Irish Collegiate Programming Competition 2018

Problem Set

University College Cork ACM Student Chapter

March 10, 2018

Instructions

Rules

- All mobile phones, laptops, and other electronic devices must be powered off and stowed away for the duration of the contest.
- The only networked resource that teams are permitted to access is the submission system.
- If a team discovers an ambiguity or error in a problem statement, they should tell an organiser. If the organisers agree that an ambiguity or error exists, a clarification will be issued to all teams.
- No multi-threading is allowed, and no sub-processes.
- No file input/output is allowed. All input to your program will be provided via standard input (stdin) and all output should be printed to standard output (stdout). Examples of how to do this in each of the languages is provided in the resources section of the submission site.

Submission Instructions

- The submission system URL is: `143.239.81.61:8596`
- Your password will be provided by the organisers. Please notify an organiser if you are unable to log in.
- Submissions should consist of a single source file, **not a compiled executable**.
- To submit, click the "Submit a Solution" link, complete the submission form, upload your source file, and click "Save". Your submission should now be listed in your submission queue.
- Java solutions should be a single source file and should not include the package header. The testing script will also be renaming your file to `Pn.java` (where n is the problem number) and as such the main class for problem 1 should be defined as: `public class P1 {...}` and will be called with: `java P1 < input-file`

- C and C++ submissions will be compiled with the flags `-lm -O2`.
- `C#` submissions will be run using mono since the test environment is running on Linux.
- Haskell submissions will be compiled with the flag `-O2`.

Testing and Scoring

- Solutions should be submitted in the form of source code, which will be compiled on the test environment. A solution will be accepted if it compiles on the test environment and produces the correct output for the sample inputs given in the question.
- The output from your program should be terminated by a new line and should not contain any additional whitespace at the beginning or end of lines.
- A solution will be tested against 10 separate test cases which are designed to test the limits and corner cases of the inputs. One point is given for each correct output.
- Programs are limited to one second of CPU time and 256MB of RAM.
- If a solution is submitted while an earlier solution to the same problem is still in the queue, the earlier submission will not be tested. The earlier submission will be marked with the message: "Old, not going to be tested".
- In the event of a tie, the winning team will be the one whose last scoring submission was submitted earliest.

1 Silly Sequence Sort

A silly sort of a sequence $\mathcal{X}$ of N integers consists in sorting the sequence such that all even-positioned integers are sorted in ascending order and all odd-positioned integers are sorted in descending order, see (1). Furthermore, for every position, the two neighbours need to be either both smaller or both bigger, see (2).

It is easy to see that such a sort puts the first smallest integer in the first position and the first largest in the second position. The third position contains the second smallest and the fourth the second largest, and so on.

More formally, if the sequence $\mathcal{X}' = \mathcal{X}_0, \ldots, \mathcal{X}_{N-1}$ is obtained by applying silly sort to sequence $\mathcal{X}$, it satisfies the following conditions:

$$\forall i \in \{0, \ldots, N-1\} \begin{cases} \mathcal{X}_i \leq \mathcal{X}_{i+2} & \text{if } i \text{ is even} \\ \mathcal{X}_i \geq \mathcal{X}_{i+2} & \text{if } i \text{ is odd} \end{cases} \tag{1}$$

and (2)

$$\mathcal{X}_0 \leq \mathcal{X}_1 \geq \mathcal{X}_2 \leq \mathcal{X}_3 \geq \ldots \tag{3}$$

For instance, the sequence $\langle 10, 2, 5, 9, 1, 3 \rangle$ would be rearranged as $\langle 1, 10, 2, 9, 3, 5 \rangle$. Provided an arbitrary sequence of integer numbers, your goal is to sort the sequence accordingly.

Input The first input line contains a single integer N, $0 < N \leq 500,000$, corresponding to the length of the sequence. The following line contains N space-separated integers with values between $-500,000$ and $500,000$ inclusive.

Output The output line should be a list of space-separated integers providing the input sequence sorted according to the rules of Silly Sort.

