

# HW3

---

## Rules & Instruction

---

- Compiler of programming problem.
  - C : `gcc -DONLINE_JUDGE -O2 -w -fmax-errors=3 -std=c11 main.c -lm -o main`
  - C++ : `g++ -DONLINE_JUDGE -O2 -w -fmax-errors=3 -std=c++17 main.cpp -lm -o main`
  - Execution: `./main`
- Can I use theorem that haven't been mentioned in class ?
  - Programming: No restriction.(無限制)
  - Handwritten: Please include its proof.(請寫上證明。)
- Rules of cheating.
  - Programming: We use tools to monitor code similarity.(我們用機器抓抄襲)
  - Handwritten: Though we forbid plagiarism, we still encourage you to discuss with each other. Remember that after discussion you must answer problems **in your own words**. (禁止抄襲，鼓勵大家有問題可以互相討論，**但請用自己的話去作答。**)
  - **Copycats will be scored 0.** You also need to have a cup of coffee with professor Yeh. (抄襲者與被抄襲者當次作業零分，另外會跟教授喝杯咖啡☕。)
  - People who committed twice or more will be punished. (情節嚴重者，如累犯...，以校規處置)
- Rules of delay.  
**NO LATE SUBMISSION IS ALLOWED.**  
That is, zero toleration of late submission.

# Non-Programming

Credits:

Icons of Bento made by Fuji.

Icons of Watermelon made by [Freepik](https://www.freepik.com) from [www.flaticon.com](https://www.flaticon.com)

## 1. Warm-up

### Knapsack problem

item	weight	value
1	2	4
2	3	2
3	1	2
4	4	3

The capacity of the knapsack is 6, which means that the sum of the weights of items taken must be at most 6.

Find the maximum possible sum of the values of items that fit into the knapsack.

All kinds of the item are having exactly one.

(1). (6 points)

Write down the state transition(recursive relation), and tabulate(draw a table) with the given items by using the transition you wrote.

Please label the meaning of the column and the row in your table.

### Game of Stone

There are  $N$  piles of stones  $P_1, P_2, \dots, P_N$ , placed in a row, where  $N$  is even,  $P_i$  is the number of stones in  $i$  -  $th$  pile. You play a game against an opponent by alternating turns.

In each turn, a player has to select either the first pile or the last pile from the residual piles, take them away from the row, and obtains the number of the stones in that pile.

Determine the maximum possible amount of stones you can definitely obtain if you move first.

Absolutely, both of the opponents and you will take the best strategy to win the game.

$dp(i, j)$  = the maximum possible amount of stones the player who moved first can get in the interval  $P_i, P_{i+1}, \dots, P_j$ .

(1). (10 points)

Assuming you move first, taking a pile from  $P_i, P_{i+1}, \dots, P_j$ , then your maximum earnings is  $dp(i, j)$ .

Now in this round, you want to take  $P_i$  (the first pile), fill in the blank "(A)" in the following equation.

$$dp(i, j) = P_i + \text{---(A)---}$$

On the contrary, if you want to take  $P_j$  (the last pile), what is the "(B)" blank should be.

$$dp(i, j) = \text{---(B)---} + P_j$$

Briefly explain your answer.

(2). (15 points)

Follow the previous problem, write down the state transition, and tabulate with the following instance.

$P_1, \dots, P_6 = 2, 8, 3, 7, 5, 3$

Build a table, and fill it with  $dp(i, j)$  in cell  $(i, j)$ .

*Hint* : the approach is similar to the Matrix-chain Multiplication problem.

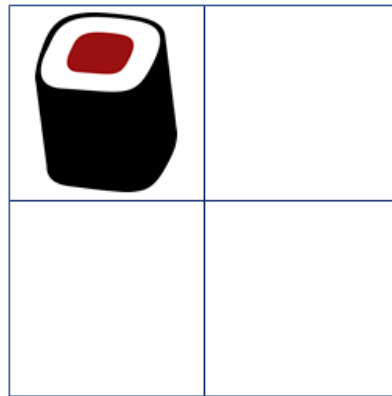
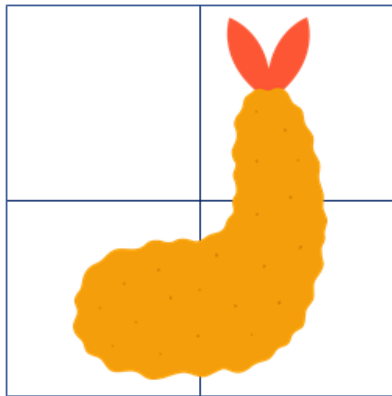
## 2. Bento

It's time for lunch!

