Trabajo práctico 3

Fecha límite de entrega: Viernes 4 de noviembre, hasta las 17:00 hs.
Fecha estimada de devolución: Tres semanas después de la fecha en la que es entregado el TP.
Primer fecha de reentrega: Dos semanas después de devueltas las correcciones, hasta las 17:00 (viernes) o hasta las 22 (martes).
Segunda fecha de reentrega: Viernes 9 de diciembre, hasta las 17:00 hs.

Este trabajo práctico consta de 4 problemas. Para aprobar el mismo se requiere aprobar todos los problemas.
Los requisitos de aprobación del mismo así como los criterios de corrección están detallados en las pautas de aprobación que pueden encontrar en el repositorio de la materia.

En este trabajo práctico se evaluarán contenidos de algoritmos y problemas sobre grafos. Los problemas que serán evaluados en el mismo serán sobre:

- Estructuras de datos para hacer consultas de manera eficiente.
- Problemas sobre cadenas (Strings).

Los ejercicios 1, 2 y 3 de este trabajo práctico tienen soluciones que usan algoritmos y estructuras de datos vistos en clase. Aún si deciden utilizar alguno de estos algoritmos o estructuras deberán demostrar su correctitud de manera tal que puedan demostrar que entienden como y porqué funcionan.

Problema A: Ayudando a los gorilas

El gorila tibetano es un animal en extinción, es por esto que la Asociación Liberadora de GORIlas del Tibet y MOngolia (A.L.GORI.T.MO.) tiene un plan para salvar a la especie.

La ALGORITMO tiene una lista de nombres de los gorilas que necesitan salvar. Cuando un miembro de la ALGORITMO salva a un gorila, este anota el apodo del gorila y manda un mensaje a las oficinas centrales para que se fijen a qué gorila salvó. Ahí, como sólo tienen el nombre de cada gorila, necesitan saber a qué gorila pertenece dicho apodo. Si bien no saben de qué gorila se trata, se lo imaginan. El apodo de un gorila es parte del nombre del gorila, al que le recortan un prefijo y un sufijo. Por ejemplo, si el gorila se llama "Aristoteles", entonces su apodo podría ser, por ejemplo, "tote".

Su tarea es escribir un programa que le sirva a la ALGORITMO para que, dado un nombre de un gorila, y un apodo, el programa decida si puede ese apodo corresponderse con el gorila que tiene ese nombre. En caso de que esto sea posible, imprimir una letra 'S', en caso de que no lo sea, imprimir una 'N'.

El nombre del gorila será un string de letras del alfabeto inglés (es decir, no puede haber enies) mayúsculas y minúsculas, de $N > 0$ caracteres (sin espacios ni ningún otro símbolo que no sean letras). El apodo será un string, también conformado por letras del alfabeto inglés (mayúsculas y/o minúsculas) de longitud $0 < A \leq N$.

El algoritmo debe tener una complejidad temporal $\mathbf{O(}N\mathbf{)}$.

Entrada

La entrada de cada caso de prueba constará en cada caso de exactamente dos líneas.

La primera línea tendrá únicamente de un string con el nombre del gorila con el formato descripto anteriormente, y la segunda línea contendrá el apodo que se desea verificar si se corresponde con dicho gorila.

La entrada contará con el siguiente formato:

```
Nombre
Apodo
```

Salida

La salida debe constar de una única línea que indique si el apodo es un apodo válido para ese nombre (con una letra 'S') o no (con una letra 'N'), la salida contará con el siguiente formato:

```
R
```

siendo R dicha respuesta escrita con un sólo caracter (sin las comillas, sólo la letra correspondiente).

Entrada de ejemplo 1	**Salida para la entrada de ejemplo 1**
ARISTOTELES TOTE	S

Entrada de ejemplo 2	**Salida para la entrada de ejemplo 2**
Aristoteles Tote	N

Entrada de ejemplo 3	**Salida para la entrada de ejemplo 3**
algoritmo ritmo	S

Entrada de ejemplo 4	**Salida para la entrada de ejemplo 4**
holamundo holmund	N

Pesos mínimo y máximo: 7 y 10.

Cotas recomendadas para testear: Se recomienda testear el problema con valores de $N \leq 10^5$.

Problema B: Buscando a los alumnos

Este cuatrimestre decidimos que la materia Problemas, Algoritmos y Programación tenga sólo trabajos prácticos, ya que no fue una buena experiencia cuando se tomaron parciales.

El problema que hubo el cuatrimestre pasado cuando se decidió tomar parciales, fue que era muy difícil avisarle a los alumnos de sus notas, ya que no recordabamos las direcciones de correo de los alumnos y no teníamos como avisarles si habían aprobado o no. El cuatrimestre que viene, gracias a que ustedes nos ayudarán a resolver este problema, podremos volver a tomar parciales.

Para evitar esta confusión, les pasamos la lista de las direcciones de correo electrónico de todos los alumnos que cursarán el cuatrimestre que viene, omitiendo la última parte que es común a todos (@dc.uba.ar).

También les pasamos el prefijo que recordaremos de cada dirección (por ejemplo, si el mail es cosmefulanito@dc.uba.ar, les pasaremos cosmefulanito como dirección, y el prefijo que recordaremos será cualquier prefijo no vacío de dicho string, por ejemplo, cosme o cosmeful entre otras opciones).

Con esta información, ustedes nos dirán cuál es el mínimo valor de T que cumpla que para cada dirección de correo electrónico que les pasemos junto con su prefijo, existen a lo sumo T direcciones que comparten ese prefijo. De esta manera, configuraremos nuestro cliente de correo electrónico para que al tipear un prefijo nos muestre T direcciones que tengan ese prefijo (o todas en caso de que menos de T tengan el prefijo) y así podamos ver la correcta, recordarla y elegirla para notificar al alumno de su nota.

