Lab 6

In order to get credit for the lab, you need to be checked off by the end of lab. You can earn a maximum of 3 points for lab work completed outside of lab time, but you must finish the lab before the next lab. For extenuating circumstances, contact your lab TAs and Jennifer Parham-Mocello.

(2 pts) Worksheet

You can complete this section individually or in pairs

Let's take a minute to revisit the second part of Lab 5. Suppose our main function is as follows:

```
int main() {
   string sentence;

   get_sentence(sentence);
   cout << sentence << endl;

   return 0;
}
```

Based on the function call above, notice that get_sentence() is a void function because we don't do anything with the function call (it is on a line by itself). It has a string as an argument in the function call (between the parenthesis is sentence). Therefore, the prototype for this function is as follows.

void get_sentence(string s);

From lab 5, you may have already noticed that we can use the string inside the function, but we cannot change the value of the string inside the function. In order to change the value of the string inside the function, we need to add an ampersand (&) in front of the parameter.

void get_sentence(string &s);

Before moving on, here are a couple of questions to think of:

1. Can we change the value of the string inside the function if we match up the parameter and argument names? For example, change the function prototype to:

void get_sentence(string sentence);

2. What is passed into the function if an ampersand (&) is added in front of the parameter?

Once you understand the concept, trace your code, line by line, use a diagram to compare and explain what is actually happening in the above two functions.

Now, let's think of something more interesting. Read through the following code:

```
int factorial (int n); //line 1

int main()
{
     int n;
     cout << "Enter a positive integer:";
     cin >> n;

     cout<< "Factorial of " << n << "is " << factorial(n);

     return 0;
}

int factorial (int n)
{
     if (n==1)
          return 1;
     else
          return n * factorial (n - 1);
}
```

What if you have a function calls itself? As we did above, use a diagram to trace the code, line by line, and explain what is happening behind the scene.

In addition, answer the following questions:

1. If user entered 5, how many times the factorial() will be executed?
2. What if line 1 is missing? Explain why.
3. What if "`if (n==1) return 1;`" is missing? Explain why.
4. There is an int variable n in main function, and an int variable n in factorial function, are they the same variable?

(5 pts) Work on assignment 4

Now, continue working on assignment 4, and finish at least 3 functions by the end of the lab (do NOT include main). Ask TAs for clarifications if needed.

Show your completed work to the TAs for credit. You will not get points if you do not get checked off!