# Task Overview Sheet

| | Encryption | Grades | Basketball | Happy Numbers |
|---|---|---|---|---|
| **Type** | Batch (stdin/stdout) | Batch (stdin/stdout) | Batch (stdin/stdout) | Batch (stdin/stdout) |
| **Time Limit (per test case)** | 0.5 seconds | 0.5 seconds | 0.5 seconds | 0.5 seconds |
| **Memory Limit (per test case)** | 16 MB | 16 MB | 16 MB | 16 MB |
| **Points** | 100 | 100 | 100 | 100 |

# Encryption

Hareedi and Hanadi are currently working with a security agency on a mission. They exchange messages between them and they do not want anyone else to be able to understand these messages. The language they use consists of integers between **0** and **N**. In order to encrypt their messages, Hareedi and Hanadi decided to replace every integer in their messages with its inverse. The inverse of an integer **D** is **(N-D)**.

## TASK

Write a program that given **N** and an unencrypted message consisting of **M** integers, outputs the message in its encrypted form.

## CONSTRAINTS

| | |
|---|---|
| 0 <= **N** <= 1,000,000,000 | The upper limit of the numbers in Hareedi and Hanadi's special language. |
| 1 <= **M** <= 100,000 | The number of integers in the message. |
| 0 <= $D_i$ <= N | The value of the **i**$^{th}$ integer in the message. |

## INPUT

- Line 1 contains an integer **N**.
- Line 2 contains an integer **M**.
- Line 3 contains the unencrypted message represented as **M** integers separated by spaces.

## OUTPUT

- A single line containing the encrypted message represented as **M** integers separated by spaces.

## EXAMPLE

| Sample Input | Sample Output |
|---|---|
| 7<br>5<br>4 0 2 7 6 | 3 7 5 0 1 |

The unencrypted message consists of 5 integers with the upper limit **N** equal to 7.
The encrypted message is:
7-4 7-0 7-2 7-7 7-6 = 3 7 5 0 1

# Grades

Hanadi is currently teaching **M** courses at a local school. She teaches **N** students the same **M** courses. She wants to find the lowest and the highest grade in each course among her **N** students.

## TASK

Write a program that given the grades of **N** students in **M** courses, finds the highest and the lowest grade in each course.

## CONSTRAINTS

1 <= **N** <= 1,000
1 <= **M** <= 100
0 <= **G** <=1,000,000,000 Grades values.

## INPUT

- Line 1 contains 2 integers separated by a single space **N** and **M**.
- Each of the next **N** lines contains **M** integers, representing the grades of the $i^{th}$ student.

## OUTPUT

- **M** lines, the $j^{th}$ line contains 2 integers the lowest then the highest grade in the $j^{th}$ course.

## EXAMPLE

| Sample Input | Sample Output |
|---|---|
| 5 4<br>70 80 60 90<br>55 70 80 80<br>90 86 77 90<br>60 90 66 50<br>98 95 90 92 | 55 98<br>70 95<br>60 90<br>50 92 |

There are 5 students and 4 courses. The $1^{st}$ student's grades are 70, 80, 60 and 90 in order, etc. The minimum grade in the $1^{st}$ course is 55 and the maximum is 98, the minimum grade in the $2^{nd}$ course is 70 and the maximum is 95, etc.

# Basketball

The mayor of Hareedi's village decided to organize a basketball tournament between the schools in the village. In order to choose the best team to represent his school in the tournament, Hareedi asks all the **N** students in his school to stand in a circle. Then, he decides that his team must consist of **Q** consecutive (standing next to each other) students from the circle. Finally, he chooses the tallest **Q** consecutive students in the circle such that the sum of their heights is as large as possible.

## TASK

Write a program that given the lengths of **N** students standing in the circle, chooses **Q** consecutive students, such that this choice results in a team having the largest sum of heights possible.

## CONSTRAINTS

| | |
|---|---|
| 1 <= **N** <= 5,000 | The total number of students. |
| 1 <= **Q** <= **N** | The required number of team members. |
| 1 <= $L_i$ <= 10,000 | The height of the $i^{th}$ student. |

## INPUT

- Line 1 contains an integer **N**.
- Line 2 contains an integer **Q.**
- Each of the next **N** lines contains an integer $L_i$, the height of the $i^{th}$ student in the circle.

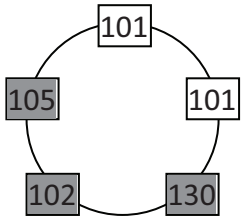
## OUTPUT

- A single line containing a single integer which is the sum of the heights of the best team.

## EXAMPLE

| Sample Input | Sample Output |
|---|---|
| 5<br>3<br>105<br>101<br>101<br>130<br>102 | 337 |

There are 5 students; you should choose the three tallest students that are standing consecutively. Since this is a circle then the student having height 105 is standing next to the student having height 102 and vice versa. So the sum of the lengths of the team members of the best team will be 130 + 102 + 105 = 337.



# Happy Numbers

The process of "**breaking"** an integer is defined as summing the squares of its digits. For example, the result of breaking the integer 125 is $(1^2 + 2^2 + 5^2) = 30$. An integer **N** is **happy** if after **"breaking"** it **repeatedly** the result reaches 1. If the result never reaches 1 no matter how many times the "breaking" is repeated, then **N** is **not a happy number**.

## TASK

Write a program that given an integer **N**, determines whether it is a happy number or not.

## CONSTRAINTS

2 <= **N <=** 2,147,483,647

## INPUT

- A single line containing a single integer **N**.

## OUTPUT

- A single line containing a single integer **T** which is the number of times the process had to be done to determine that **N** is happy, or -1 if **N** is not happy.

## GRADING

- For some test cases, worth **50 points N** will be a happy number.

## EXAMPLE A

| Sample Input | Sample Output |
|---|---|
| 19 | 4 |

1) 19 : $1^2 + 9^2 = 82$
2) 82 : $8^2 + 2^2 = 68$
3) 68 : $6^2 + 8^2 = 100$
4) 100 : $1^2 + 0^2 + 0^2 = 1$

The solution is 4 because we discovered that the integer 19 is happy after we repeated the process 4 times.

## EXAMPLE B

| Sample Input | Sample Output |
|---|---|
| 204 | -1 |

204 –> **20** –> 4 –> 16 –> 37 –> 58 –> 89 –> 145 –> 42 –> **20** –> 4 –> 16 –> 37 –> 58 –> 89 –> 145 ……..

204 is not a happy number because after breaking it several times the results start repeating so we can deduce that if we continue breaking it, the result will never reach 1.