DFS LAB – ASSIGNMENT 1

MTech(CS) I year 2019–2020

Deadline: 10 August, 2019

Total: 60 marks

SUBMISSION INSTRUCTIONS

1. Naming convention for your programs: `cs19xx-assign1-progy.c`
2. To submit a file (say `cs19xx-assign1-progy.c`), go to the directory containing the file and run the following command from your account on the server (IP address: 192.168.64.35)

   ```
   cp -p cs19xx-assign1-progy.c ~dfslab/2019/assign1/cs19xx/
   ```

   If you want to submit your files from a computer with a different IP address, run

   ```
   scp -p cs19xx-assign1-progy.c \
   mtc19xx@www.isical.ac.in:/user1/perm/pdslab/2019/assign1/cs19xx/
   ```

   (enter your password when prompted).

   To submit all `.c` and `.h` files at one go, use `cp -p *.c *.h ~dfslab/2019/assign1/cs19xx/` or similar.

NOTE: All programs should take the required inputs from stdin, and print the desired outputs to stdout.

Q1. Suppose we define the angle between a pair of hands (denoting hour, minute, and second) of a clock as the lowest of the two possible angles. Let the hands are said to be in EQUIANGULAR position if the angles between them are the same. Further assume that the hands of a clock are said to be in SEMI-EQUIANGULAR position if the angles between any two pairs of hands are the same. Write a program that takes the time as user input (in the 24-hour format) and returns whether the clock is in EQUIANGULAR, SEMI-EQUIANGULAR, or NONE of the above positions. Consider that there is a tolerance level to ensure whether a pair of angles are *same* or not. Given the two angles α and β, they are said to be *same* for a tolerance level δ if $|\alpha - \beta| \leq \delta$. [15 marks]

Input Format
The input (to be read from stdin) is three positive integers h, m, and s representing the time in hours, minutes, and seconds, respectively, which are separated by colons. This is followed by the tolerance level of angle δ (in radian).

Output Format
The output (to be printed to stdout) will show whether the hands of the clock are in position EQUIANGULAR, SEMI-EQUIANGULAR, or none.

Sample Input 0
12:00:00 0

Sample Output 0
EQUIANGULAR

Sample Input 1
2:00:05 0

Sample Output 1
NONE

Sample Input 2
2:00:05 0.1

Sample Output 2
SEMI-EQUIANGULAR

Q2. Given a set of distinct positive integers A as user input, write a program to partition A into an arbitrary number of distinct and non-null subsets $A_1, A_2, \ldots, A_n$ ($n \geq 2$, $\bigcup_{i=1}^{n} A_i = A$, $A_i \neq \emptyset$ and $A_i \cap A_j = \emptyset$) such that the following function is maximized.

$$\mathcal{F}(A_1, A_2, \ldots, A_n) = \mathcal{BAND}(\mathcal{BXOR}(A_1), \mathcal{BXOR}(A_1), \ldots, \mathcal{BXOR}(A_n))$$

Note that, $\mathcal{BAND}()$ and $\mathcal{BXOR}()$ represent the results of bitwise AND of bitwise XOR operations applied pairwise on a set of numbers, respectively. In case of a tie, return all the optimal results. [25 marks]

Input Format
The input is a set of integers taken from the standard input.

Output Format
The output is the partition of integers separated by -1 followed by the computed maximum value of $\mathcal{F}()$ in the next line.

Sample Input 0
1 2 3

Sample Output 0
1 2 -1 3
3

Sample Input 1
1 2 3 4

Sample Output 1
1 2 -1 3 4
1 2 4 -1 3
3

Sample Input 2
2 4 8 16

Sample Output 2
2 -1 4 -1 8 -1 16
2 4 -1 8 16
2 8 -1 4 16
2 16 -1 4 8
2 4 8 -1 16
2 4 16 -1 8

2 8 16 -1 4
4 8 16 -1 2
0

Q3. Suppose you are given with a sequence of n integers. Write a program that will find out a pair of indices i and j ($j \geq i$) such that the sum of the values in the sub-sequence starting at index i and ending at index j (both inclusive) is maximum. Consider the starting index to be 0. You need to output both the indices and the maximum value of summation. In case, there are multiple subsequences possible as the optimal answer, return the shorter subsequence. If there is a further tie return all the results. [20 marks]

Input Format
The sequence of integers taken from the standard input

Output Format
The pair of indices for which the summation is maximum followed by the summation value in a new line.

Sample Input 0
-2 11 -4 13 -5 2

Sample Output 0
1 3
20

Sample Input 1
1 -3 4 -2 -1 6

Sample Output 1
2 5
7

Sample Input 2
1 -1 1 -1 1 -1 -1 -1

Sample Output 2
0 0
2 2
4 4
1