Programming Design, Spring 2015
Homework 5

Instructor: Ling-Chieh Kung
Department of Information Management
National Taiwan University

To submit your work, please upload a PDF file for Problem 1 and two CPP files for Problems 2 and 3 to PDOGS at http://pdogs.ntu.im/judge/. Each student must submit her/his individual work. No hard copy. No late submission. The due time of this homework is ***8:00am, April 6, 2014***. Please answer in either English or Chinese.

Before you start, please read Sections 5.20–5.22 and Chapter 19 of the textbook.[1] The TA who will prepare the solution for this homework is ***Tammy Chang***.

Problem 1

(10 points) Consider the following recursive function

```
int f(int n, double m, int a, int b)
{
  if(n == 1)
  {
   if(m > 1)
      return f(1, m / 2, a, b) + b;
    else
      return 1;
  }
  else
    return f(n - 1, m, a, b) + a;
}
```

(a) (2 points) What is the value of `f(50, 40, 1, 2)`? Just write down a single number; no explanation needed.

(b) (5 points) Mathematically, what is the value of $f(n, m, a, b)$ as a function of n, m, a and b? You may assume that $n \geq 1$, $n \in \mathbb{Z}$, and $m > 1$.

(c) (3 points) Try to replace `n - 1` by `n--` in the program. Now, once you invoke `f(50, 40, 1, 2)`, you get an infinite loop. Explain why.

Problem 2

(45 basic points and 10 bonus points) Consider the following two-player game. There are 9 coins on the table. Players 1 and 2 take turns to remove 1, 2, or 3 coins. The player who removes the last coin is the loser. If you can choose to be player 1 (the leader, who removes coins first) or player 2 (the follower, who removes coins after the leader), what should you do?

The answer to the above question is somewhat standard. We can see that player 2 has a *dominant strategy* that guarantees her victory: Once player 1 removes k coins, player 2 should removes $4-k$ coins. Then exactly 5 coins will be left after the first round and exactly 1 coin will be left after the second round. Then player 1 must remove the last coin, and player 2 wins.

What if today there are 10 coins? Then you should choose to be player 1. All you need to do is to first remove 1 coin. Then there are 9 coins left, and player 2 becomes the "leader" in the remaining

[1] The textbook is *C++ How to Program: Late Objects Version* by Deitel and Deitel, seventh edition.

game. Player 1 then has a strategy to win. Technically, we call the 9-coin remaining game a *subgame* of the original 10-coin game. Given a game with $n \geq 10$ coins, what really matters is to create a 9-coin subgame and become the follower in that subgame. The same idea applies to similar games where the number of coins that each player must remove in each turn is not 3.

In this problem, you will play this game with a virtual opponent. You will be given n coins, and the maximum number of coins that a player can remove in a turn is m. Both n and m are positive integers while $n \leq 10000$ and $2 \leq m \leq n-1$. The bad news is that you are player 2. The good news is that $n = (m+1)k+1$ for some $k \in$ N, which means that you can definitely win the game. What player 1 will do has been predetermined and will be provided to you in the input files. All you need to do is to follow the above strategy to win. Your task is to print out the full history of each given game by indicating the player that removes any given coin.

For example, suppose in a given game $n = 26$ and $m = 4$. Moreover, you are given that player 1 will repeatedly remove 1, 2, 3, and 4 coins in his turn. Then your program should output

```
1 2 2 2 2 1 1 2 2 2 1 1 1 2 2 1 1 1 1 2 1 2 2 2 2 1
```

This means that player 1 removes coins 1, 6, 7, 11, 12, 13, 16, 17, 18, 19, 21, and 26, and player 2 removes all other coins. Note that player 1's strategy is described as a few integers $(x_1, x_2, ..., x_t)$, where in round i he will remove x_p coins if $p-1$ equals the remainder of dividing $i-1$ by t.

Input/output formats

There are 15 input files. In each file, there is a line of positive integers n, m, t, x_1, x_2, ..., and x_t. n is the number of coins in this game, m is the maximum number of coins to be removed by a player in each turn, t is the number of values in player 1's strategy, and $(x_1, ..., x_t)$ forms player 1's strategy as described above. Each two values are separated by a white space. You may assume that the $n \leq 10000$, $2 \leq m \leq n-1$, $t \geq 1$, and $n = (m+1)k+1$ for some $k \in$ N.

Given the input file, you output n values y_1, y_2, ..., and y_n in a single line in your output file to represent the game history, where $y_i = j$ means that coin i is removed by player j, $i \in \{1, ..., n\}$, $j \in \{1, 2\}$. Each two values should be separated by a white space.

What should be in your source file

Your .cpp source file should contain C++ codes that will both read testing data and complete the above task. For this problem, you are NOT allowed to use techniques not covered in lectures. You should write relevant comments for your codes.

Grading criteria

- 30 points for this program will be based on the correctness of your output. PDOGS will compile your program, feed testing data into your program, and check the correctness of your outputs. Each correct output gives you 2 points.
- 15 points for this program will be based on how you write your program, including the logic and format. Please try to write a robust, efficient, and easy-to-read program.
- The game has a natural recursive structure: The game history of a game contains the game history of a subgame. Therefore, given a game, one may first decide what players 1 and 2 will do in one round, create a subgame (with a smaller number of remaining coins), and then repeat the process (by invoking a recursive function).

  10 bonus points may be earned by solving this problem by recursion. All you need to do is to upload a recursion-based program to PDOGS. If, and only if, you get 45 points on PDOGS with your recursion-based program, you will get 10 bonus points.[2]

[2]These 10 points will not show up on PDOGS; TAs will add these 10 points to you on CEIBA manually.

Problem 3

(45 points) Given two sorted lists of positive integers A and B, let's merge them together!

Input/output formats

There are 15 input files. In each file, there are two lines of integers. The first line contains x_1, x_2, ..., x_n, and -1, where $x_i > 0$ for all $i = 1, ..., n$. x_is form the first sorted list. You know that either $x_1 \leq x_2 \leq \cdots \leq x_n$ or $x_1 \geq x_2 \geq \cdots \geq x_n$, but you need to read the value to see the order by yourself. The last integer -1 marks the end of the list. The second line contains y_1, y_2, ..., y_m, and -1, where $y_i > 0$ for all $i = 1, ..., m$. y_is form the second sorted list. Again, you need to determine whether y_is are in the ascending or descending order by yourself. It is possible that x_is and y_is are sorted in the opposite directions. In each line, each two values are separated with a white space. You may assume that n and m are both no greater than 1000.

Given the input file, you output $n + m$ values z_1, z_2, ..., and z_{n+m} such that $z_1 \leq z_2 \leq \cdots z_{n+m}$ in a single line in your output file as the merged list. Each two values should be separated by a white space.

As an example, suppose that the input file contains

```
1 3 5 7 9 -1
2 4 5 10 -1
```

in two lines, then the output file should contain

```
1 2 3 4 5 5 7 9 10
```

in one single line.

What should be in your source file

Your .cpp source file should contain C++ codes that will both read testing data and complete the above task. For this problem, you are NOT allowed to use techniques not covered in lectures. You should write relevant comments for your codes.

Grading criteria

30 points for this program will be based on the correctness of your output. PDOGS will compile your program, feed testing data into your program, and check the correctness of your outputs. Each correct output gives you 2 points.

15 points for this program will be based on how you write your program, including the logic and format. Please try to write a robust, efficient, and easy-to-read program.