A. Requirements

Code (100%)

You can write your code in Java, Python, C, or C++. The *time limit* may vary among different languages, depending on the performance of the language. Your code must be a complete excutable program instead of only a function. We guarantee test data strictly compliance with the requirements in the description, and you do not need to deal with cases where the input data is invalid.

No AI Assistance or Plagiarism: All code must be your own. The use of AI tools (e.g., ChatGPT, GitHub Copilot) or copying from external sources or peers is **strictly forbidden**.

Violations of the plagiarism rules will result in 0 points or even **failure** of this course.

Libraries in this assignment:

- For C/C++, you can only include standard library.
- For Java, you can only `import java.util.*`
- For Python, you can only import standard library. In other words, you cannot import libraries such as `numpy`.

We provide an example problem to illustrate the information above better.

B. Example Problem: A + B Problem

Description

Given 2 integers A and B, compute and print A + B

Input

Two integers in one line: A, and B

Output

One integer: A + B

Sample Input 1

```
1 2
```

Sample Output 1

```
3
```

Problem Scale & Subtasks

For 100% of the test cases, $0 \leq A, B \leq 10^6$

Solutions

Java

```java
import java.util.*;

public class Example {
    public static void main(String[] args) {
        int a, b;
        Scanner scanner = new Scanner(System.in);
        a = scanner.nextInt();
        b = scanner.nextInt();
        scanner.close();
        System.out.println(a + b);
    }
}
```

Python

```
AB = input().split()
A, B = int(AB[0]), int(AB[1])
print(A + B)
```

C

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  int A, B;
  scanf("%d%d", &A, &B);
  printf("%d\n", A + B);
  return 0;
}
```

C++

```
#include <iostream>

int main(int argc, char *argv[])
{
  int A, B;
  std::cin>> A >> B;
  std::cout<< A + B << std::endl;
  return 0;
}
```

C. Submission

After finishing this assignment, you are required to submit your code to the Online Judge System (OJ), and upload your `.zip` package of your code files to BlackBoard.

C.1 Online Judge

Once you have completed one problem, you can submit your code on the page on the Online Judge platform (`oj.cuhk.edu.cn`, campus only) to gain marks for the code part. You can submit your solution of one problem for **no more than 80 times**.

After you have submitted your program, OJ will test your program on all test cases and give you a grade. The grade of your latest submission will be regarded as the final grade of the corresponding problem. Each problem is tested on multiple test cases of different difficulty. You will get a part of the score even if your algorithm is not the best.

Note: The program running time may vary on different machines. Please refer to the result of the online judge system. OJ will show the time and memory limits for different languages on the corresponding problem page.

If you have other questions about the online judge system, please refer to OJ wiki (campus network only). If this cannot help you, feel free to contact us.

C.2 BlackBoard

You are required to upload your **source codes** to the BlackBoard platform. You need to name your files according to the following rules and compress them into `A3_<Student ID>.zip` :

```
A1_<Student ID>.zip
|-- A1_P1_<Student ID>.java/py/c/cpp
|-- A1_P2_<Student ID>.java/py/c/cpp
```

For Java users, **you don't need to consider the consistency of class name and file name**.

For example, suppose your ID is 123456789, and your problem 1 is written in `Python`, problem 2 is written in `Java` then the following contents should be included in your submitted `A1_123456789.zip`:

```
A1_123456789.zip
|-- A1_P1_123456789.py
|-- A1_P2_123456789.java
```

C.3 Late Submissions

Submissions after Feb.16 2025 23:59:00(UTC+8) would be considered as LATE.

The LATE submission page will open after deadline on OJ.

Submisson time = max{latest submisson time for every problem, BlackBoard submisson time}

There will be penalties for late submission:

- 0–24 hours after deadline: final score = your score×0.8
- 24–72 hours after deadline: final score = your score×0.5
- 72+ hours after deadline: final score = your score×0

FAQs

Q: My program passes samples on my computer, but not get AC on OJ.

A: Refer to OJ Wiki Q&A

Authors

If you have questions for the problems below, please contact:

- Problem 1. Yihan Wang: 123090588@link.cuhk.edu.cn
- Problem 2. Wangmeiyu Zhang: wangmeiyuzhang@link.cuhk.edu.cn

CSC3100 Data Structures Spring 2025

Programming Assignment 1

Yihan Wang: 123090588@link.cuhk.edu.cn
Wangmeiyu Zhang: wangmeiyuzhang@link.cuhk.edu.cn

Due: Feb.16 2025 23:59:00

Assignment Link: `https://oj.cuhk.edu.cn/d/csc3100_2025_spring/homework/678c89763f956b4eb8ff899d`

1 Bio-Sequence Alignment (50% of this assignment)

1.1 Description

Scientists have discovered a new organism whose genetic sequence can be represented by an array of integers. Each integer represents a specific base pair. To study the characteristics of this organism, scientists need to compare DNA fragments extracted from samples with the organism's genetic sequence.

A DNA fragment matches the genetic sequence if and only if:

1.The length of the DNA fragment is equal to the length of the genetic sequence.
2.The same base pairs correspond to the same integers, and different base pairs correspond to different integers.

For example, if the genetic sequence is [1, 2, 1, 3], then the DNA fragment "ACAT" matches, while "AACT" does not match, because "C" and "T" correspond to different integers, but in "AACT" they correspond to the same integer 1.

1.2 Input

The first line contains an integer t ($1 \leq t \leq 10^4$), representing the number of test cases.

The first line of each test case contains an integer n ($1 \leq n \leq 2 \cdot 10^5$), representing the length of the genetic sequence.

The second line of each test case contains n integers a_i ($-10^9 \leq a_i \leq 10^9$), representing the genetic sequence.

The third line of each test case contains an integer m ($1 \leq m \leq 2 \cdot 10^5$), representing the number of DNA fragments.

The following m lines each contain a DNA fragment s_j ($1 \leq |s_j| \leq 2 \cdot 10^5$), consisting of uppercase letters 'A' to 'Z'.

It is guaranteed that the sum of n across all test cases does not exceed $2 \cdot 10^5$, and the sum of the lengths of all DNA fragments does not exceed $2 \cdot 10^5$.

1.3 Output

For each test case, output m lines. On the i-th line output ”YES” if the i-th DNA fragment matches the genetic sequence, and ”NO” otherwise. (You may output any case combination of ”YES” or ”NO”)

Sample Input 1

