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` :

```
A2_<Student ID>.zip
|-- A2_P1_<Student ID>.java/py/c/cpp
|-- A2_P2_<Student ID>.java/py/c/cpp
|-- A2_P3_<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 and 2 is written in `Python`, problem 3 is written in `Java` then the following contents should be included in your submitted `A2_123456789.zip`:

```
A2_123456789.zip
|-- A2_P1_123456789.py
|-- A2_P3_123456789.py
|-- A2_P3_123456789.java
```

C.3 Late Submissions

Submissions after Mar.17 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:

- Hanjun Zheng: 122090797@link.cuhk.edu.cn
- Diyuan Deng: 123090079@link.cuhk.edu.cn

CSC3100 Data Structures Spring 2025

Programming Assignment 2

Hanjun Zheng: 122090797@link.cuhk.edu.cn
Diyuan Deng: 123090079@link.cuhk.edu.cn

Due: Mar.17 2025 23:59:00

Assignment Link: https://oj.cuhk.edu.cn/d/csc3100_2025_spring/homework/67c3e17024b1bb4be9a265f1

Problem 1: Sparse Matrix Addition (20% of this assignment)

Description

Given two sparse matrices A and B, calculate their sum $C = A + B$.

Sparse Matrix Format

The following format is used to represent sparse matrices in all problems of this assignment, both for input and output.

- The first line contains two integers: n (number of rows) and m (number of columns).
- Next, there will be n lines, each describing one row of the matrix. The lines is formatted as follows:
  - Each line starts with an integer c, denoting the number of non-zero elements in that row.
  - This is followed by c pairs of integers (j, a), where:
    - j is the 0-based column index of the non-zero element.
    - a is the value of that element, which **must not be zero**.
  - The pairs must be listed in **strictly increasing order of** j.
- All integers within a line are separated by a **single space**.
- Matrix representations are separated by a **single blank line**. There should be no extra blank lines before the first matrix.
- All indices **start with 0**.

Example

```
3 4
0
2 0 -1 2 6
1 3 3
```

This describes the following matrix: $\begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 6 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$.

Input Format

Two matrices with identical dimensions, give in the format above.

Output Format

The resulting matrix C in the given format.

Constraints

- $1 \leq n, m \leq 10^5$.
- For 30% of the test cases, $1 \leq n, m \leq 10^3$.
- $\sum c \leq 5 \times 10^5$, i.e. total number of non-zero elements in a matrix $\leq 5 \times 10^5$.
- $|a| \leq 100$, i.e. every elements in a matrix $\in [-100, 100]$.
- A and B have identical dimensions.

Sample Input

```
2 3
0
2 0 3 1 2

2 3
2 1 1 2 -1
1 0 -5
```

Sample Output

```
2 3
2 1 1 2 -1
2 0 -2 1 2
```

Problem 2: Sparse Matrix Multiplication (40% of this assignment)

Description

Given two sparse matrices A and B, compute their product $C = A \times B$.

Sparse Matrix Format

Same as Problem 1 specification.

Input Format

Two matrices given in the specified format. The number of columns of the first matrix equals to the number of rows of the second matrix.

Output Format

Result matrix C in the specified format.

Constraints

- $1 \leq n, m \leq 10^4$.
- For 30% of the test cases, $1 \leq n, m \leq 100$.
- For 40% of the test cases, $1 \leq n \times m \leq 10^4$.
- For the other 60% of the test cases, non-zero elements are distributed "uniformly".

- $\sum c \leq 10^5$, i.e. total number of non-zero elements in a matrix $\leq 10^5$.
- $|a| \leq 100$, i.e. every elements in a matrix $\in [-100, 100]$.
- $m_A = n_B$, i.e. the number of columns of the first matrix equals to the number of rows of the second matrix.

Sample Input

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

3 2
2 0 5 1 6
1 1 7
0
```

Sample Output

```
2 2
2 0 10 1 12
1 1 21
```

Problem 3: Power Iteration (40% of this assignment)

Description

Approximate the dominant eigenvalue (largest absolute value) of a symmetric square matrix.

Sparse Matrix Format

Same as Problem 1 specification.

Input Format

Single **square symmetric** matrix, given in the specified format.

Output Format

A single number representing the **absolute value** of the dominant eigenvalue (**4 decimal places without rounding**).

Hint: The power iteration algorithm converges to correct answers gradually. Its outputs may not be perfectly precise. If your results are not as expected, consider adjusting number of iterations or the random seed to improve accuracy. Experimenting with these parameters can help you achieve a balance between performance and precision.

Constraints

- $1 \leq n, m \leq 10^3$.
- $\sum c \leq 10^5$, i.e. total number of non-zero elements in a matrix $\leq 10^5$.
- Non-zero elements are distributed "uniformly".
- $|a| \leq 100$, i.e. every elements in a matrix $\in [-100, 100]$.
- $A_{i,j} = A_{j,i}$ and $n = m$, i.e. the given matrix is a symmetric square matrix.

Sample Input

```
3 3
1 0 2
1 1 3
1 2 -5
```

Sample Output

```
5.0000
```