Пам'ятка

Загальне:

На тренувальному турі вам дані задачі для ознайомлення з роботою системи. Результати тренувального туру **ніяк** не впливають на ваш результат на олімпіаді.

На кожному з турів учасникам буде запропоновано по 4 задачі. На кожному турі ви можете відправити свої розв'язки не більше 60 разів. Розмір кожного файлу не має перевищувати 64KB.

Оцінювання:

Усі тести поділені на блоки, які описані в умові задачі. В кожному блоці, крім загальних умов, виконуються додаткові умови. Бали нараховуються лише при проходженні **всіх** тестів блоку. Якщо обмеження блока i не менші за обмеження блока j, то для нарахування балів за блок i, також потрібно, щоб пройшли всі тести блока j. В умові про це не буде сказано. Також є «нульовий блок», який складається з прикладів, він оцінюється в 0 балів. В умові про це також згадувати не будуть. Якщо приклади підходять під обмеження блока, то для проходження того блока, потрібно, щоб ваш розв'язок давав правильні відповіді на всі приклади.

В деяких задачах можуть бути часткові бали за блоки. Про таке оцінювання буде написано детальніше в кожній з задач, де таке використовується.

Технічні вимоги:

Деякі задачі будуть стандартними, тобто потрібно зчитувати дані зі стандартного потоку введення файлу (або файлу) та виводити у стандартний потік виведення (або у файл). Ви можете використовувати будь-який з двох методів введення та виведення даних (або стандартні потоки, або файли). Тестувальна система чутлива до регістру у назвах файлів. Назви всіх файлів повинні бути написані в нижньому регістрі, як зазначено в умові задач. Наприклад: `problem.in`, `problem.out` (а не `PROBLEM.IN` чи `Problem.out`).

Зверніть увагу, деякі задачі будуть «з модулями». Тобто вам потрібно реалізувати функції, які описані у задачі. Зверніть увагу, що в цьому виді задач **суворо забороняється** зчитувати та виводити дані, також використовувати змінні та функції, які не були вами оголошені, а також намагатись отримати дані про файли, до яких у вас немає доступу. Ви маєте працювати з даними лише в той спосіб, який описаний в умові задачі. Порушення цих правил може призвести до дискваліфікації.

Вам дадуть архів, в якому будуть файли для кожної мови програмування. Ці файли потрібно компілювати разом.

C++:

Вам дано файли: `problem.cpp`, `problem.h`, `grader.cpp`. Ви маєте дописати файл `problem.cpp` та відправити лише його. Файли потрібно компілювати так:

```
g++ problem.cpp grader.cpp && ./a.out
```

Зверніть увагу, що файл `problem.h` відсутній у команді.

Java:

Вам дано файли: `problem.java` та `grader.cpp`. Ви маєте дописати файл `problem.java` та відправити лише його. Файли потрібно компілювати так:

```
javac "grader.java" "problem.java" && jar cfe "a.jar" "grader" *.class \
    && java -jar "a.jar"
```

Python 3:

Вам дано файли: `problem.py` та `grader.py`. Ви маєте дописати файл `problem.py` та відправити лише його. Файли потрібно компілювати так:

```
python3 grader.py problem.py
```

Pascal:

Вам дано файли: `problem.pas` та `grader.pas`, а також, можливо, `lib.pas`. Ви маєте дописати файл `problem.pas` та відправити лише його. Файли потрібно компілювати так:

```
fpc grader.pas && ./grader
```

Функції:

У кожній задачі «з модулями» дано псевдокод всіх функцій. Кожна функція має вигляд `<тип, який повертається> <ім'я функції>(<параметри>)`, де в параметрах вказані всі параметри через кому в форматі `<тип> <ім'я>`.

Типи бувають такими:

- `integer` — ціле число;
- `string` — рядок;
- `character` — символ;
- `boolean` — логічний тип даних, буває або `true` (істина), або `false` (хиба);
- `array of <тип>'s` — масив, елементи якого мають тип `<тип>`, зверніть увагу, що елементи в масивах нумеруються, починаючи з нуля, а не з одиниці;
- `void` (лише перед іменем функції) означає, що функція нічого не повертає.

