Honors Computer Science C++
Mr. Clausen
Program 4A, 4B, 4C (4D, 4G)

Program 4A: Function: Comments And Output 10 points

Write a program that practices comments, output, and void functions. Save the program as LastNameFirstNameP4A.cpp in your "S:" directory.
Rewrite program 2A into a program that has all the output information in a void function. Call this function void Display_My_Info ( ) which includes all the information you are supposed to include in every program displaying your information for the class. Turn this into a program using the **int** main function and just this one function call to Display_My_Info ( ).
The example on the bottom of page 137-138 and the function on page 139 will help you with this program.
As you type all your programs this year, be sure not to type past the 80-column line. If you have any statements longer than 80 columns, press the return key to "wrap" the statement around to the next line.

1) Type comments at the beginning of the program to display your name and other information just like those used for program 2A.
2) Include <iostream> so we can use the **cout** command. Include <iomanip> so we can use the **setw** command.
3) Type: using namespace std;
4) Don't forget to include comments for your function, right before your function declaration.
5) Declare the function Display_My_Info ( ) in the function declaration section of the program.
6) Use a comment line with **equal signs** to separate all of the above from the int main function. For example:
   //===
7) Inside the **int** main() function, call the function: Display_My_Info ( ). This function call should be on the first few lines after the left curly bracket that begins the main function. Don't forget that void functions can be called just by writing the function name including any of the "actual" parameters. Then tell the user to press any key to continue, have your cin.get() command, and your return 0 command.
8) After the **int** main() function, have a comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:
   //---
9) Implement the function Display_My_Info ( ) which will output all of the information from program 2A.
10) After the last function implementation of every program, end your program with two comment lines of equal signs. This signifies the end of your source code.

For Example:
//===
//===

When you are finished with your program, have tested it thoroughly to make sure that everything is correct, and are sure that you don't need to make any changes, then save your program in the "T" network mapping, and the Program 4A folder.

Program 4B Function Cone Heads (20 points)

Write a program to calculate the volume and surface area of a right circular cone using functions. This program is a rewrite of program 3A using functions. Save the program as LastNameFirstNameP4B.cpp in your "S:" directory. To see a model for this program look at the source code for the program "pizza.cpp" or "pizza2.cpp" in the network directory titled: HnrCSCPPFiles. Look in the folder, Text Book Programs and Ch4. As you type all your programs this year, be sure not to type past the 80-column line. If you have any statements longer than 80 columns, press the return key to "wrap" the statement around to the next line.

1. Type comments at the beginning of the program to display your name and other information just like those used for program 2A.
2. Include <iostream> so we can use the **cout** command. In order to use the square root function, sqrt you need to include <cmath>. Include <iomanip> so we can use the **setprecision** command.
3. Type: using namespace std;
4. Declare constants of type **double** for PI and ONE_THIRD and use these identifiers in your formulas to calculate area and volume. ONE_THIRD needs to be declared as **double** too, and be sure to use 1.0/3.0 to insure a real number quotient.
5. Don't forget to include comments for your functions, right before each of your function declarations.
6. Declare the function Display_My_Info ( ) in the function declaration section of the program. This should be a void function that doesn't take any parameters and doesn't return any values. In addition, declare functions for Get_Data(), Area(), Volume(), and Display_Output(). The Get_Data() function should be a void function using reference parameters of type **double** for radius and height**.**. Area() should be a value returning function of type **double** that has parameters for the radius and the height. Volume() should be a value returning function of type **double** that has the parameters for radius and height. Display_Output() should be a void function using constant reference parameters to echo the radius and height, and display the volume and surface area.
7. Use a comment line with **equal signs** to separate all of the above from the int main function. For example:
   //===

8. Inside the **int** main() function, **declare all of your variables** then, **call the functions**: Display_My_Info ( ), Get_Data(), Area(), Volume(), and Display_Output(). These function calls should be on the first few lines after the left curly bracket that begins the main function (after your variable declarations). **Don't forget that value-returning functions are "called" in assignment statements, while void functions can be called just by writing the function name including any of the "actual" parameters.** Then tell the user to press any key to continue, have your cin.get() command, and your return 0 command.
9. After the **int** main() function, have another comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:
   //----------
10. Implement all of the functions, separating each one from the other ones using comment lines of **subtraction signs**.
11. After the last function implementation of every program, end your program with two comment lines of equal signs. This signifies the end of your source code. For Example:
    //==========
    //==========

Program 4C Everybody Loves A Hero (25 points)

Write a program to calculate the area of a triangle using Hero's formula and functions.
Save the program as LastNameFirstNameP4C.cpp in your "S:" directory.
As you type all your programs this year, be sure not to type past the 80-column. If you have any statements longer than 80 columns, press the return key to "wrap" the statement around to the next line.
VERY IMPORTANT: WHEN YOU TEST YOUR PROGRAM, MAKE SURE THE NUMBERS THAT YOU ENTER FOR THE 3 SIDES ACTUALLY FORM A TRAIANGLE. THE SUM OF ANY 2 SIDES MUST BE GREATER THAN THE THIRD SIDE. Try 3, 4, 5 (a right triangle), then 5, 5, 5 (an equilateral triangle), and then make up the sides for a scalene triangle.
Here are the formulas necessary to calculate the area of a triangle using Hero's formula:

$$Area = \sqrt{s(s-side1)(s-side2)(s-side3)}$$

$$s = \frac{(side1 + side2 + side3)}{2}$$

1. Type comments at the beginning of the program to display your name and other information just like those used for program 2A.

