HW1

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 or the source.(請寫上證明或來源。)
- 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

1. Complexity

For each function of the following C program, find out the tightest bound using Θ notation.
Explanation or proof is needed.
Answers without any explanation or proof will **NOT** be given any credits.

(1). $(10\ points)$

```c
void f(int n) {
    int cnt = 0;
    for ( int i=1; i<=n; i++ ) {
        for ( int j=i+1; j<=n; j++ ) {
            cnt = cnt + 1;
        }
    }
}
```

(2). $(10\ points)$

```
void h(int n, int m) {
    int sum = 0;
    for ( int i=0; i<n; i++ ) {
        sum = sum + i;
    }
    for ( int i=0; i<m; i++ ) {
        sum = sum + i;
    }
}
```

(3). (10 *points*)

```
void g(int n) {
    if ( n <= 0 ) {
        return;
    }
    g(n-2);
    g(n-1);
}
```

(4). (10 *points*)

```
void j(int n) {
    if ( n == 1 ) {
        return;
    }
    j(n/2);
    j(n/2);
    int total = 0;
    for ( int i=0; i<n; i++ ) {
        total = total + i;
    }
}
```

(5). (10 *points*)

```
void f(int n) {
    int cnt = 0;
    for ( int i=1; i<=n; i++ ) {
        for ( int j=i; j<=n; j+=i ) {
            cnt = cnt + 1;
        }
    }
}
```

Hint: `Squeeze theorem`

2. Fibonacci

The Fibonacci numbers form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1.

That is,
$F_0 = 0,\ F_1 = 1,$
and
$F_N = F_{N-1} + F_{N-2},\ for\ N > 1$

The beginning of the sequence is thus:
$0,\ 1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34,\ 55,\ 89,\ 144,\ \ldots$

and it can be described in Matrix form:

$$\begin{pmatrix} F_{k+2} \\ F_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} F_{k+1} \\ F_k \end{pmatrix}$$

Thus, we can figure out F_N with $F_0,\ F_1$ by:

$$\begin{pmatrix} F_N \\ F_{N-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{N-1} \begin{pmatrix} F_1 \\ F_0 \end{pmatrix}$$

(1). $(5\ points)$
To multiple two n-dimension matrix, we can implement by a Naïve algorithm.

```
for ( int i=0; i<n; i++ ) {
    for ( int j=0; j<n; j++ ) {
        c[i][j] = 0;
        for ( int k=0; k<n; k++ ) {
            c[i][j] = c[i][j] + a[i][k]*b[k][j];
        }
    }
}
```

What is the complexity of this algorithm? Using Θ notation with n.

(2). $(20\ points)$

Let $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$

To obtain A^{N-1} is quite simple:

```
Power(A):
    ans = 1;
    for ( int i=0; i<N-1; i++ ) {
        ans *= A; // Matrix Multiplication
    }
    return ans;
```

The complexity is $O(N)$, we want to make it better, and we've learned that Divide & Conquer can calculate x^n more efficient.

Please **provide a Divide & Conquer algorithm or method**, which can obtain A^{N-1} in $o(N)$, and analyze the complexity of your solution by **Master Theorem**.
You can provide C code, Pseudo code or just explain the method in plain text. However, make sure your method is clearly described.

3. Stack

Here we defined 3 operations than can apply to stack.

1. $Push(x)$: push an element x into stack, $O(1)$.
2. $Pop()$: pop out the top element from stack, $O(1)$.
3. $Grab(k)$: pop out the top k elements from stack, if the number of elements is less than k, just pop out all elements, here is the pseudo code.

```
Grab(k):
    while ( !stack.empty() || k > 0 ) {
        Pop();
        k = k-1;
    }
```

now answer the following questions

(1). $(10\ points)$
Let n be the number of elements in stack, what is the running time of operation: $Grab(k)$? Using O notation with k or n (or both).

(2). $(15\ points)$
Define a function: apply n random operations consecutively to a stack which is empty initially, what is the overall running time of this function?
You should find out the bound using Θ notation, and clearly describe your reason or proof.

Congratulations! You've done the handwritten part, remember to submit your PDF file on online judge.
Next part are programming problems, good luck and have fun!

Programming

pA: A-Hua's Software Engineering

> Time Limit: 1s
> Memory Limit: 536871KB

Description

Although A-Hua took software engineering last semester, he could not concentrate on Linux Chi's lecture. During the class, A-Hua usually studied tâi-gí in software engineering class, so Bogay decided to set a math problem for A-Hua to prevent him from his being addicted to tâi-gí.

The following is the description of this math problem. There is a sequence starts with $0, 1$, such that each number is the sum of the two preceding ones.

A-Hua said that it's as easy as pie. It's just the common Fibonacci sequence. So clever as A-Hua is, he modified the problem to make it more difficult and return it to Bogay.

The new sequence starts with a_1 and a_2, the following number in this new sequence follow the rule. $a_n = xa_{n-1} + ya_{n-2}$. Given a_1, a_2, x, y, please help Bogay to find a_n.

Input Format

There is only one line containing a_1, a_2, x, y, n.