Sample Input 1

```
6
10 2 5 9 1 3
```

Sample Output 1

```
1 10 2 9 3 5
```

Sample Input 2

```
3
1 0 -1
```

Sample Output 2

```
-1 1 0
```

2 Delightful Downhill-skiing Dinner

Skiing competitors in the Winter Olympics are known to be vicious. There are many conflicts between individual athletes. Thus, the seat plan for a banquet dinner for all Olympic athletes requires careful planning. Luckily, there are two big round tables ordered for the dinner.

Your task is to split the athletes into two conflict-free groups, so that each of the two tables enjoys a peaceful dinner. All conflicts between athletes are captured in a square symmetric matrix M. If the entry in row i and column j is 1, then athletes i and j have a dispute and should not sit at the same table. This also means that the entry in row j and column i is 1, since a dispute is always symmetric. All other entries of the matrix are 0. A conflict-free table is a set of athletes $a_1, \ldots, a_k$ such that $M(a_i, a_j) = 0$ for all $i, j \in \{0, \ldots, k\}$.

Thus the following matrix of conflicts between athletes results in a split with athletes 0, 2, 4 at one table and athletes 1, 3 at the other.

```
0 1 0 1 0      Athlete 0 is in dispute with athlete 1 and 3.
1 0 1 0 1      Athlete 1 is in dispute with athlete 1, 2 and 4.
0 1 0 1 0      Athlete 2 is in dispute with athlete 1 and 3.
1 0 1 0 1      Athlete 1 is in dispute with athlete 1, 2 and 4.
0 1 0 1 0      Athlete 4 is in dispute with athlete 1 and 3.
```

Input The input consists of a single integer N in the first line such that $1 \leq N \leq 2000$ which corresponds to the number of athletes present for dinner. The next N lines contain N space separated integers with value 0 or 1 providing the conflicts. This corresponds to the conflict matrix M as defined above. Note that every input of conflicts will have a possible solution of splitting athletes into two (non-empty) tables.

Output The output should consist of two lines providing the index of athletes sitting at the first and second table, respectively.

The first line corresponds to the table at which athlete 0 is seated and thus starts with integer 0. The second line contains the indices of athletes sitting at the other table.

All values on each of the lines should be given in ascending order and space-separated.

Sample Input 1

```
5
0 1 0 1 0
1 0 1 0 1
0 1 0 1 0
1 0 1 0 1
0 1 0 1 0
```

Sample Output 1

```
0 2 4
1 3
```

Sample Input 2

```
6
0 0 1 0 0 0
0 0 1 0 1 0
1 1 0 1 0 1
0 0 1 0 0 0
0 1 0 0 0 0
0 0 1 0 0 0
```

Sample Output 2

```
0 1 3 5
2 4
```

3 Desperately Dire Deck Deathtrap[1]

You eventually make your way into a dimly lit room. Thump! The door slams shut behind you! You desperately scramble for a door handle, to no avail; the door has now completely receded into the wall. You're trapped!

After a moment of panic, you are drawn to the only piece of furniture in the room: a rustic wooden desk, lit only by a flickering clerk's lamp, on which you find a deck of cards and what looks like a set of instructions.

Intrigued, you inspect the deck. It's composed of N cards. They have a nice gloss finish, which makes them easy to handle. On each card, however, no suit to be seen; only a unique natural integer ($0 \leq i < N$), printed in an ornamented font, representing the card's value. The deck is complete, i.e., it contains all integers from 0 to $N - 1$.

Your attention now turns to the instruction sheet:

> Sort the deck in such a way that, by
>
> 1. drawing one card from the top of the deck and placing it on the desktop,
> 2. moving the next card from the top to the bottom of the deck,
> 3. repeating steps 1 & 2 until the deck is exhausted,
>
> you end up laying all the cards on the desktop in ascending order of their values. And make haste, for your iron shroud shall not wait!

