Review for CS150 Final

The final is comprehensive in the sense that everything builds on itself. Possible questions will center around (but not be limited to) the following:

- Writing (or explaining) program segments dealing with:
  if statements
  switch statements
  while loops
  for loops
  do-while loops
- Expressions using relational and/or logical operators
- Advanced output setw, fixed, setprecision
- Rewriting if blocks as switch blocks and visa versa
- Rewriting loops in one form to another
- Nested loops (tracing or writing solutions to nested loop problems)
- Correcting syntax, logic, and/or runtime errors
- Writing or explaining data file segments including reading from and/or writing to files
- Tracing function calls with various parameter passing options (your favorite)
- Writing functions including passed by reference and passed by value
- Functions calling other functions
- Make sure you understand the various data types and type casting
- Arrays (1D and 2D)
- Character processing

As always, you will be required to write program segments, functions, programs, or combinations of all three. Let's take a look at some example problems you could be expected to solve and their difficulty and NO **the solution to these questions will not be posted** but you are more than welcome to come to my office to discuss your solutions to any or all of these problems.

P#1 (Easy): Write a function **computeEven** that accepts an array **values** and the number of values currently in the array, **howmany**. The function computeEven returns through the function name the number of values in the array that are even. Your function might be called by a statement of the form:

```
cout << "Number of evens = " << computEven (values, howmany) << endl;
```

P#2 (Medium): Write a function **bIsPrime** that accepts an integer value and returns true if the number is prime; otherwise, return false. A prime number is a number that is only divisible by 1 and itself.

P#3 (Hard): Write a function **computePrimes** that accepts an array as defined in P#1 and the number of elements in the array. The function computePrimes is to call bIsPrime to help determine the number of prime numbers in the array values which is passed to computePrimes.

computePrimes is to return the number of prime values in the array passed in to the function.

P#4 (Easy): Write the function prototype for computePrimes in P#3.

P#5 (Easy): Show what a call to computePrimes would look like.

P#6 (Easy): An integer array **values** has been defined and filled beginning at subscript 0 with **howmany** elements in the array.

a) Write a prototype for a void function **insert** that will accept the array **values**, **howmany** elements in the array, and a new integer value to be inserted at the end of the array.

b) Write the actual function **insert** described in a) completely.

P#7: (Hard): Assume the array in P#6 is sorted from smallest to largest. Insert the value into the array in the proper position of the array preserving the sorted nature of the array.

P#8: (Medium): What will the following C++ program print?

```
#include <iostream.h>
void change (int&, int, int&);
void main()
{
  int i, j, k;
  i = 22;
  j = 4;
  k = 7;
  change (i, j, k);
  cout << i << "  " << j << "  " << k << "  " << endl;
  change (k, i, j);
  cout << i << "  " << j << "  " << k << "  " << endl;
}
void change (int& i, int j, int& k)
{
  i = i + k;
  j = i / 3;
  k = i % j;
}
```

P#9 (Hard): Many infinite series have sums equal to pi. One formula goes as follows:

```
pi = 4 - 4/3 + 4/5 - 4/7 + 4/9 - 4/11 + ....
```

This sequence consists of alternating positive and negative terms. You are to write a single C++ function which estimates pi using the above formula. The function might be called by a statement of the following form:

```
  cout << "Pi = " << computePi(numterms) << endl;
```

where **numterms** is an integer value containing the number of terms to be used from the above sequence.

P#10 (Easy): Given the following prototypes and declarations, place a check by only the illegal procedure calls in the following list AND specify what caused the error.

```
int compute (int, int&, int);
float doit (float&, int, float, char&);

void main()
{
  int i,j;
  int nums[10];
  float x,y;
  char ch;
```

_____ a) `compute (i, j, i + j);`

_____ b) `compute (i, nums[1], char ch);`

_____ c) `doit (x, i, y, 'c');`

_____ d) `doit (x + y, nums[i], y, ch);`

_____ e) `doit (x, 3, x, static_cast<char>( static_cast<int>('a') + 1)));`

P#11 (Easy): Explain what is meant by the scope of a variable.

P#12 (Medium): Write a program segment which inputs a character value into the variable **ch** from the keyboard in the range of '0' to '9' inclusive. With one assignment statement, convert the value in **ch** to the numerical value of the digit and place this value in the integer variable **val**. As an example, if ch = '0', then val would equal the integer 0.

P#13 (Medium): Will the following program segment count and print the number of lines in a data file? Why or why not? If not, what does the program do?

```
int count = 0;
char ch;

cin.get (ch);
while (!cin.eof())
{
  if (ch = '\n')
  {
    count++;
  }
  else
  {
    cin.get (ch);
  }
cout << count << endl;
```

P#14 (Easy): Which of the following declarations are legal?

a) int arry1[5] = {5, 4, 3, 2, 1, 0};

b) int arry2[5] = {1, 2};

c) char ch[5] = {'12345'};

P#15 (Hard): An array **values** exists containing howmany elements. Write a function **bAnyDuplicates** that returns true if there are any duplicate values in the array; otherwise, return the value false. A call to your function might be as follows:

bDuplicates = bAnyDuplicates (values, howmany);

P#16 (Easy): Create a 2D array **values** with 4 rows and 3 columns. Place values of your chosing into the array.

P#17 (Medium): Write a function **writeValues** that accepts the array created in P#16 along with the number of rows and columns. You are to write each value from the passed in array to a file **data.txt** one row at a time with each value separated by a single space.

P#18 (Hard): Rework problem #15 but instead of passing in a 1D array, you are to accept the 2D array from P#17 and the number of rows and columns.

a) Write the function prototype.

b) Write the function definition.

c) Show what a call to your function might look like.

P#19 (Medium): Write a program segment that prints the number of characters (including whitespace) in the data file story.txt.