Наприклад, функція `solve`, яка має два параметри: ціле число n та масив цілих чисел a, та повертає рядок, матиме вигляд:

```
string solve(integer n, array of integers a)
```

Загальні вимоги:

Учасники будуть відсторонені від участі в олімпіаді за:

1. Реалізацію у програмі дій, які можуть бути кваліфіковані як такі, що навмисно призводять до нестабільної роботи сервера під час перевірки роботи.
2. Використання мережі будь-яким чином, окрім як для роботи з онлайн-системою журі.
3. Взаємодію з даними не в той спосіб, який описаний в умові задачі (в тому числі за зчитування та виведення даних у задачах «з модулями»).
4. Спілкування у будь-який спосіб під час турів (окрім спілкування з представниками оргкомітету).
5. Наявність на робочому місті:
   (a) Електронних носіїв інформації.
   (b) Друкованих та рукописних матеріалів (окрім наданих оргкомітетом).
   (c) Засобів мобільного зв'язку та портативних обчислювальних пристроїв (мобільних телефонів, калькуляторів, ноутбуків і таке інше).

Рекомендації:

Радимо вам протягом туру дотримуватися таких правил:

1. Уважно прочитайте та зрозумійте умови усіх задач та вчасно поставити запитання, якщо виникнуть.
2. Розв'язуйте спочатку ту задачу, яка здається вам простішою.
3. Не забувайте регулярно зберігати проміжні версії програм-розв'язків.
4. Перевіряйте правильність роботи ваших програм-розв'язків на різних наборах вхідних даних, в тому числі й на «крайніх» випадках.
5. Перед здачею програми до онлайн-системи перевірте, чи виконані всі технічні умови.
6. Після закінчення роботи над певною задачею спробуйте відразу здати її розв'язок до онлайн-системи, а не залишати це на кінець туру, але пам'ятайте, що кількість спроб здачі розв'язків обмежена.
7. У разі, якщо вашу програму не було прийнято системою та причина не очевидна з реакції системи, серед іншого перевірте:
8. Зауважте, що існує різниця у введенні та виведені 64-бітних цілих чисел у компіляторі `GCC` через `printf` / `scanf`: у тестувальній системі використовується формат `%Ld` (або `%Lu`), тоді як у Windows: `%I64d` (або `%I64u`). Або використовуйте механізми введення-виведення `C++` (`iostream` / `fstream`).
   (a) Відсутність залежності програми від нестандартних модулів чи файлів заголовків.
   (b) Назви файлів вхідних та вихідних даних, наприклад, що вони не містять великих літер.
   (c) Формат вхідних та вихідних даних, наприклад, що наприкінці рядка явно не виводиться символ із кодом 13.
   (d) Розмір вихідного файлу розв'язку та час компіляції.

Питання:

Ви можете ставити питання за допомогою системи. Зверніть увагу, що питання мають бути такими, на які можна відповісти «Так» або «Ні». Ви можете отримати одну з наступних відповідей:

- «Відповідь в умові», якщо на ваше питання можна відповісти, прочитавши умову або пам'ятку.
- «Без коментарів», якщо питання стосується інформації, яку журі не бажає розголошувати, наприклад, методу розв'язання задачі.
- «Незрозуміле питання», якщо журі не зрозуміло ваше питання.
- «Так» або «Ні».

Наприклад, якщо ви поставите питання «Граф орієнтований чи неорієнтований?», то відповідь у будь-якому випадку буде «Без коментарів» через те, що на це питання неможливо відповісти «Так» чи «Ні».

Задача A. Сума

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Дано два числа: a та b. Потрібно знайти їхню суму.

Протокол взаємодії

Вам потрібно реалізувати одну функцію:

```
integer solve(integer a, integer b)
```

- a — перше число;
- b — друге число;
- ця функція має повертати одне ціле число — суму двох чисел.

Формат вхідних даних

Перший рядок містить два цілі числа a та b ($-10^{18} \leqslant a, b \leqslant 10^{18}$) — числа, які потрібно додати.

Формат вихідних даних

Буде виведено одне число — суму двох чисел.

Приклади

Вхідні дані	Вихідні дані
3 -9	-6
10 50	60

Оцінювання

1. (10 балів) $-100 \leqslant a, b \leqslant 100$;
2. (30 балів) $-10^9 \leqslant a, b \leqslant 10^9$;
3. (60 балів) без додаткових обмежень.

Задача B. Сума масиву

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Є масив a довжиною n. Знайдіть суму цього масиву.

Протокол взаємодії

Вам потрібно реалізувати одну функцію:

```
integer solve(integer n, array of integers a)
```

- n — розмір масиву;
- a_i ($|a| = n$) — i-те число;
- ця функція має повертати одне ціле число — суму всіх чисел масиву.

