Lab 1 Decoder Lab
(Getting Started, Review)

60 pts *Due Fri, Feb 10, midnight*

This lab is associated with week 1's videos
(https:/www.eecis.udel.edu/~yarringt/CISC220/Latest/week1.html) Watch the videos first.

Approximate Time Requirement: Approximately 6 hours or less. This should be largely a review, coding-wise, and an introduction to the subtle differences between c and c++.

Approximate Level of Difficulty: ★½☆☆☆ In terms of difficulty, I consider this to be a 1.5 out of 5

Everyone should have this working on their computer to make sure everyone has access to a working C++ compiler and has a basic understanding of C++ syntax. Please help others. But everyone should have this working on their own computer before turning it in.

For us to grade this lab, it must compile. That holds for all labs/projects in this class.

To turn in: The cpp file, along with a screenshot of the running code. The screenshot helps us in situations when the code does not run on our computer (even though it should work across all cpp compilers) yet it runs on your computer.

Contents

Purpose of this lab:

This lab should serve a few purposes:

1) Make sure your compiler is working properly
2) Get you familiar with C++ syntax and the differences between C and C++
3) Refresh your memory for coding
4) Get you used to my style
5) Ease you back into yet another crazy semester
6) Allow you to experience the joy and happiness of writing functions and getting your code to work

How this lab works:

This lab is in essence a decoder lab. If you write the functions as described successfully, and run it with the main I've given you, you will decode the input I've given into intelligible output. Hint – if your output is gibberish, you did something wrong.

Note:

This lab does not have test cases for you to test your code. Normally I will provide or you should create test cases for all functions/methods, with input parameters and resulting output results. HOWEVER in this case, the lab is a secret decoder lab – you should know you executed the function correctly if you get a word/phrase that makes sense.

What you need to do:

If you are using eclipse, in eclipse, create a new project. Once you've created a project, create a new source file. If you are not familiar with how to do this, please watch my introductory videos on my web site (https://www.eecis.udel.edu/~yarringt/CISC220). I go over how to create a project and a source file, and then how to compile.

Once you've created your source file (make sure it has a .cpp extension!), copy my code below.

For the lab you will need to:

1) Make sure the function declarations (above main) match your function definitions you are writing
2) Fill in the function definitions, as described, below the main.
   (If you do not know the difference between the function declarations and the function definitions, I highly recommend you watch week 1's videos on my web site, https://www.eecis.udel.edu/~yarringt/CISC220)

The main function is fine as it is.

Don't change or modify the main function, other than to comment out functions you haven't written yet so that you can test functions as you write them. But otherwise, you should not make ANY modifications to the main function!

/**/

/* Code */

/*Copy these libraries into the top of your .cpp file*/

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

```
#include <stdlib.h>
#include <string.h>
using namespace std;


/*******************************************************
```

Function Declarations

```
Make sure the function Declarations (here) match the function definitions you will be
writing.
/* Note: function declaration goes above main, function definition gets
* written below main
*/
void func1(string s, int ct);
void func2(string s, int len);
void func3(string s, int len);
int func4(int arr[], int len);
int func5(int arr[], int val, int len);
void func6(int arr[], int ind, int len);
void func7(string s, int arr[], int len, int i1, int i2,bool do_pr);
void func8(string s, int len, int arr[], int len2);
void func9(string s7,int f9arr1[],int f9arr2[], int len1,int len2);
void func10(string str10,int f10arr[],int len);
void func11(string s7,int len,int msize);
void func12(string s,int len,int msize);
void func11a(char mat[5][5],int xcoord[],int ycoord[], int len);
void func13(string s,int len,int msize);

*******************************************************/
```