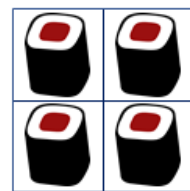
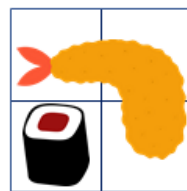
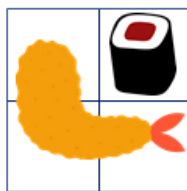
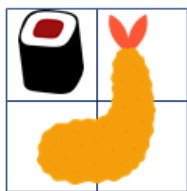
Bogay got a bento box, its width is 2 and length is  $N$ , where  $N$  is a positive number.

He wants to place only Tempura and Tekkamaki in his bento box.

- Tempura: need an L-shaped space, which costs 3 cells.
- Tekkamaki: need a square space, which costs 1 cell.



Bogay wondered how many different ways he can place Tempuras and Tekkamakis into his bento box for every number  $N$ . For example, the following figures show the 5 ways to place bento of  $N = 2$ , that is, the size of the bento box is  $2 \times 2$ .



Let  $dp(N)$  be the ways to place into box that is length  $N$ , so  $dp(2) = 5$ .

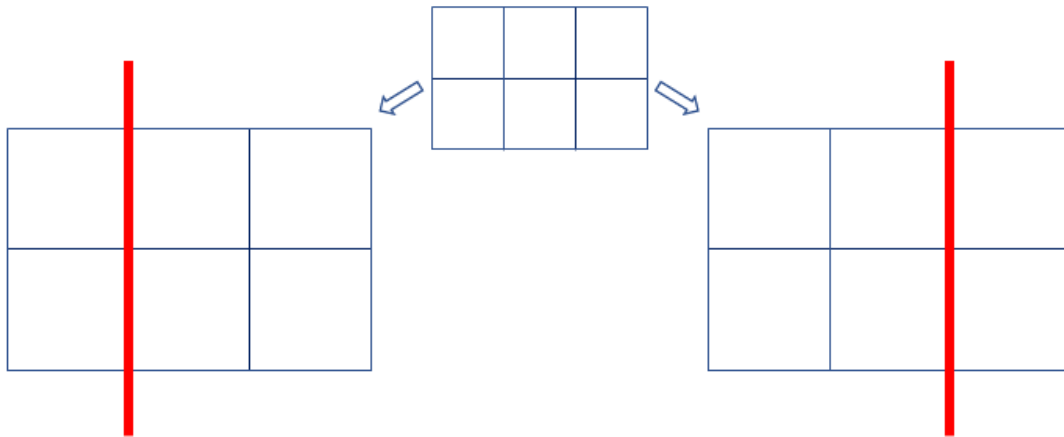
Answer the following questions.

(1). (10 points)

Find  $dp(1)$  and  $dp(3)$ , and draw down all the possible cases of  $dp(1)$  and  $dp(3)$  just like the way we show  $dp(2)$  in above figure.

(2). (10 points)

When box's length is 3 ( $N = 3$ ), we can split the box into two part, left part is a box with length 1 and right part with length 2, and vise versa(left part is length 2 and right is length 1).



Since we can split the box into above two cases, does the equation

$dp(3) = (dp(1) + dp(2)) + (dp(2) + dp(1))$  hold?

For both answers (yes/no) you have to explain why the equation does/doesn't hold.

Your explanation has to describe the meaning inside, not only "Because  $x \neq y+z+z+y$ " for example, or your answer won't be given any credits.

(3). (5 points)

Find  $dp(4)$  by using  $dp(1)$ ,  $dp(2)$ ,  $dp(3)$ .

(4). (20 points)







Formulate the state relationship of  $dp(i)$ , note that you have to explain your approaches.

### 3. Watermelon

After lunch, Bogay wants to have some fruit. Unfortunately, the 2020 NTNU watermelon festival has been canceled, so he can't participate in the all-you-can-eat contest.

Thus, he decides to plant watermelons by himself.

He has farmland composed of  $N \times M$  squares, the following is an example  $G$  of a  $2 \times 3$  farmland.



















1 	1 	0 
0 	1 	0 

Some of the squares are infertile and can't be planted, which are represented by 0, and 1 means it can be planted.

Moreover, Bogay wants to **avoid** planting watermelons on adjacent squares, because he believes in this way the watermelons will grow bigger.

