

# Introduction To Computer Programming C++

## Mr. Clausen

### Program C11A, C11B

#### Program 11A: Cone Heads with Functions 30 points

Write a program that calculates the volume of a right circular cone using functions. **Do Not Use Any Global Variables in this program! You will have 2 global CONSTANTS, but will use all local variables.**

Save the program as LastNameFirstNameP11A.cpp in your "S:" directory. To see a model for this program, look at the source codes for the programs "pizza.cpp" in the network directory titled: IntroCompProgFiles. Look in the folder, Our Textbook Resources, Data Files Students, Unit 3 C++, and the Lesson 11 folder. As you type all your programs this year, be sure not to type past the 80-column line in Borland C++ 5.02 for Windows. 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 1A. **Make sure to change the program name and program description in these comments, so that the program number, name, and description say what is listed above.**

2) Include <iostream.h> so we can use the **cout** and **cin** commands. Also include <conio.h> so you can use **getch()** to leave your output displayed on the screen until the user presses any key to continue, and include <iomanip.h> so we can use the **setprecision** command. Also include <math.h> so we can use the **pow** function to raise any base to any power.

4) Declare a constant of type **double** for PI. Also declare a constant named ONE\_THIRD to represent 1.0/3.0, being careful **not** to use integer division.

For example: **const double** PI = 3.14159;  
**const double** ONE\_THIRD = 1.0/3.0;

5) Don't forget to include comments for your functions, right before each of your function declarations. Remember that these are the five lines of comments for each function as listed below:

```
//Function: Type the function name here
//Type a brief description of what the function does here
//
//Input: list the variables that are passed into the function here
//Output: list the variables that are returned from the function here
```

6) Declare the function **void 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: **void Get\_Data(double & radius, double & height); double Volume(double radius, double height);** and **void Display\_Output(double radius, double height, double volume);** The

Get\_Data() function should be a void function using reference parameters of type **double** for radius and height. Volume() should be a value returning function of type **double** that has value parameters for radius and height and returns the volume of the cone. Display\_Output() should be a void function using value parameters to echo the radius and height, and display the volume. **Do not use reference parameters ( & ) except where indicated above.**

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 variables of type **double** for radius, height, and volume. Initialize each of these variables to 0.0. Call the function: Display\_My\_Info ( ). This function call should be right after the variable declarations. After this, call the Get\_Data() function. Next, call the function Volume in an assignment statement, for example: volume = Volume(radius, height). Finally, call the function Display\_Output(). Don't forget to send the actual parameters (arguments) to the functions when you call them. **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 (arguments). Also remember that actual parameters (arguments) don't include the data type.** Then tell the user to press any key to continue, have your getch() command, and your return 0 command.

9) After the **int** main() function, have a comment line of **subtraction signs** to separate the above from your function implementation lines below. For example:

```
//-----
```

8) Implement all of the functions, separating each one from the others using comment lines of **subtraction signs**. In the Get\_Data function you will need to ask the user to enter the radius and height of the cone. In the Volume function, you will need to calculate the volume of the cone using the following formula:  $\text{volume} = \text{ONE\_THIRD} * \text{PI} * \text{pow}(\text{radius}, 2) * \text{height}$ ; Remember that you need to declare a local variable named volume in the beginning of this function, and don't forget to return this value using a **return** statement as the last line of this function. In the Display\_Output function, echo back the values of the radius, and height, and give the volume of the right circular cone. All of these should be displayed to one decimal place using the setprecision command at the beginning of this function after the variable declaration.

9) After the last function implementation of this program, end your program with two comment lines of equal signs. This signifies the end of your source code. For Example:

```
//=====
```

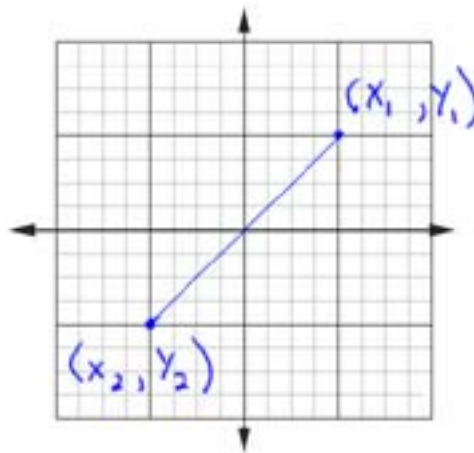
```
//=====
```

When you are finished with your program, have tested it thoroughly to make sure that your program is correct, and are sure that you don't need to make any changes, then save your program in the "W" network mapping, and the Program 11A folder.

### Program 11B: Distance Between Two Points 40 points

Write a program that calculates the distance between any two points on a graph. The formula is as follows and is a variation of the Pythagorean Theorem.

$$\text{distance} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$



Save the program as LastNameFirstNameP11B.cpp in your “S:” directory. To see a model for this program, look at the source codes for the programs “pizza.cpp” in the network directory titled: IntroCompProgFiles. Look in the folder, Our Textbook Resources, Data Files Students, Unit 3 C++, and the Lesson 11 folder. As you type all your programs this year, be sure not to type past the 80-column line in Borland C++ 5.02 for Windows. 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 1A. **Make sure to change the program name and program description in these comments, so that the program number, name, and description say what is listed above.**
- 2) Include <iostream.h> so we can use the **cout** and **cin** commands. Also include <conio.h> so you can use **getch()** to leave your output displayed on the screen until the user presses any key to continue, and include <iomanip.h> so we can use the **setprecision** command. Also include <math.h> so we can use the **pow** function to raise any base to any power and use the square root function.
- 3) There are no constants to declare. **Do Not Use Any Global Variables in this program!**
- 4) Don't forget to include comments for your functions, right before each of your function declarations. Remember that these are the five lines of comments for each function as outlined below:

```
//Function: Type the function name here
//Type a brief description of what the function does here
//
//Input: list the variables that are passed into the function here
//Output: list the variables that are returned from the function here
```

5) Declare the function **void 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: **void Get\_Data(double & x1, double & y1, double & x2, double & y2); double Distance(double x1, double y1, double x2, double y2);** and **void Display\_Output(double x1, double y1, double x2, double y2, double distance);** **Do not use reference parameters ( & ) except where indicated above.**

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 variables of type **double** for x1, y1, x2, y2 and distance. Initialize each of these variables to 0.0. Call the function: **Display\_My\_Info ( )**. This function call should be right after the variable declarations. After this, call the **Get\_Data()** function. Next, call the function **Distance** in an assignment statement, for example: **distance = Distance(x1, y1, x2, y2)**. Finally, call the function **Display\_Output()**. Don't forget to send the actual parameters (arguments) to the functions when you call them. **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. Also remember that actual parameters (arguments) don't include the data type. Also, don't forget that void functions are 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 a 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 others using comment lines of **subtraction signs**. In the **Get\_Data** function you will need to ask the user to enter two ordered pairs: (x1, y1) and (x2, y2). In the **Distance** function, you will need to calculate the distance between the two points using the following formula:

**distance = sqrt(pow((x1 - x2), 2) + pow((y1 - y2), 2));** Remember that you need to declare a local variable named **distance** in the beginning of this function, and don't forget to return this value using a **return** statement as the last line of this function. In the **Display\_Output** function, echo back the two ordered pairs, and give the distance between them. All of these should be displayed to one decimal place using the **setprecision** command at the beginning of this function after the variable declaration.

10) After the last function implementation of this 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 your program is correct, and are sure that you don't need to make any changes, then save your program in the "W" network mapping, and the Program 11B folder.