IISER THIRUVANANTHAPURAM

PHY 423/623: Computational Techniques & Programming languages

School of Physics, IISER Thiruvananthapuram

Vasanth, 2018

Problem Set 1: Due Jan 14, 2018

Note: Name the programs as 'qnxx_yourname.cpp' where 'xx' is the question number and 'yourname' is your own name. Put all the files a folder named as your name and upload it in the shared google drive folder.

1. Write a program that prints out a table with Fahrenheit degrees 0, 10, 20, . . . , 100 in the first column and the corresponding Celsius degrees in the second column.

2. Write a program that generates all odd numbers from 1 to n. Set n in the beginning of the program and use a *while loop* to compute the numbers.

3. Write a program that asks the user to enter the number of seconds as an integer value (use type long, or, if available, long long) and that then displays the equivalent time in days, hours, minutes, and seconds. Use symbolic constants to represent the number of hours in the day, the number of minutes in an hour, and the number of seconds in a minute.The output should look like this:

   ```
   Enter the number of seconds: 31600000
   31600000 seconds = 365 days, 17 hours, 46 minutes, 40 seconds
   ```

4. Write a program that prints a nicely formatted table of t and $y(t)$ values, where

   $$y(t) = ut - \frac{1}{2}gt^2$$

   $u = 10$ m/s and g is acceleration due to gravity. Use 20 equally spaced t values through the interval $[0, 2u/g]$.

5. The Gaussian function is given by,

   $$f(x) = \frac{1}{2\pi a} \exp\left[-\frac{1}{2}\left(\frac{x-m}{a}\right)^2\right]$$

   Create a user defined function to evaluate it and call the function to calculate its values for $-10 \leq x \leq 10$ in steps of $\Delta x = 0.01$. Take $m = 0$, $a = 2$. Print the table of x vs $f(x)$ values into a file named 'gaussian.txt'.

6. Wriate a program where the main() calls a user defined function to calculate the dot product of two vectors. The vectors should be declared in the calling function.

7. Write a program that uses an array of char and a *loop* to read one word at a time from input until the word done is entered.The program should then report the number of words entered (not counting done). A sample run could look like this:

```
Enter words (to stop, type the word done):
anteater birthday category dumpster envy finagle geometry done for sure
You entered a total of 7 words.
```

You should include the cstring header file and use the strcmp() function to make the comparison test.

8. The Ising model is defined as,

$$\mathcal{H} = J \sum_{\langle i,j \rangle} S_i S_j$$

where the sum runs over nearest neighbour sites i and j. S_i-s are classical spin variable which can take values ± 1. J is the exchange interaction. Here assume $J = 1$. The thermal average of a quantity A at inverse temperature β $(1/k_B T)$ is given by,

$$\langle A \rangle = \frac{\sum_s A_s e^{-\beta E_s}}{Z}, \quad Z = \sum_s e^{-\beta E_s}$$

where s denote a basis state configuration of spins and the sum is over all possible states. A_s is the value of the physical quantity corresponding to state s and E_s is the energy of the state. **Consider a one dimensional lattice with periodic boundary condition**.

Write a C++ program to evaluate avarage values of energy and absolute net magnetization ($|m|$) at a given $\beta \in (0,\ 1.0)$. The program should read the values for number of lattice sites and β from input. You will also use the `clock()` function declared in `ctime` header file to calculate the execuation time. Start the `clock()` after reading the input and calculate the program execuation time after that point. In the output, print the energy, magnetization and the time taken. A sample program run can look like this:

```
Enter the lattice size = 4
Enter the value of beta = 0.5
Result:
Total number basis states = xx
Average energy = xx
Average magnetization = xx
Time taken = xx sec
```

Note: You will have to generate all the possible basis states or spin configurations for a given number of lattice sites using a general algorithm. *Do not* store the basis states. Generate it in a loop and calculate required quantities on the fly.

Try to organize the code neatly by modularizing the whole calculations into smaller parts and implementing each part by writing separate functions.

—○—