For all test data, it is guaranteed:

- $0 \leq a_1, a_2, x, y \leq 10^9$
- $3 \leq n \leq 2^{31} - 1$

Subtask1 (10%)

- $3 \leq n \leq 5$

Subtask2 (20%)

- $3 \leq n \leq 10^6$

Subtask3 (70%)

- No other restrictions.

Output Format

Because a_n may be so large, please let answer mod 1000000007 before outputting it.

Sample Input 1

```
0 1 1 1 3
```

Sample Output 1

```
1
```

Sample Input 2

```
970279151 873914621 606321275 988849410 731274
```

Sample Output 2

```
570653198
```

Sample Input 3

```
10013856 814986795 320398367 26310064 1041062793
```

Sample Output 3

```
176925719
```

Hint

Handwritten 2. (2)

10^8 iterations will take around 1 second on platforms.
If you get **TLE**(Time Limit Exceed), it may mean that the complexity of your solution is not good enough to pass this problem.

pB: Teleport

> Time Limit: 1s
> Memory Limit: 536871KB

Description

Bogay, one of the greatest wizard in NTNU, bought a Mastery Book with all his money, because he want to sharpen the skill: $Teleport$.

This Mastery Book contain N pages, each page has exactly one magic number on it.
Let's call the magic number on i-th page is M_i.

At first, Bogay can choose any page(Let's say he chose i-th page), and he can launch $Teleport$: moving forward for exactly M_i distance.

However, if Bogay want to launch $Teleport$ again, he is forced to choose the magic number on $(i-1)$-th page, that is, moving forward for M_{i-1} distance, and the next time, $(i-2)$-th page, and so on.

Bogay can launch any number of time, and can choose any page at first.
He wonders whether or not he can move forward for given D distance exactly after launching some $Teleport$s.

Input Format

First line contains two space-separated integers $N,\ D$, indicating the number of pages of Mastery Book, and the desired distance.
Second line contains N space-separated integers $M_1,\ M_2,\ \ldots,\ M_N$, indicating the magic numbers on each page.

For all test data, it is guaranteed:

- $1 \leq N \leq 2 \times 10^5$
- $1 \leq D \leq 2 \times 10^{10}$
- $1 \leq M_i \leq 10^5$

Subtask1 (25%)

- $1 \leq N \leq 10^3$
- $1 \leq D \leq 10^6$
- $1 \leq M_i \leq 10^3$

Subtask2 (75%)

- No other restrictions.

Output Format

Print "yes" (without quote), if Bogay can move forward exactly D distance after launching some $Teleport$s, and "no" otherwise.

Sample Input1

```
6 11
4 6 3 1 3 7
```

Sample Output1

```
yes
```

In this example, Bogay can choose 6th page (move 7), and decide to move again (move 3), and again (move 1), so that the sum of distance is exactly 11 .

Sample Input2

```
6 8
4 6 3 1 3 7
```

Sample Output2

```
no
```

In this example, no matter which page Bogay chooses at first, and what number of $Teleport$s he wants to use again, he can not achieve the sum 8 eventually.

Hint

Because the input files are large, please add

```
std::ios_base::sync_with_stdio(false);
std::cin.tie(nullptr);
```

to the beginning of the main function if you are using std::cin.

Bonus: Group Up!

> Time Limit: 1s
> Memory Limit: 536871KB

Description

At the end of Software Engineering class, Linux Chi asked everyone to hand in a report, including a very hard question: "Find a good way to split people into groups."

Assuming N students are lining up, Linux Chi wants to **slice them into at least M groups**, that is, only adjacent students can form a group.

A-Hua thought: "To form a group well, the difference of ability between groups should be small!", so he decided to let the weakest group as strong as possible among all grouping combination.

Formally, each student has a AP(ability point), and the AP of a group is the sum of all group members' AP. A-Hua wants to know the maximum possible AP of the group whose AP is minimum of all groups.

Please see the example and hint down below.

Input Format

First line: 2 space separated integers N and M, indicating the number of students and the number of groups that at least to be made.

Second line: N space separated integers, $a_1, a_2, \ldots, a_n$, indicating each student's AP.

For all test data, it is guaranteed:

- $1 \leq N \leq 10^5$
- $1 \leq M \leq N$
- $1 \leq p_i \leq 10^4, for\ 1 \leq i \leq N$

Subtask1 (100%)

- No other restrictions.

Output Format

An integer, indicating the maximum possible AP of the group whose AP is minimum of all groups.

Sample Input

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

Sample Output

```
7
```

Hint

Sample Explanation:
The following is a part of grouping results to slice a queue into at least two groups.

- $\{1\}, \{5, 2, 4, 3\} \rightarrow 1, 14 \rightarrow 1$
- $\{1, 5\}, \{2, 4, 3\} \rightarrow 6, 9 \rightarrow 6$
- $\{1, 5, 2\}, \{4, 3\} \rightarrow 8, 7 \rightarrow 7$
- $\{1, 5, 2, 4\}, \{3\} \rightarrow 12, 3 \rightarrow 3$
- $\{1\}, \{5, 2, 4\}, \{3\} \rightarrow 1, 11, 3 \rightarrow 1$
- ...etc.
  In all possible grouping combinations, the maximum AP among every minimum groups' AP is 7.