As soon as you finish reading the instructions, you hear an ominous click... You realise with horror that the room has started shrinking around you! Your survival now hinges on your sorting skills! Will you solve the puzzle and make it out alive before getting crushed?

Input The input consists of a single line containing a single integer N, such that $1 \leq N \leq 200,000$.

Output The output should consist of a single line containing the N space separated card values sorted in the desired order from the top to the bottom of the deck.

Sample Input 1

```
3
```

Sample Output 1

```
0 2 1
```

Sample Input 2

```
6
```

Sample Output 2

```
0 3 1 5 2 4
```

[1] courtesy of Poppulo

4 Super Sub-Sequence

A super-sub sequence is a (possibly empty) consecutive sub-sequence of maximum weight. That is, for a (non-empty) sequence of numbers $a_1, \ldots, a_N$, a consecutive sub-sequence $a_i, a_{i+1}, \ldots, a_j$ with $1 \leq i \leq j \leq N$ is a super sub-sequence if its weight $a_i + a_{i+1} + \cdots + a_j$ is maximum among all other consecutive sub-sequences. Similarly, the empty sequence, which has weight 0, is a super sub-sequence if all other consecutive sub-sequences have a weight ≤ 0. For example, the sequence

-17 40 -24 24 57 -80 -81 -83 70 -39

has the super sub-sequence 40 -24 24 57 of total weight of 97 since all other sub-sequences have a weight ≤ 97. Your task is to find the weight of a super sub-sequence.

Input The input consists of two lines. The first line contains a single integer N such that $1 \leq N \leq 10,000$, which corresponds to the length of the sequence. The second line contains the N space separated integers of the sequence, with values between $-10,000$ and $10,000$.

Output The output should consist of a single line containing one integer: the weight of a super sub-sequence.

Sample Input 1

```
10
-17 40 -24 24 57 -80 -81 -83 70 -39
```

Sample Output 1

```
97
```

Sample Input 2

```
10
-4 -35 36 0 32 17 -22 16 23 32
```

Sample Output 2

```
134
```

5 Crazy Climbing Countdown[2]

You're given a list of positive integers. A *target integer* is said to be *reachable* by those integers if, by combining one or more integers from the list using addition, subtraction, multiplication, division, and parentheses, you can construct an expression whose value is the target.

Each integer in the list can be used at most once in the expression, and all numbers involved, including intermediate results (if any), must be positive integers. In particular, the use of negative numbers, zero, and fractions (such as $10 \div 3$) is not allowed.

The question is: what is the smallest positive (> 0) target integer that is *not* reachable by the integers given?

Input The input consists of two lines. The first line contains a single integer n such that $1 \leq n \leq 6$, which corresponds to the number of integers given. The second line consists of the n integers in question separated by spaces, each integer being in the set $\{1 \ldots 10, 25, 50, 75, 100\}$.

Output The output should consist of a single line containing a single integer: the smallest positive target integer that is not reachable by the integers given.

Sample Input 1

```
2
5 3
```

Sample Output 1

```
1
```

Sample Input 2

```
4
25 75 2 1
```

Sample Output 2

```
10
```

[2] courtesy of Poppulo

6 Relatives Recognition Row

Adam, Bob, and Carl are three very competitive brothers. Carl has a successful career in Software Development but, being the youngest, is looked down upon by his older brothers, even though he is earning more money than them! All three brothers are planning to build houses and this is Carl's chance to finally show them by building his house as big, in terms of volume, as his brothers houses combined. However, Bob also has a secret agenda and wants to ensure his house is at least one cubic metre bigger than Adam's. A government effort to simplify regulations recently enforced that all new houses must be perfectly cubed and can only have integer length. After a bit of thinking, Carl figures out that it is never possible to build his house exactly as big as his brothers houses combined. However, depending on the sizes of his brothers houses, it is sometimes possible to build a house that has a volume which is exactly one cubic meter smaller than the volume of the other two houses combined, which he finds sufficiently big.