2. Include <iostream> so we can use the **cout** command. In order to use the square root function, sqrt you need to include <cmath>. Include <iomanip> so we can use the **setprecision** command.
3. Type: using namespace std;
4. Don't forget to include comments for your functions, right before each of your function declarations.
5. Declare the function Display_My_Info ( ) in the function declaration section of the program. This should be a void function that doesn't take any parameters and doesn't return any values. In addition, declare functions for Get_Data(), S(), Area(), and Display_Output(). The Get_Data() function should be a void function using reference parameters of type **double** for side1, side2, and side3. S() should be a value returning function of type **double** using value parameters of type **double** for side1, side2, and side3. Area() should be a value returning function of type **double** using value parameters of type **double** for s, side1, side2, and side3. Display_Output() should be a void function using constant reference parameters to echo side1, side2, side3, tell us the value of "s", and display the area of the triangle.
6. Use a comment line with **equal signs** to separate all of the above from the int main function. For example:
   //==========
7. Inside the **int** main() function, declare all of your variables then, call the functions: Display_My_Info ( ), Get_Data(), S(), Area(), and Display_Output(). These function calls should be on the first few lines after the left curly bracket that begins the main function (after your variable declarations). Call all of the functions from **int main**. **Don't forget that value returning functions are "called" in assignment statements, while void functions can be called just by writing the function name including any of the "actual" parameters** Then tell the user to press any key to continue, have your getch() command, and your return 0 command.
8. After the **int** main() function, have another comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:
   //----------
9. Implement all of the functions, separating each one from the other ones using comment lines of **subtraction signs**.
10. After the last function implementation of every program, end your program with two comment lines of equal signs. This signifies the end of your source code. For Example:
    //==========
    //==========

Program 4D: Get the Points (50 points)

Write a computer program that enters 2 integer ordered pairs: (x1, y1) and (x2, y2). The program should calculate the midpoint of those 2 points, the distance between the points, and the slope of the line between the points. distance = sqrt (pow ((x1-x2),2) + pow ((y1-

y2), 2)). Slope Numerator = (y2-y1) and Slope Denominator = (x2-x1) and the midpoint is just averaging the x points and the y points **double**(x1+x2)/2.0 and **double**(y1+y2)/2.0. **Make sure that this program is written using functions**. You should have a void function Display_Whose_Output to display all the information on the screen that says it is your output. Use a void function with reference parameters to Get_Data, a void function with value and reference parameters to Calculate_Midpoint, a void function with value and reference parameters to Calculate_Slope returning integers for the slope numerator and the slope denominator, a value returning function with value parameters returning a **double** precision Distance_Between_Points, and a void function with value parameters to Display_Output. Carefully consider using value parameters for all data that are passed into functions and don't need to be returned. It is good style to list all the value parameters before the reference parameters. Make sure that your funciton declarations, function implementations and function calls match.
Separate all functions from each other with a comment line of dashes. Separate the function declarations from the main function with a comment line of equal signs. Don't forget to include all the comments for each of the function declarations, including function name, brief description, input and output. **Remember, void functions have active verb names and are called by the function name as it's own statement, and value returning functions have noun names and are called as a part of other statements (cout or assignment). We are going to use a style where the first letter of each word in our function names start with capital letters and variables start with lower case letters. Remember also, global constants are good, and global variables are bad, so use local variables.**
Write this program using the Top Down Design and Stub Program approach so you can test each function independently.
For now, don't worry about data points that give you a denominator of zero for the slope. We will change this program later, once we learn "if then" statements. Also don't worry about 2 data points being entered as the same point with a distance of zero, we can fix that later too. And don't fret about reducing the slope, we will learn how to find the gcf and reducing fractions later also. **Leave your slope as a separate numerator and denominator.**

Program 4G Harry or Harriette (25 points)

Write a graphics program that draws a face. Use the graphics commands for: Line, Rectangle, Circle, Ellipse, Arcs, and Points (**Please make sure to use all of them or you will not receive full credit for your program.**). Draw a picture of a face (the face can be a square, rectangle, circle, etc.) including eyes, nose, mouth, etc. Have the face fill the screen (no little faces here) and add as much detail as you can.
Open the source code for the program "GraphicsTemplate.cpp" in the network directory titled: HonorsCompSciCFiles. Look in the folder named Graphics. Save the program as LastNameFirstNameP4G.cpp in your "S:" directory.
Don't forget that you need to create a DOS project with the proper settings for your program to work. Here is the link to the notes on how to set up the project:

http://www.lcusd.net/lchs/dclausen/cs_cpp_hnr/Lectures/Borland_Graphics_Project.pdf

The Name of the project should be your ID number, then P4G. For example: 9999P4G.ide

As you type all your programs this year, be sure not to type past the 80-column line. If you have any statements longer than 80 columns, press the return key to "wrap" the statement around to the next line.

1) Type comments at the beginning of the program to display your name and other information just like those used for program 2A.
2) Leave your graphics displayed on the screen until the user presses any key to continue. Include <graphics> so you can draw your graphics commands.
3) Declare functions for each part of your face that you will draw: Left_Eye, Right_Eye, Nose, Mouth, etc.
4) Use a comment line with **equal signs** to separate all of the above from the int main function. For example:
   //===
5) Inside the **int** main() function should be the graphics variable declarations and initializations and function calls to all of the functions declared in Step #3 above. Then tell the user to press any key to continue, have your cin.get() command, and your return 0 command.
6) After the **int** main() function, have another comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:
   //---
7) Implement the functions declared in Step #3 adding your commands to the program to complete the assignment.
8) Separate each function implementation from the other function implementations with comment lines of subtraction signs.
9) After the last function implementation of every program, end your program with two comment lines of equal signs. This signifies the end of your source code. For Example:
   //===
   //===