Формат вхідних даних

Перший рядок містить одне ціле число n ($1 \leqslant n \leqslant 10^5$) — розмір масиву.
Другий рядок містить n цілих чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant 10^6$) — числа масиву.

Формат вихідних даних

Буде виведено одне ціле число — суму масиву.

Приклади

Вхідні дані	Вихідні дані
5 2 3 8 1 3	17
1 7	7

Оцінювання

1. (5 балів) $n \leqslant 1$;
2. (15 балів) $n \leqslant 100$; $a_i \leqslant 100$;
3. (30 балів) $n \leqslant 10^4$; $a_i \leqslant 10^4$;
4. (50 балів) без додаткових обмежень.

Задача C. Знову сума

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Дано поле, яке розбите на $n \times m$ квадратів. Рядки нумеруються від 1 до n зверху вниз, а стовпці від 1 до m зліва направо. На кожній клітинці записане певне число. Знайдіть суму всіх чисел.

Протокол взаємодії

Вам потрібно реалізувати одну функцію:

```
integer solve(integer n, integer m, integer g)
```

- n та m — розміри поля;
- g — номер блока;
- ця функція має повертати одне ціле число — суму всіх чисел на полі.

Ви можете використовувати функцію:

```
integer getValue(integer x, integer y)
```

- x ($1 \leqslant x \leqslant n$) — номер рядка;
- y ($1 \leqslant y \leqslant m$) — номер стовпчика;
- ця функція повертає число, яке записане в клітинці (x, y).

Формат вхідних даних

Перший рядок містить три цілі числа n, m та g ($1 \leqslant n, m \leqslant 100$, $0 \leqslant g \leqslant 2$) — розміри поля.

Кожний з наступних n рядків містить по m цілих чисел $a_{i1}, a_{i2}, \ldots, a_{im}$ ($1 \leqslant a_{ij} \leqslant 10^9$) — числа на полі.

Формат вихідних даних

Буде виведено одне ціле число — суму всіх чисел.

Приклад

Вхідні дані	Вихідні дані
2 3 0 3 5 9 4 1 10	32

Оцінювання

1. (30 балів) $n \leqslant 1$;
2. (70 балів) без додаткових обмежень.

Задача D. Знайдіть максимум

Обмеження використання часу: 1 секунда
Обмеження використання пам'яті: 256 мегабайтів

Дано поле, яке розбите на $n \times m$ квадратів. Рядки нумеруються від 1 до n зверху вниз, а стовпці від 1 до m зліва направо. Ваш робот знаходиться в клітинці $(1, 1)$.

На кожній клітинці записане певне число. Ваш робот бачить лише те число, яке знаходиться в тій клітинці, на якій він знаходиться. Також він може рухатись вліво, вправо, вниз та вгору. На жаль, він не знає розміри прямокутника.

Знайдіть максимум серед всіх чисел на полі, використовуючи робота.

Протокол взаємодії

Вам потрібно реалізувати одну функцію:

```
integer solve()
```

- ця функції має повертати одне ціле число — максимальне число на полі.

Ви можете використовувати наступні функції:

```
integer getValue()
```

- ця функції повертає одне ціле число — число в клітинці, на якій знаходиться робот.

```
boolean canMoveLeft()
boolean canMoveRight()
boolean canMoveUp()
boolean canMoveDown()
```

- кожна з цих функцій повертає `true`, якщо робот може пересунутись на клітинку в певному напрямку (тобто він не вийде за межі поля, якщо рухатиметься в тому напрямку), або `false` в іншому випадку.

```
void moveLeft()
void moveRight()
void moveUp()
void moveDown()
```

- кожна з цих функцій пересуває робота на одну позицію в певному напрямку.

Формат вхідних даних

Перший рядок містить два цілі числа n та m ($1 \leqslant n, m \leqslant 100$) — розміри поля.

Кожний з наступних n рядків містить по m цілих чисел $a_{i1}, a_{i2}, \ldots, a_{im}$ ($1 \leqslant a_{ij} \leqslant 10^9$) — числа на полі.

Формат вихідних даних

Буде виведено одне ціле число — максимум серед всіх чисел.

Приклад

Нехай $n = 2$, $m = 3$ та $a =$

```
1 3 4
8 1 5
```

Якщо робот знаходиться в клітинці $(1, 1)$, то функції `canMoveLeft` та `canMoveUp` повернуть `false`, а `canMoveRight` та `canMoveDown` повернуть `true`. А якщо в тій позиції викликати функцію `getValue`, то вона поверне 1.