Carl only has a rough idea of how big his house should be and knows about Bob's plan to build a bigger house than Adam. The length of Carl's house will be between limits C_{min} and C_{max} (inclusive). He wants you to write a program to calculate the number of possible sizes of Adam's and Bob's houses, so that he can build his house to be exactly one cubic metre smaller than their houses combined.

As an example, assume that Carl wants his house to have a length between $C_{min} = 1$ and $C_{max} = 15$ metres. If Adam was to build a house with a length of 9 metres and Bob builds a house with a length of 10 metres, then Carl could build a house with a length of 12 metres. This achieves his goal by having a volume of $1728m^3$, whereas his brothers houses only have a volume of $729m^3$ and $1000m^3$, respectively. In fact, this is the only possible house sizes for Adam and Bob such that Carl can achieve his goal within the given range of $C_{min} = 1$ and $C_{max} = 15$ metres.

Input The input consists of one line containing two integers C_{min} and C_{max} such that $3 \leq C_{min} < C_{max} \leq 25,000$ corresponding to the minimum and maximum length of Carl's house.

Output The output should consist of one line containing a single integer corresponding to the number of possible house sizes of Adam and Bob such that Carl can build a house within the given length limits of C_{min} and C_{max} and such that the volume of Carl's house is exactly one cubic metre smaller than the volumes of Adam's and Bob's houses combined.

Sample Input 1

```
1 15
```

Sample Output 1

```
1
```

Sample Input 2

```
1 200
```

Sample Output 2

```
3
```

7 Cooperative Card Collection

Cooperative Card Collection is a game in which thirteen players try to build up a street of cards 2,3, ... , K, A out of a standard card deck with 52 cards. Each player is dealt four cards and has to select a single card out of his or her hand. The goal is that the players cooperate so that no two players put down a card with the same rank (suits can differ). Note that the playing cards are ordered by their natural rank. That is 2, 3, 4, . . . , J, Q, K, A where 2 has the lowest rank and A the highest.

It can be proven that, for any distribution of hands, the players can succeed in their goal. Your task is to select a card from each player's hand to achieve this goal, i.e. the selected cards together form the set 2,3, ... , K, A. Furthermore, every player aims at putting down the highest ranked possible card in his or her card. Since players put down cards in turn, the first player will select the highest ranked card in his or her hand with which it is possible to achieve the collective goal. After that, the second player puts down his or her highest ranked card with which it is still possible to achieve the collective goal, and so on.
Thus, for the hands

```
2  2  2  2
3  3  3  3
4  4  4  4
..........
K  K  K  K
A  A  A  A
```

we select the cards 2, 3, 4, . . . , K, A in this order, i.e. the first player puts down 2, the second player puts down 3 and so on.
For the hands

```
A  A  3  3
K  K  4  4
Q  Q  5  5
..........
3  3  A  A
2  2  2  2
```

the selection of cards will be A, K, Q, ... , 3, 2 since every player, starting from the first one, is supposed to put down the highest-ranked card in his or her hand with which it is possible to complete the selection.

Input The input consists of thirteen lines, each corresponding to a player. Each line consists of four space separated symbols in $\{2, 3, \ldots, J, Q, K, A\}$ corresponding to the four cards in the player's hand.

Output The output should consist of a single line containing thirteen symbols representing the cards played by the players in the order of their turn.

Sample Input 1

```
2 2 2 2
3 3 3 3
4 4 4 4
5 5 5 5
6 6 6 6
7 7 7 7
8 8 8 8
9 9 9 9
10 10 10 10
J J J J
Q Q Q Q
K K K K
A A A A
```

Sample Output 1

```
2 3 4 5 6 7 8 9 10 J Q K A
```

Sample Input 2

```
A A 3 3
K K 4 4
Q Q 5 5
J J 6 6
10 10 7 7
9 9 8 8
8 8 9 9
7 7 10 10
6 6 J J
5 5 Q Q
4 4 K K
3 3 A A
2 2 2 2
```

Sample Output 2

```
3 4 5 6 7 8 9 10 J Q K A 2
```