COMPUATIONAL PHYSICS
PHZ 4151/5156

Homework 3
Due in class, Tuesday, February 5.

(1) Read all of Chapter 6

(2) [Assignment 6 from chapter 5.15: *"Writing a first Program"*]

Write a program that computes all factorials 1!, 2!, 3!, ... up to a number input from the keyboard. (Hint, look at assignment 5 from chapter 5.15 to obtain the proper format of the for loop that is required in your program.) Part I: declare all variables to be int. Enter the number 17 from the keyboard and, using the fact that an int is a 4-byte quantity in the gnu compiler, explain the output values by the program and include your explanation (which should include a manual calculation of the values) with your answer.
Part II: Now show that, by changing a single variable declaration from int to double, you can reproduce the correct result for the first 17 factorials.

(3) [Adapted from assignment 7 of chapter 5.15]

Type in and run the following program, which calculates the magnitude of the gravitational field of a point particle with a mass equal to the earth's mass and displays the result as either a contour plot, a color graph, or a three-dimensional line plot. [Hint, see the *splot* command in *gnuplot*.]

```
// Gravitational field of a point earth.
// Illustrates simple 3-dimensional
// plotting routines.

#include<iostream>
using namespace std;

const double km = 1000;  // Note this procedure I
                         // An alternate naming
                         // convention capitalizes
                         // all global constants
```

```
const double gravitationalConstant = 6.67e-11;
const double earthMass = 5.97e24;
const double earthRadius = 6380 * km;

const int matSize = 20;        // This MUST be declared a const int
                               // to be used in the subsequent
                               // type statements !

double gravitationalField (double aX, double aY) {
        return gravitationalConstant * earthMass
                / (aX * aX + aY * aY);
}


int main() {
   double position[matSize];          // x and y coordinate space
   float offset = matSize / 2 - 0.5;  // Determines the starting point of the grid

   for (int loop = 0; loop < matSize; loop++)
      position[loop] = 0.1 * earthRadius * (loop - offset);

  float x, y;
  for (int outerLoop = 0; outerLoop < matSize; outerLoop++) {
    x = position[outerLoop];
    for (int innerLoop = 0; innerLoop < matSize; innerLoop++) {
      y = position[innerLoop];
      cout << x << " " << y << " " << gravitationalField(x,y) << endl;
    }
  }
}
```

By introducing an appropriate if statement in the gravitationalField function, redesign the above program so that it instead yields the gravitational field magnitude of the actual earth. (One way of doing this is to employ more than one return statement in the function.) Recall that the gravitational field for the earth is $G M_E r / r_E^3$ for $r < r_E$, where r_E is the earth's radius. Hand in your program together with surface and contour plots.