//COUT (PRINTING) Info:

```
/* How to write output on one line versus separate lines:
        * cout << "hi"<< endl;
        * pipes "hi" into a buffer, and the endl flushes the buffer and moves to
        * the next line.  If you want to wait until all the characters are in the
        * buffer before you flush it and move to the next line, skip the endl;
        * so you'd have
        * cout << "hi";
        * and then whenever you want to flush the buffer, you would just add the
        * line,
        * cout << endl;
        * One final point: you can always add something after each thing that isn't
        * an endl;.  YOu can add a comma:
        * cout << x << ",";
        * or a tab:
        * cout << x <<"\t";
        * or even a star:
        * cout << x <<"*";
*

*********************  MAIN START    ************************************/
/* Copy and paste the main into your .cpp file. Ctrl-shift-v to maintain formatting*/
int main() {
       string alpha= "abcdefghijklmnopqrstuvwxyz";
       cout << "Hello World!" << endl;
       /*************************************************/
```

```
        /* for func1 */
        string str0 = "coffee! ";
        int x = 4;
        func1(str0,x);


        /*****************************************************/
        /* for func2 */
        string string1 = "kceacruanmpedle sluastethev"; // 28 characters total
        func2(string1,28);



        /*****************************************************/
       /* for func3 */
       string string2 = "speliknuweorrubs tergudeunfe cestmanlbowctoshac";
        func3(string2,47);


        /*****************************************************/
        /* for func4 */
        int f4arr1[6] = {17, 10, 5, 20, 18, 2};
        int f4arr2[8] = {7, 21, 9, 6, 2, 39, 3, 25};
        int f4arr3[4] = {-2,1,7,2};
        int f4arr4[5] = {5, 8, 2, 13, 7};
        int f4arr5[7] = {12, -4, -2, 3, -7, -5, -3};
        cout <<
alpha[func4(f4arr1,6)]<<alpha[func4(f4arr2,8)]<<alpha[func4(f4arr3,4)]<<alpha[func4(f
4arr4,5)]<<alpha[func4(f4arr5,7)]<<endl;


        /*****************************************************/
       /* for func5 */
        string string5 = "vapkfslifbhnoueweoihsakbix";
        int f5arr1[14] = {57,16, 46, 41,75,31,64,21,18,13,66,68,81,10};
        int f5arr2[20] =
{22,66,67,85,90,92,70,81,42,79,32,27,99,25,49,74,68,12,55,94};
        int f5arr3[17] = {96,64,10,73,4,14,71,37,77,52,54,93,19,26,89,84,47};
        int f5arr4[13] = {28,49,44,19,37,42,26,78,12,88,67,51,66};
        int f5arr5[29] =
{11,32,82,90,97,84,30,77,46,6,69,41,79,48,17,65,25,53,33,16,5,7,21,94,38,27,95,100,92
};
        int f5arr6[15] = {84,30,77,46,6,69,1,79,48,17,65,25,53,33,16};
        cout << string5[func5(f5arr1,48,14)] << string5[ func5(f5arr2,24,20)
]<<string5[func5(f5arr3,80,17) ] << string5 [func5(f5arr4,39,13) ] << string5
[func5(f5arr5,36,29) ] << string5 [func5(f5arr6,27,15) ]<< endl;


        /*****************************************************/
       /* for func6 */
        string string6= "leztiwrqekrktfmwrbfhyljmuekpvonitapsxp";
        int f6arr1[10] = {17,4,32,12,8,6,14,24,21,11};
        int f6arr2[7] = {21,33,13,18,5,8,35};
        int f6arr3[4] = {15,19,32,4};
        int f6arr4[5] ={35,20,27,24,6};
```

```
	/**/
	func6(f6arr1,7,10);
	for (int i = 0; i < 10; i++) {
		cout << string6[f6arr1[i]];
	}
	cout << " ";
	func6(f6arr2,0,7);
	for (int i = 0; i < 7; i++) {
		cout << string6[f6arr2[i]];
	}
	cout << " ";
	func6(f6arr3,1,4);
	for (int i = 0; i < 4; i++) {
		cout << string6[f6arr3[i]];
	}
	cout << " ";
