GRADING STRATEGY FOR ASSIGNMENT - I, PDS LAB, AUTUMN 2015

This documents the grading strategy for the first assignment in Data and File Structures Laboratory (aka PDS Lab), Autumn 2015.

Grading:

Suppose a problem carries 100 marks.
A. Correct Output: 80
B. Efficient Solution: 20

Linux assignment:
Problem 1: 5 (4+1)
Problem 2: 10 (8+2)
Problem 3: 15(12+3)

C assignment:
Problem 1: 25 (20+5)
Problem 2: 25 (20+5)

Model solution for each of the problems are enclosed with this document. These solutions follow the best algorithm (based on the course topics discussed so far), and will be the basis for grading B.

For any clarification, please contact me after Wednesday lecture.

-- Ashwin

```
#!/bin/bash
#
# Model solution for problem 1 (a-c), Assignment 1, PDS Lab, Autumn 2015
#

# Q1
wc -l apj.txt

# Q2
grep -wic "President" apj.txt

# Q3
sed -i '1 i Former President Dr. APJ Abdul Kalam No More\n' apj.txt

# Alternate solution
# echo -e "Former President Dr. APJ Abdul Kalam No More\n" |\
# cat - apj.txt > a.txt
# mv a.txt apj.txt
```

```c
/*****************************************************************************

      Model solution for problem 2a, Assignment 1, PDS Lab, Autumn 2015

*****************************************************************************/
#include<stdio.h>

int main()
{
      unsigned int i, t;
      double rn=1.0, rn1=1.0;
      printf("\nPlease enter a positive integer strictly greater "
               "than 2 (press CTRL+C to exit):\t");
      while(1)
      {
            scanf("%u",&t);/* Get user input */
            if(t > 2) break;/* Break if valid input */
            else printf("\nYou have entered an invalid number. "
                        "Please try again.\nPlease enter a "
                        "positive integer strictly greater "
                        "than 2 (press CTRL+C to exit):\t");/* Else ask for
                                                               a new input */
      }
      /* The logic follows from a simple observation:
       * f_n = f_{n-1} + f_{n-2} where f_n is the n-th fibonacci term.
       * f_n/f{n-1} = 1 + 1/(f{n-1}/f{n-2})  for n > 2.
       * r_n = 1 + r{n-1}    where r_n is the ratio of n-th term to
       *                     (n-1)-th term.
       * Observe that the ratio converges to the golden ratio
       * phi=1.61803398...
       *
       * Also note that r_3=1.0
       */
      for(i=3;i<=t;i++)
      {
            rn1 = 1.0 + 1.0/rn;/* Compute the ratio of (i+1)-st term
                                  to i-th term */
            if(rn1 == rn) break;/* Ratio converges to golden ratio.
                                   No need to compute further. */
            else rn = rn1;/* Update ratio for next term computation */
      }
      printf("\nThe ratio of fibonacci terms "
            "%u and %u is:\t%0.20lf\n",t+1, t, rn);/* Print the ratio */
      return 0;
}
```

```
/*****************************************************************************

      Model solution for problem 2b, Assignment 1, PDS Lab, Autumn 2015

*****************************************************************************/
#include<stdio.h>

int main()
{
      float x1, y1, x2, y2, x3, y3;
      double curl;
      int i, n;

      printf("\nPlease enter whitespace separated x1 and y1:\t");
      scanf("%f%f",&x1,&y1);
      printf("\nPlease enter whitespace separated x2 and y2:\t");
      scanf("%f%f",&x2,&y2);
      printf("\nPlease enter the no.of points to be checked:\t");
      scanf("%d",&n);

      for(i=0;i<n;i++)
      {
            printf("\nIteration %d:\n",i+1);
            printf("Please enter whitespace separated "
                  "x%d and y%d:\t",3+i,3+i);
            scanf("%f%f",&x3,&y3);

            /* The logic follows from a simple observation on the sign of
             * curl of two vectors.
             * left turn     if curl is +ve
             * right turn    if curl is -ve
             * straight      otherwise
             */
            curl = (x2-x1)*(y3-y2) - (x3-x2)*(y2-y1);/* Compute curl of
                                                        vector ((x2-x1),
                                                               (y2-y1))
                                                        and ((x3-x2),
                                                             (y3-y2)) */
            if(curl > 0) printf("Left\n");/* +ve curl, print left */
            else if(curl < 0) printf("Right\n");/* -ve curl, print right*/
            else printf("Straight\n");/* 0 curl, print straight */

            x1=x2;           /* Update coordinates for next iteration */
            y1=y2;
            x2 = x3;
            y2 = y3;
      }
      return 0;
}
```