

Lab5: Modular Program Design with Functions

Objective

The objective of this lab is to learn to design and implement modular programs using functions:

- Define and implement user-defined functions in C++.
- Invoke (call) function and pass parameters to the functions.
- Process return value from functions.
- Reinforce all the programming concepts learned in Labs 1- 4.

Description

For this lab, you will revisit the three (3) problems from Labs 3 and 4:

1. **Taylor:** For computing the sine and cosine using Taylor series (as defined in Lab 3) define ***taylorSin*** and ***taylorCos*** functions. Call these functions whenever you need to calculate the sin and cos.
2. **Projectile:** Define and use ***projectileHeight*** and ***projectileDistance*** functions that compute the distance and height of a projectile as a function of time as specified in Lab 3.
3. **Stick Figure:** Define and use three function ***drawTriangle***, ***drawRectangle***, and ***drawHollowRectangle*** to draw a stick figure as specified in Lab 3 and another fun figure of your choice using the same principles.

We will use a combination of files, standard input (keyboard) and standard output (monitor screen). For each solution you will develop an algorithm, code the algorithm in C++, develop and test it in the Visual C++ environment.

Visual C++ Environment

A **project** in the VC++ environment is a program. A **workspace** is a folder in which all project-related information is stored. When you create a project you may create it in a new workspace or add it to an existing workspace. Refer to the handout "Working with VC++ programs" for more details. Do the following:

1. Create a workspace called Lab5.
2. Create a project for each program and add it to the Lab5 workspace. Create project names as follows:
 - a. Taylor5
 - b. Projectile5
 - c. StickFigures5

Template for Program Header

Place the following code at the beginning of every source code file that you submit for this class.

```

/*****
* NAME: your name
* PERSON NUMBER: your person number
* PROGRAM: Lab name
* PURPOSE: 1-2 line summary of the purpose of the lab
* DATE: Date of last update
* PLATFORM: Microsoft Visual C++ 6.0 Pro
* Course & Section:
*****/

```

On-line submission of your code

All source code (.cpp) and data files created by you and generated by the programs that you implement will need to be submitted using the on-line command which is available on-line on the course website (www.cse.buffalo.edu/~terril/eas230). A summary of the procedure for submission is as follows:

For example, from your command-prompt, do the following:

- ssh unix.eng.buffalo.edu
 - enter your username and password
- Change directories to where your files are located as shown in the example below,
 - cd ~/eas230/Lab5/Taylor5
- Run the following command for each file
 - submit_eas230 <file>
 - where <file> is the name of the file you need to submit.

Problem 1: Functions for Taylor Series Sin and Cos 30%

In this project you will modify the program for Taylor Series from Lab 3 so that sine and cosine computations are carried out in user-defined functions. Function prototypes are specified below:

```
double taylorSin (double degrees);
double taylorCos (double degrees);
```

Each function takes as input parameter angle in degrees, converts the degrees to radians, computes the sin /cos and returns the respective value.

Definition of Taylor series is repeated here for your reference. We used the approximation given by Taylor Series for sine and cosine, for accuracy to 6 decimal places, we stop iterating when the term is insignificant as quantified by the following expression:

$$\left| \frac{x^n}{n!} \right| < 0.0000005$$

Your computation will “iteratively” add terms after making sure it is significant using the expression given above. For iterating we suggest using the “while” control structure.

The Taylor series approximation for sine and cosine are given below:

$$\sin x = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + (x^9/9!) \dots$$

$$\cos x = 1 - (x^2/2!) + (x^4/4!) - (x^6/6!) + (x^8/8!) \dots$$

In this lab (Lab5) you will compute in a “loop” the sine and cosine for a range of x values (0 <= x <= 90) in increments of 5. Your program should output two sets (tables) of information. First set {x, sin(x)} pairs and the second set {x, cos(x)} pairs. Write the output to the monitor.

Algorithm

Write an algorithm and include it as a “preamble” comment in your program.

Code

Create a new project called **Taylor5** and add it to the workspace **Lab5**. Create a C++ source file named **Taylor5.cpp**.

Submission

Submit the following files: **Taylor5.cpp**.

Problem 2: Projectile Functions

40%

We will modify the projectile problem solved in Lab 3 to accomplish the computation of height and distance using functions for various values of the parameters: angle of launch (alpha in the description below), initial launch velocity (v_0), and launch height (z_0).

- You will input values of alpha (in degrees), v_0 and z_0 from standard input (keyboard).
- You will repeat the input—compute—output cycle if the user has more data.
- The prototype for functions are as given below:

double projectileDistance (double time, double alp, double v0);
double projectileHeight (double time, double alp, double z0, double v0);

- The main function will call the above functions for computing height and distance. It is good programming style to do all the input/output from the main function.
- For each input set, the program will compute a set of $r(t)$ and $z(t)$ and output to the screen as well as to an output file "**Plot5.out**". This is different than Lab3, where the program only generated one set of $r(t)$ and $z(t)$. It will have three columns time, distance and height. Each set of output will have heading that shows the angle of launch. A sample data set is shown below:

Projectile Motion with launch angle 15 degrees.

Time	Distance	Height
0.3	1.88356	10.5632
0.31	1.94634	10.5501
0.32	2.00913	10.5361
0.33	2.07191	10.521

Projectile Motion with launch angle 30 degrees.

Time	Distance	Height
0.52	2.92717	10.8637
0.53	2.98346	10.8447
0.54	3.03975	10.8247
0.55	3.09604	10.8037
0.56	3.15233	10.7818

The projectile problem from Labs 3 and 4 is repeated below for your reference:

This problem deals with equations dealing with motion of objects in two dimensions. In particular we will look at projectile motion as discussed in the web resource:

<http://www.glenbrook.k12.il.us/gbssci/phys/Class/vectors/u3l2b.html>. We are interested in the height and distance of the projectile with reference to time. The formulae for these two are readily available at another web resource: <http://www.egwald.com/geometry/projectile3d.php> and are repeated below. Assume that the projectile is fired from a gun that is mounted on a cliff above the sea.

z_0 = height of the gun above the x-y plane in meters.

v_0 = muzzle velocity = initial velocity of the projectile in meters/sec.

alpha = the angle between the horizontal (the x-y plane) and the muzzle of the gun in radians.

$z(t)$ = the height of the projectile t seconds after being fired (meters).

$r(t)$ = the distance of the projectile from the gun after t seconds (meters).

g = pull of gravity (9.81 meters/sec).

Then the parametric equations of motion of ($r(t)$, $z(t)$) are:

$$r(t) = v_0 * \cos(\alpha) * t$$

$$z(t) = -1/2 * g * t^2 + v_0 * \sin(\alpha) * t + z_0$$

You will input z_0 , v_0 , and alpha and compute distance and height of the at various time intervals until the projectile reaches ground. Your program should repeatedly compute the distance and height and output them in a tabular

