for all good men to come to the aid of their party." This program will allow the user to choose which way they want it printed. Fill in the blanks of the program to accomplish what is described in the program comments. Run the program with a float such as -3.97 and see what happens.

   b. Change the program so that an input of 1 from the user will print "party" at the end, a 2 will print "country" and any other number will be invalid so that the user will need to enter a new choice.

Sample Run:

```
Given the phrase:
Now is the time for all good men to come to the aid of their __
Input a 1 if you want the sentence to be finished with party
Input a 2 if you want the sentence to be finished with country
Please input your choice now
4
I'm sorry but that is an incorrect choice; Please input a 1 or 2
2
Now is the time for all good men to come to the aid of their country
```

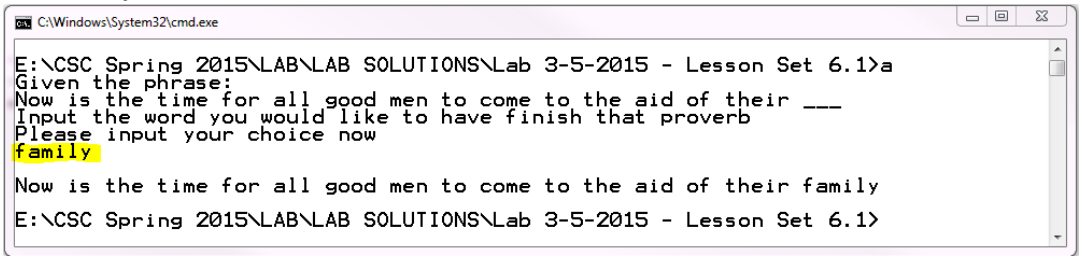
   c. Change the previous program so the user may input the word to end the phrase. The string holding the user's input word will be passed to the proverb function instead of passing a number to it. Notice that this change requires you to change the proverb function heading and the prototype as well as the call to the function.

Sample Run:

```
Given the phrase:
Now is the time for all good men to come to the aid of their _____
Please input the word you would like to have finish the proverb
family
Now is the time for all good men to come to the aid of their family
```

   d. Compile and run the program entering the word **family** for input.

   e. Take a screen capture of the program running and paste in a word document titled **Lesson Set 6.1 Screen Captures**.



4. Bring in the program **paycheck.cpp** from the Lesson Set 6.1 source files folder.
    a. Fill in the code (places in bold) and note that the function `computePaycheck` determines the net pay by subtracting 15% from the gross pay. Both `gross` and `net` are returned to the `main()` function where those values are printed.

    b. Compile and run your program with the following data and make sure you get the output shown.

```
Please input the pay per hour
9.50
Please input the number of hours worked
40

The gross pay is $380
The net pay is $323
We hoped you enjoyed this program
```

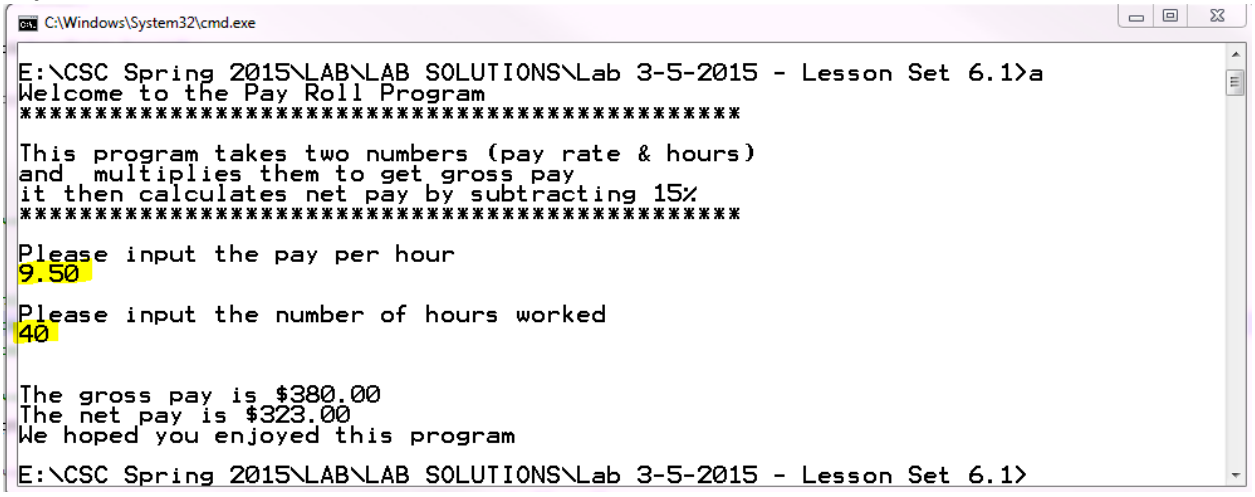
    c. Alter the program so that `gross` and `net` are printed in the function `computePaycheck` instead of in `main`(). The `main`() function executes the statement

```
cout << "We hoped you enjoyed this program" << endl;
```