	func6(f6arr4,2,5);
	for (int i = 0; i < 5; i++) {
		cout << string6[f6arr4[i]];
	}
	cout << endl;


	/*****************************************************/
	/* for func7 */
	string s7 = "skldhbeoieldoodwac hkq cocu iaoe crhpogusyiahhifcet";
	int f7arr[27] =
{25,45,48,12,10,16,50,9,32,17,19,36,27,33,30,24,20,28,7,6,18,14,37,39,38,42,44};
	func7(s7, f7arr, 27, 18,2,false);  /*  won't decode */
	func7(s7,f7arr,27,12,25,true);  /* will decode */


	/*****************************************************/
	/* for func8 */
	 string s6 = "xezuhnbl_uiplypdhqlb";
	 int len = 20;
	 int f8arr[] = {2351,92837,482,65,1039,233,23095,1,2037,693842,283};
	 int len2 = 11;
	 func8(s6,len,f8arr,len2);


	/*****************************************************/
	/* for func9 */
	 string s9 = "slckfoiold,eryt coteidkfzlvkaljwoeihy
dnivgphweugaltzsdxls,gfxnbcm wakrm qblanuaksdhzlkjets, anuakd shojqhwtis
uzcdoalbicsoiabasboihs";
     int f9arr1[30] = {2,7,9,10,15,24,36,37,42, 44,51,57,66,68,70,74,76,80,88,
89,92,94,98,101,102,113, 115, 120, 122, 124};
	 int f9len1 = 30;
	 int f9arr2[17] = {8,16,17,39,40,58,67,71,72,75,77,90,91,93, 99,107,110};
	 int f9len2 = 17;
	 func9(s9,f9arr1,f9arr2,f9len1,f9len2);


	/*****************************************************/
```

```
      /* for func10 */
        string str10 = "alettioe twhaaabosi hkcxl";
        int f10arr[23] = {22,11,13,22,5,13,8,3,3,11,7,14,3,19,11,5,1,3,12,9,4,2,22};
        func10(str10,f10arr,23);

     /***********************************************/
        /* for func11a: */
        char mat[5][5] =
{{'d','k','e','l','p'},{'a','y','a','m','r'},{'a','i','r','k','f'},{'s','e','w','o','
z'},{'r','e','c','u','x'}};
        int xcoord[6] = {3,1,2,2,0,4};
        int ycoord[6] = {2,0,4,4,3,1};
        int  len_a = 6;
        func11a(mat,xcoord,ycoord, len_a);

      /*****************************************************/
      /* for func11 */
        string s11 = "mboehdkuxkwpmnfzicosyfmeuqlaitcvwdgn";
        len = 36;
        func11(s11,len,6);


      /*****************************************************/
      /* for func12 */
        string str12 = "gboehdpixkwpangzicjsygmeaqlaltmvwdgeabndckswoshbk";
        len = 49;
        func12(str12,len,7);


      /*****************************************************/
}
/*******************************END MAIN ***************************************/


/*******************************************************************************/
Function Definitions (Where you'll be writing the function definitions!
*/
/*******************************************************************************/


      /*****************************************************
      func1:(2 pts)
       * Write a function that takes as input:
       *     the string string0
       *     and an integer
       * The function returns:
       *     nothing
       * The functions should loop such that it prints that string the integer number
       * of times.  So if the string was "glub" and the int was 5, the
       * function would print out
       * >>glubglubglubglubglub (or, alternatively,
       * >>glub
       * glub
       * glub
```

```
 * glub
 * glub
 *
 *  Point: to make sure you're comfortable with a basic loop
 *
 * (See note above on how to make things print on the same line below)
 * NOTE 2:  All function definition go BELOW the main function
 */
/*****************************************************/
void func1(string s, int ct) {
/* write function definition here */
}

/*****************************************************
```