To satisfy all the conditions, what is the number of ways he can choose the squares to plant?

The answer of  $G$  is 4.

Case 1			Case 2			Case 3			Case 4		
	1	0	1		0	1	1	0		1	0
0		0	0	1	0	0		0	0		0
											

In this problem, we can use a special technique called *bitmask*. Since each square is either 0 or 1, a row of farmland can be encoded by a sequence of  $M$  bits (we will call this sequence "a mask"), for example, the first row of  $G$  is 110, which is 6 in decimal, and the second row is 010, which is 2.

Thus, we can only use an integer to store the state of a row of farmland, furthermore, we can store a farmland  $F$  as an Array.

$F[i]$  = the state of  $i$  - th row of  $F$

So, for the above example  $G$  we have  $G[0] = 6, G[1] = 2$ .

Answer the following questions.

(1). (5 points)


Assuming there are  $M$  squares in a row of farmland, how many different states may need for representing this row?

(2). (5 points)

Follow the previous problem, all the states are represented as decimal integers, and what is the range of these integers?

Now we define another *bitmask*, similarly, we use a mask to store planting situations of a row, for each square, 1 indicates Bogay want to plant a watermelon on it, 0 otherwise.

One of the above example, case1, can be written as:

1 	0	0
0	0	0

because only the left top square is planted a watermelon.

The problem: "Find the number of ways Bogay can choose the squares to plant." has a dynamic programming solution.

Let  $dp(i, S)$  be the answer where the planting situation of  $i$  - th row is  $S$ .

$dp(i, S) = \sum \{dp(i - 1, P) \mid P \text{ satisfies the conditions}\}$

**conditions**

- can't plant watermelons on infertile square
- can't plant watermelons on adjacent squares

We can check a mask whether it is satisfying the conditions through some bit operations.

For example, to check if some planted squares are adjacent in the same row, we can check  $S \& (S \ll 1)$  is 0 or 1, if 0 then means no planted squares are adjacent.

Answer the following questions.

(3). (7 points)

Find a method to check if some planted squares are adjacent between  $(i - 1) - th$  row and  $i - th$  row, the planting situation in  $(i - 1) - th$  row is  $P$  and the planting situation in  $i - th$  row is  $S$ .

(4). (7 points)

Find a method to check if there is any watermelon planted on an infertile square, the planting situation in  $i - th$  row is  $S$ , and the state in  $i - th$  row of farmland is  $F[i]$ .

# Programming

## pA: Sandbox Late-Night Supper

Time Limit: 1s

Memory Limit: 409600KB

### Description

As members of Sandbox Team, Bogay and **aserf18766**(two people) usually stay up all night to design a secure sandbox. Obviously, they need some late-night supper to revive themselves. However, both of them don't want to run some errands. So their PM, Da-Hei-Hei, design a game to help them decide who should run errands.

First of all, there are some numbers on the table. Bogay and **aserf18766** take  $K$  numbers from leftmost or rightmost by turns(Bogay first). After some rounds, all those numbers will be taken by Bogay or **aserf18766**. Then, Bogay sum up all numbers( $B$ ) he takes away, **aserf18766** sum up his also( $A$ ). The person who gets more sum will win.

As their fair PM, Da-Hei-Hei will announce who wins the game and needn't go to GUO-JI(a street vendor where they can buy potsticker and fried chicken cutlet) after the game.

**Absolutely, both of them will take the best strategy to win the game.**

### Input Format

First line contains two integer  $N$ ,  $K$ , and second line contain  $N$  integer numbers( $a_1, a_2, \dots, a_n$ ).

For all test data, it is guaranteed:

- $1 \leq N \leq 10^6$
- $1 \leq K \leq N$
- $N \bmod 2K = 0$
- $1 \leq \frac{N}{K} \leq 10^4$
- $-10^3 \leq a_i \leq 10^3, \forall i \in [1, N]$

#### Subtask1 (10%)

- $1 \leq N \leq 25$
- $K = 1$

#### Subtask2 (15%)

- $1 \leq N \leq 25$

#### Subtask3 (15%)

- $K = 1$

#### Subtask4 (60%)

- No other restrictions.

### Output Format

If the game ends in a tie, please output "TIE" in the first line, or please output who wins the game otherwise.

Output  $B$  and  $A$  separated by white space in the second line.

