Physics 2200 Midterm I Project

- The goal of the programming project is to implement two algorithms that calculate π, and compare their performance:

  1. Leibniz series (discovered in 1668 by James Gregory):

  $$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots\right) = 4\sum_{i=0}^{\infty} \frac{(-1)^i}{2i+1} \tag{1}$$

  2. Bailey-Borwein-Plouffe series (discovered in 1995):

  $$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right) \tag{2}$$

- *Reproducible research* is a concept relevant for a data analysis or computational project. It requires that the data and the computer code which went into doing that analysis be available to others so that they might examine what you've done and reproduce your findings.

  In the spirit of reproducible research approach, your code must be 'compilable' (using make) and 'runnable' and produce the results without any grader's intervention.

- You are welcome to discuss the project's ideas with others in order to better understand them but the work you turn in must be your own. In particular, you must write your own code, run your own calculations, and communicate and explain the results in your own words.

Name:

Date:

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	5	5	0	10	5	10	10	5	5	55
Score:										

Suggested project steps:

1. (5 points) Write a C function with the following declaration

```
    double pi_leibniz (int n);
```

   that calculates π by summing the Leibniz series, Eq. (1). The parameter of the function is the number of the terms in the series to keep.

2. (5 points) Write a C function with the following declaration

```
    double pi_bbp (int n);
```

   that calculates π by summing the Bailey-Borwein-Plouffe series, Eq. (2). The parameter of the function is the number of the terms in the series to keep.

3. (0 points) Talk to the instructor and and get the approval of your code written so far.

4. (10 points) Write a C program that loops over the value of the parameter n mentioned above, starting from $n = 1$, and doubling it at every iteration. At each iteration print the number of terms in the series, the result produced by your function, and the absolute error of the result. Stop iterations when you have enough terms in the series, to produce the absolute error less than $tol = 10^{-6}$.

```
    i = 1;
    do
    {
        pi = pi_leibniz (i);
        abserr = fabs (pi - M_PI);
        printf ("%8d   %20.15f   %20.15f\n", i, pi, abserr);
        i *= 2;
    }
    while (abserr > tol);
```

   Record the number of terms of n_1 here: ____________________

5. (5 points) Contunue expanding your program by coding the same for the second algorithm.

   Record the value of n_2 here: ____________________

6. (10 points) Continue expanding your program by adding code that measures the time per function call of the two functions. Conduct calculations using your values of n_1 and n_2. To increase the precision of the time measurements repeat functions calls multiple times such that the total running times for each algorithm is between 1 and 2 seconds.

```
    count = 1000;
    do
    {
        timer_start ();

        // place your code to time here

        time = timer_stop ();
        time1 = time / count;
        printf (" %10.2f usec     %10.6f sec    %10d\n",
            time1 * 1.e6, time, count);
        /*
         * adjust count such that cpu time is between
         * tmin and tmax
         */
        count = adjust_rep_count (count, time, tmin, tmax);
    }
    while ((time > tmax) || (time < tmin));
```

The code for timing and adjusting the value of `count` is provided on the class website.

Record time per function call for both algorithms, t_{Leibniz} and t_{BBP}, as well as the ratio $t_{\text{Leibniz}}/t_{\text{BBP}}$ (rounded to the nearest integer).

t_{Leibniz}: ______________ t_{BBP}: ______________ $t_{\text{Leibniz}}/t_{\text{BBP}}$: __________

7. (10 points) Your C code should be well commented, written elegantly, and properly and consistently formatted.

8. (5 points) Create a Github repository for your project, call it m1, and upload there all your C code, your Makefile, and .gitignore and .indent.pro files.

   The URL of your repository: https://github.com/ ______________________________

9. (5 points) Chose a license for your project and create README.md file that is written in markdown and describes the goals and the results of your project.