func2: (2 pts)

```
 * Write a function that takes as input:
 *     the string below (string 1) and
 *    the length of the string.
 * The function returns:
 *     nothing
 * This function should use a while loop to print out every other
 * character, starting at 1

 * Point: I want to make sure you can loop through a string and that you
 * know how to use a while loop
 */
void func2(string s, int len) {
/* write function definition here */
}

/*****************************************************
```

func3: (4 pts)

```
 * A) Make a function header that matches the function's declaration, above
 *    the main.  The function that takes as input:
 *    a string and
 *    the length of the string.
 * The function returns:
 *    nothing
 * B) Using a for loop, print out every other character in the string,
 * starting at the last index in the string and working back to the first
 * character
 * Point: I want to make sure you know how to use a for loop and how to
 * write a function's header.
 */
func3() {
/* write function definition here NOTE: FOR THIS FUNCTION YOU MUST FILL IN THE
FUNCTION's HEADER so that it matches the function/s declaration (above main)*/
}


/*****************************************************
```

func4: (2 pts)

```
 * write a function that takes as input:
 *    an array of ints, and
```

```
        *    the length of an array.
        * It returns:
        *    an int.
        * The function returns the average of the array values.
        *
        * Point: I want to make sure you know how to traverse an array, and sum
        * numbers in an array, and return a value.
        */
int func4(int arr[], int len) {
/* write function definition here */
}

       /**************************************************
        func5: (3 pts)
        * write a function that takes as input:
        *     an array of ints,
        *     the length of an array,
        *     and an integer
        * It returns:
        *     an int.
        * The function should find the value in the array closest to the integer, and
        * return the index of that integer
        *
        * Point: To make sure you understand how to update values in a loop
        */
int func5(int arr[], int val, int len) {
/* write function definition here */
}

       /**************************************************
        func6: (3 pts)
        * write a function that takes as input:
        *     an array of ints,
        *     the length of an array, and
        *     an integer (which acts as an index into the array).
        * It returns:
        *     nothing.
        * The function should find the smallest value in the array, and swap
        * that value with the value at the index integer.
        * For instance, if you've got the following array:
        * [8,3,5,1,2,7,6,9,4]
        * and the index sent in was 7:
        * the smallest value in the array is 1, at index 3.
        * The value at index 7 is 9.  So 1 and
        * 9 would be swapped in the array, resulting in:
        * [8,3,5,9,2,7,6,1,4]
        *
        * Point: To make sure you can find (and update) the smallest value, and
        * can swap
        */
void func6(int arr[], int ind, int len) {
/* write function definition here */
}
```

```
/*************************************************
```

func7: (4 pts)

```
 * write a function that takes as input:
 *     string, an array of ints,
 *     the length of an array,
 *     2 ints that will act as indices into the array of ints
 *     and a boolean value that is set true if the function should print out
 *     the values in the string (as described below) or false if the function
 *     should not print.
 * The function returns:
 *   nothing
 *
 * The function shifts all values to the right between the first and second
index parameters,
 * and replaces the value at the first index with the value at the last index.
 * So, for instance, if the array was:
 * [1,2,3,4,5,6,7,8,9]
 * and the 2 parameters were:
 * 3, 7 (or 7, 3 - you don't know the order),
 * the result should be
 * [1,2,3,8,4,5,6,7,9]
 *
 * The function should then use the array numbers as indices into the string
 * parameter and should print out those characters.
 * an integer (which acts as an index into the array).
 *
 * Point: To emphasize the number of steps needed in inserting values into an
array...
 */
void func7(string s, int arr[], int len, int i1, int i2,bool do_pr){
/* write function definition here */
}

/*************************************************
```