### Sample Input 1

```
2 1
-5 -5
```

### Sample Output 1

```
TIE
-5 -5
```

### Sample Input 2

```
8 1
-7 7 0 -4 10 -8 3 -8
```

### Sample Output 2

```
Bogay
6 -13
```

### Sample Input 3

```
24 6
8 -6 4 10 10 7 -9 -2 -1 4 -5 -1 8 1 9 -8 2 7 6 -7 7 2 -3 10
```

### Sample Output 3

```
Bogay
52 1
```

### Hint

Actually, Bogay seldom loses games, so **aserf18766** usually go to GUO-JI for supper.



## pB: LHS

Time Limit: 1s

Memory Limit: 134218KB

### Description

So intelligent as A-Hua is, he's good at solving subsequence problems, such as LCS(Longest Common Subsequence), LIS(Longest Increasing Subsequence) or even LCIS(Longest Common Increasing Subsequence) problem.

Now, he brought a new subsequence problem to you: **Liông-Hó Subsequence** problem, as known as LHS.

To say a sequence is **liông-hó** if it contains at least 2 elements and it contains 2 neighbors with the difference no larger than  $M$ , where  $M$  is a positive integer.

Given a sequence with  $N$  elements, calculate the number of LHS in it.

Note that in HINT section contains an explanation of sample input.  
Bonus of HW3 is the hard edition of this problem.

### Input Format

First line contains two space-separated integers  $N$ ,  $M$ .

Second line contains  $N$  space-separated integers  $a_1, a_2, \dots, a_N$ , indicating the elements in the sequence.

For all test data, it is guaranteed:

- $2 \leq N \leq 10^4$
- $1 \leq M \leq 10^3$
- $1 \leq a_i \leq 10^5$

#### Subtask1 (25%)

- $2 \leq N \leq 20$

#### Subtask2 (75%)

- No other restrictions.

### Output Format

Output the number of LHS of the given sequence. Since the number might be too large, please let the answer modulo  $1000000007$  ( $10^9 + 7$ ).

### Sample Input

```
4 2
5 3 8 6
```

### Sample Output

```
8
```

### Sample Input

```
3 2
5 3 3
```

## Sample Output

```
4
```

## Hint

The LHS of the first sample input are:

```
1: values {5, 3}
2: values {5, 6}
3: values {5, 3, 8}
4: values {5, 3, 6}
5: values {5, 8, 6}
6: values {5, 3, 8, 6}
7: values {3, 8, 6}
8: values {8, 6}
```

The LHS of the second sample input are:

```
1: values {5, 3}
2: values {5, 3}
3: values {3, 3}
4: values {5, 3, 3}
```

## Bonus: LHS (Hard Edition)

Time Limit: 1s

Memory Limit: 134218KB

### Description

So intelligent as A-Hua is, he's good at solving subsequence problems, such as LCS(Longest Common Subsequence), LIS(Longest Increasing Subsequence) or even LCIS(Longest Common Increasing Subsequence) problem.

Now, he brought a new subsequence problem to you: **Liông-Hó Subsequence** problem, as known as LHS.

To say a sequence is **liông-hó** if it contains at least 2 elements and it contains 2 neighbors with a difference no larger than  $M$ , where  $M$  is a positive integer.

Given a sequence with  $N$  elements, calculate the number of LHS in it.

Note that in HINT section contains the explanation of sample input.

Bonus of HW3 is the hard edition of this problem.

### Input Format

First line contains two space-separated integers  $N$ ,  $M$ .

Second line contains  $N$  space-separated integers  $a_1, a_2, \dots, a_N$ , indicating the elements in the sequence.

For all test data, it is guaranteed:

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq M \leq 10^3$
- $1 \leq a_i \leq 10^5$

### Subtask1 (100%)

- No other restrictions.

### Output Format

Output the number of LHS of the given sequence. Since the number might be too large, please let the answer modulo 1000000007 ( $10^9 + 7$ ).

### Sample Input

```
4 2
5 3 8 6
```

### Sample Output

```
8
```

### Sample Input

```
3 2
5 3 3
```

## Sample Output

4

## Hint

The LHS of the first sample input are:

```
1: values {5, 3}
2: values {5, 6}
3: values {5, 3, 8}
4: values {5, 3, 6}
5: values {5, 8, 6}
6: values {5, 3, 8, 6}
7: values {3, 8, 6}
8: values {8, 6}
```

The LHS of the second sample input are:

```
1: values {5, 3}
2: values {5, 3}
3: values {3, 3}
4: values {5, 3, 3}
```