    after the return from the function `calPaycheck`.

    d. Compile and run the program again using the same data as in b. – 9.50 for the pay per hour and 40 for the number of hours worked. You should get the same results. All parameters should now be passed by value.

    e. Take a screen capture of the program running and paste in the document titled **Lesson Set 6.1 Screen Captures**.

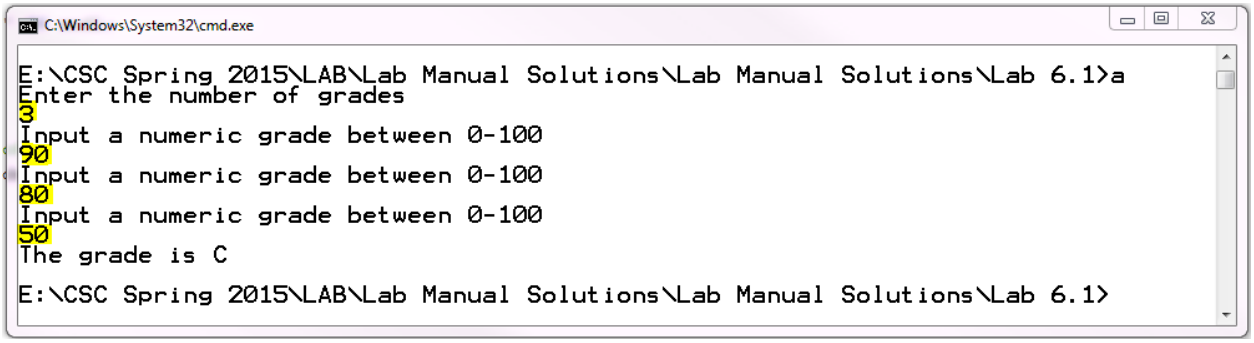


5. The next program you will write from scratch – there is no code to bring in from the source files folder. Write a program that will read in grades, the number of which is also input by the user. Name your program **grade.cpp**.
    a. The program will find the sum of those grades and pass it, along with the number of grades, to a function which has a "pass by reference" parameter that will contain the numeric average of those grades as processed by the function. The `main` function will then deter- mine the letter grade of that average based on a 10-point scale.

90–100	A
80–89	B
70–79	C
60–69	D
0–59	F

    b. Compile and run the program again using the data in the example screen capture below.

    c. Take a screen capture of the program running and paste in the document titled **Lesson Set 6.1 Screen Captures**.



What to Turn In: (by Wednesday, March 18, 2015)

- newproverb.cpp
- paycheck.cpp
- grade.cpp
- Lesson Set 6.1 Screen Captures

How you will be graded

newproverb.cpp	5 points	Program compiles with no errors
	5 points	Program code is neat & includes complete comment block
	10 points	Follows specifications / correct calculations
	5 points	Neat output
paycheck.cpp	5 points	Program compiles with no errors
	5 points	Program code is neat & includes complete comment block
	10 points	Follows specifications / correct calculations
	5 points	Neat output
newproverb.cpp	5 points	Program compiles with no errors
	5 points	Program code is neat & includes complete comment block
	20 points	Follows specifications / correct calculations
	5 points	Neat output
Lesson Set 6.1 Screen Captures	15 points	Screen capture for all three programs was included (5 points each)