Якщо ви викликаєте функцію `moveDown`, то ваш робот буде в позиції $(2, 1)$. Тоді функція `canMoveDown` поверне `false`, а `getValue` — 8.

Зверніть увагу, що приклад не рішає задачу, а лише показує, як потрібно використовувати функції.

Оцінювання

1. (до 50 балів) $n \leqslant 1$;
    - припустимо, що c — мінімальна кількість рухів, які потрібно зробити, щоб точно знайти відповідь, а t — максимальна кількість рухів, які ви зробили в усіх тестах, тоді ви отримаєте $\max(\lfloor 50 - \sqrt{t - c} \rfloor, 0)$ балів;
2. (до 50 балів) без додаткових обмежень;
    - припустимо, що c — мінімальна кількість рухів, які потрібно зробити, щоб точно знайти відповідь, а t — максимальна кількість рухів, які ви зробили в усіх тестах, тоді ви отримаєте $\max(\lfloor 50 - \sqrt[3]{t - c} \rfloor, 0)$ балів.

Задача E. Запити

Обмеження використання часу:	1 секунда
Обмеження використання пам'яті:	256 мегабайтів

У вас є масив a довжиною n. Вам дають q запитів двох типів:

1. змінити p-те число в масиві a на v;
2. знайти p-те число в масиві a.

Протокол взаємодії

Вам потрібно реалізувати три функції:

```
void init(integer n, array of integers a)
```

- n — довжина масиву;
- a — масив цілих чисел;
- ця функція викликається першою лише один раз. Вона потрібна для того, щоб повідомити вам розмір масиву та сам масив. Лише після виклику цієї функції, будуть викликатись інші дві.

```
void upd(integer p, integer v)
```

- p — позиція;
- v — нове число;
- ця функція викликається, коли потрібно задати запит першого типу.

```
integer ask(integer p)
```

- p — позиція;
- ця функція викликається, коли потрібно задати запит другого типу;
- функція має повернути ціле число — відповідь на запит.

Формат вхідних даних

Перший рядок містить два цілі числа n та m ($1 \leqslant n \leqslant 100$, $1 \leqslant m \leqslant 1\,000$) — довжина масиву та кількість запитів.

Другий рядок містить n цілих чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant 100$) — числа масиву.

Кожний з наступних m рядків описує запит і має один з наступних форматів:

1. «`1 p v`» ($1 \leqslant p \leqslant n$, $1 \leqslant v \leqslant 100$) — позиція та нове число.
2. «`2 p`» ($1 \leqslant p \leqslant n$) — позиція.

Формат вихідних даних

Після кожного запиту другого типу буде виведене одне ціле число — відповідь на нього.

Приклад

Вхідні дані	Вихідні дані
5 7 1 2 3 4 5 2 3 1 1 3 1 3 9 2 1 2 2 2 3 2 5	3 3 2 9 5

Оцінювання

1. (20 балів) $n \leqslant 10$; $m \leqslant 100$; $a_i, v_i \leqslant 20$;
2. (20 балів) $n \leqslant 30$; $m \leqslant 300$; $a_i, v_i \leqslant 30$;
3. (20 балів) $n \leqslant 50$; $m \leqslant 500$; $a_i, v_i \leqslant 50$;
4. (20 балів) $n \leqslant 75$; $m \leqslant 750$; $a_i, v_i \leqslant 75$;
5. (20 балів) без додаткових обмежень.

Задача F. Знову сума 2

Назва вхідного файлу:	`f.in`
Назва вихідного файлу:	`f.out`
Обмеження використання часу:	1 секунда
Обмеження використання пам'яті:	256 мегабайтів

Дано поле, яке розбите на $n \times m$ квадратів. Рядки нумеруються від 1 до n зверху вниз, а стовпці від 1 до m зліва направо. На кожній клітинці записане певне число. Знайдіть суму всіх чисел.

Формат вхідних даних

Перший рядок містить три цілі числа n, m та g ($1 \leqslant n, m \leqslant 100$, $0 \leqslant g \leqslant 2$) — розміри поля.

Кожний з наступних n рядків містить по m цілих чисел $a_{i1}, a_{i2}, \ldots, a_{im}$ ($1 \leqslant a_{ij} \leqslant 10^9$) — числа на полі.

Формат вихідних даних

Потрібно вивести одне ціле число — суму всіх чисел.

Приклад

f.in	f.out
2 3 0 3 5 9 4 1 10	32

Оцінювання

1. (30 балів) $n \leqslant 1$;
2. (70 балів) без додаткових обмежень.