func8: (5 pts)

```
  * write a function that takes as input:
 *     a string,
 *     the length of the string,
 *     an array of integers,
 *     and the length of the array of numbers.
 * The function  returns:
 *     nothing
 * For each of the numbers in the array of numbers, the function adds up the
 * digits in the number.  It then calculates an index into the string(s6,
below) by
 * finding the remainder when dividing by the string size.
 * So if the string is:
 * "grandma"
 * the length of the string is 7.
 * Say the number in the array of numbers is 5497
 * Adding the digits results in 25.
 * If we divide by 7, the remainder is 3.
 * So the resulting character would be 'n' (which is at index 3 in the string)
 * The function should print out the letter in the string at that index.
```

```
        * Point: number manipulation
        */
void func8(string s, int len, int arr[], int len2) {
/* write function definition here */
}


       /************************************************
        func9: (8 pts)
        * write a function that takes as input:
        *    a string,
        *    2 arrays of ints,
        *    2 ints (the lengths of both of those arrays)
        * The function returns:
        *    nothing
        * Note: BOTH of the input int arrays are in order, from smallest integer to
        * largest integer!
        * The functions creates a new array whose length is the length of the first
        * array plus the length of the second array.  The function then combines the
        * first array and the second array into the newly created array such that all
        * the values are in order.

        * In other words, if the two arrays are:
        * [2,4,6,7,8,8,14,15] and [1,2,3,5,6]
        * the resulting new array would be:
        * [1,2,2,3,4,5,6,6,7,8,8,14,15]
        *
        * The function then uses the integers in the array to print out the
        * corresponding character in the input string,
        * e.g.,
        * If the input string was "saligatorihaboiwhbwoihbw"
        * the char at index1 is a, the char at index2 is s, etc. so the output would
be:
        * alligattorriw (yeah, not a word, but  you get the idea...)
        */
void func9(string s9,int arr1[],int arr2[], int len1,int len2){
/* write function definition here */
}

       /************************************************
        func10: (9 pts)
        * write a function that takes as input:
        *    a string,
        *    an array of integers
        *    an integer that indicates the length of the array
        * The function returns:
        *    nothing
        *
        * The functions takes the array of integers, and removes all duplicates
        * (there is more one way to do this).
        *
        * In other words, if you've got:
        * 8,3,6,7,8,1,4,3,6,2,3
        * the resulting array (aka sub-array) would be:
```

```
        * 8,3,6,7,1,4,2
        *
        * The function then uses the remaining integers to print out the
        * corresponding character in the input string,
        *  Point:  array manipulation, fun challenge!
        */
void func10(string str10,int arr[],int len) {
/* write function definition here */
}
       /***********************************************/
```

func11a: (6 pts)

```
        /* func11a: write a function that takes as input:
         *a character matrix,
         *an array of ints (for the x coordinate)
         *an array of ints( for the y coordinate)
         *an int (for the length of the arrays)
         *the function returns nothing;
         *The function should print out the matrix at each of the x and y coordinates
(so it should
         *     print the matrix at the first x and first y value, then the matrix at
the second x
         *     and second y value, etc., for all values in the x and y array)
        */void func11a(char mat[5][5], int xarr[], int yarr[], int len) {
/* write function definition here */
}


       /************************************************
```

func11: (6 pts)

```
       * write a function that takes as input:
        *      a string
        *      the length of the string,
        *      and a square matrix dimension size.
        * The function returns:
        *      nothing
        * The function creates a matrix of size by size and fills in the characters
        * such that the first size characters are in the first row, the second size
        * characters are in the second row, etc.
        * Then the function should print out the diagonal characters (0,0 to
size,size).
        * Point: creating and accessing a matrix
        *
        * */
void func11(string s7,int len,int msize) {
/* write function definition here */
}

       /************************************************
```

func12: (6 pts)

```
       *  write a function that takes as input:
        *    a string
        *    the length of the string,
        *    and a square matrix dimension size.
        * The function returns:
```

```
        *    nothing
        * The function creates a matrix of size by size and fills in the characters
        * such that the first size characters are in the first row, the second size
        * characters are in the second row, etc.(so far just like the function above)
        * Then the function should print out the reverse diagonal characters
        * Top right corner to bottom left corner
        * Point: creating and accessing a matrix
        */

void func12(string s,int len,int msize){
/* write function definition here */
}
```

TO TURN IN:

Either a **zipped file** (compressed file) or at **least 2 separate files**:

1) Your COMPILING C++ Code with your name at the top and which compiler you used (this is the .cpp file)
2) A screenshot of your code running (Not all of it, just enough to let us know your code is working on your computer in case we hit any snags trying to run it on our computers).