```
3
5
3 5 2 1 3
2
ABFDA
AFBFA
2
1 2
3
AB
ABC
AA
4
5 -3 5 -3
4
AAAA
BCBC
ABA
CBCB
```

Sample Output 1

```
YES
NO
YES
NO
NO
NO
YES
NO
YES
```

Problem Scale & Subtasks

This problem has 10 test cases, each with the same weights.

Test Case No.	Constraints
1-4	$1 \leq n, m \leq 100$
5-7	$1 \leq n, m \leq 10^4$
8-10	$1 \leq n, m \leq 2 \cdot 10^5$

2 K-Base (50% of this assignment)

2.1 Description

Alice developed a game called *K-Base*, but she's not sure whether the difficulty of *K-Base* is appropriate. Now she comes to seek your help. The description of the game is as follows.

- Once *K-Base* is opened, you will receive a number k.
- There are T levels in *K-Base*, and **you need to provide the correct answers to these levels in order**. The answers for each level must be separated by a new line. Note that Alice requires that **all** the T answers you provide be correct.

- In level i, you will receive a row of n_i blocks, each with a k-base positive integer written on it. From left to right, these n_i integers are connected in order, forming a new k-base integer. You need to sort these blocks so that the newly connected integer is the **largest**. Due to Alice's special requirements, you must place the a_ith block of the input that given at the beginning of this level on the far left, which is the position closest to the **highest** digit of the newly connected integer. The answer to this level is the newly connected integer, and you don't need to tell Alice the position of each block.

2.2 Input

The first line contains two integers: $k(2 \leq k \leq 10)$ and $T(1 \leq T \leq 5)$, representing the base of integers on blocks and the number of game levels, respectively. In the next T lines, each line contains the following:

- n_i $(1 \leq n_i \leq 10^5)$, representing the number of blocks in this level;
- n_i integers($1 \leq$the number of digit of each integer≤ 8), representing the integers on blocks, given in order from left to right;
- a_i $(1 \leq a_i \leq n_i)$, representing Alice's special requirements.

2.3 Output

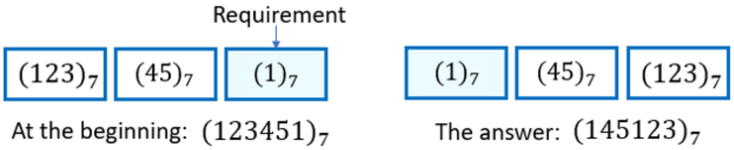
Output T rows, each row consists of one k-base integer, representing the newly connected integer of that level.

Sample Input 1

```
7 1
3 123 45 1 3
```

Sample Output 1

```
145123
```



Sample Input 2

```
10 3
4 1011 1001 1000 1111 1
4 1 11 10 110 2
5 1 12 123 1234 12345 1
```

Sample Output 2

```
1011111110011000
11111010
112345123412312
```

Sample Input 3

```
See attached q2sample.in
```

Sample Output 3

```
See attached q2sample.out
```

2.4 Problem Scale & Subtasks

Hint: the output may be very large.

Constraint A: the lengths of each integer in the same level are same.

Test Case No.	Constraint
1-2	$T = 1,\ k = 10,\ 1 \le n_i \le 10^3$, Constraint A
3-4	$T = 1,\ k = 2$
5-6	$1 \le T \le 5,\ 2 \le k \le 10,\ 1 \le n_i \le 10^5$, Constraint A
7-8	$1 \le T \le 5,\ 2 \le k \le 10,\ 1 \le n_i \le 10^3$
9-10	$1 \le T \le 5,\ 2 \le k \le 10,\ 1 \le n_i \le 10^5$