El algoritmo debe tener una complejidad temporal $\mathbf{O}(S)$ siendo S la suma de las longitudes de los correos electrónicos de todos los alumnos ($|D_1| + ... + |D_A|$ según la descripción de la entrada).

Entrada

La entrada de cada caso de prueba constará en cada caso de varias líneas.

La primera línea constará de un entero A indicando la cantidad de alumnos.

A dicha línea le seguirán A líneas con un string D_i que indica la dirección de correo del i-ésimo alumno hasta antes de la @ y constará de $1 \leq |D_i|$ caracteres que serán letras mayúsculas y minúsculas del alfabeto inglés (no hay enies) y un entero $1 \leq P_i \leq |D_i|$ que indicará la longitud del prefijo que recordaremos los docentes (por ejemplo si la dirección es cosmefulanito@dc.uba.ar y solo recordamos cosme, esa línea será cosmefulanito 5)

La entrada contará con el siguiente formato:

```
A
D_1 P_1
D_2 P_2
...
D_A P_A
```

Salida

La salida debe constar de un único entero T, el mínimo valor que nos asegure que para cada dirección de correo de uno de los alumnos, y el prefijo que recordamos de dicha dirección, hay a lo sumo T direcciones que comienzan con ese prefijo, con el siguiente formato:

```
T
```

Entrada de ejemplo 1	Salida para la entrada de ejemplo 1
6 mimaileseste 3 mimailesesteotro 14 estemailesmio 5 holaesteesmimail 10 miramimail 2 miramiotromail 4	4

Entrada de ejemplo 2	Salida para la entrada de ejemplo 2
4 aaaaa 1 bbbbb 1 ccccc 1 ddddd 4	1

Entrada de ejemplo 3	Salida para la entrada de ejemplo 3
7 estos 4 estosmails 5 estosmailsno 6 estosmailsnoson 7 estosmailsnosonlibres 8 estosmailsnosonlibresde 9 estosmailsnosonlibresdeprefijos 10	7

Pesos mínimo y máximo: 8 y 10.

Cotas recomendadas para testear: Se recomienda testear el problema con valores de $S \leq 10^5$.

Problema C: Ciencia argentina

La Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires (FCEN-UBA) siempre se ha destacado por formar excelentes científicos. En un contexto en el que los resultados de la investigación científica ayudan a lograr el crecimiento de nuestro país, debemos estar seguros de que nuestros docentes e investigadores reciban el reconocimiento que les corresponde.

Para estar seguros de que los sueldos docentes son realmente adecuados, un grupo de docentes nos ha pedido que les ayudemos a decidir si la escala de sueldos es apropiada o no.

Los sueldos están basados en dos datos de cada docente: su cargo y su antigüedad. Existen C cargos distintos y A niveles de antigüedad distintos (la antigüedad depende de los años que lleve el docente ejerciendo la docencia y no es algo continuo sino que se divide en escalas).

El sueldo está dividido en varios componentes. Si un docente tiene el c-ésimo cargo (empezando desde el de menor jerarquía y ordenados de manera creciente por jerarquía) y la a-ésima antigüedad (empezando por la menor antigüedad y en orden creciente en la escala), el docente cobrará la suma de todos los montos $M_{i,j}$ con $1 \leq i \leq c$ y $1 \leq j \leq a$.

La pregunta que nos hacen para evaluar la situación es la siguiente: Dados el cargo c_1 y la antigüedad a_1 de un docente, y el cargo c_2 y la antigüedad a_2 a la que aspira a llegar para fin de año (asumiendo que se conserva la escala salarial) ¿Qué parte de su sueldo cobraría, si tuviera cargo c_2 y antigüedad a_2, pero no podría cobrar teniendo cargo c_1 (sin importar si consigue la antigüedad a_2) o antigüedad a_1 (sin importar si tiene el cargo c_2).

La respuesta debe ser la suma de los montos $M_{i,j}$ que son parte del sueldo de un docente con cargo c_2 y antigüedad a_2, y que no pueden cobrar docentes con cargo c_1 o antigüedad a_1 (sin importar lo que cobre un docente con mejor cargo que c_2 o mayor antigüedad que a_2).

Por ejemplo, si un docente tiene cargo 4 y antigüedad 3, y aspira a cargo 6 y antigüedad 8, la respuesta deberá ser:

$$\sum_{i=5}^{6} \sum_{j=4}^{8} M_{i,j}$$

El algoritmo debe tener una complejidad temporal $\mathbf{O}(AC + Q)$ donde A es la cantidad de niveles en la escala de antigüedad, C es la cantidad de cargos docentes distintos que existen en la facultad, y Q es la cantidad de queries (consultas) que recibimos como las descriptas anteriormente.

Entrada

La entrada de cada caso de prueba constará en cada caso de varias líneas.

La primera línea constará de tres enteros C, A y Q que indican la cantidad de cargos docentes, niveles en la escala de antigüedad y consultas.

Las siguientes C líneas constarán de A enteros cada una, siendo el j-ésimo entero de la i-ésima de estas líneas (indexando siempre desde 1) el entero $M_{i,j}$.

Luego seguirán Q líneas con 4 enteros c_1, a_1, c_2, a_2 cada una que representarán lo descripto anteriormente.

La entrada contará con el siguiente formato:

```
C A Q
M11 M12 ... M1A
M21 M22 ... M2A
...
MC1 MC2 ... MCA
c1(1) a1(1) c2(1) a2(1)
c1(2) a1(2) c2(2) a2(2)
...
c1(Q) a1(Q) c2(Q) a2(Q)
```

Siendo por ejemplo c1(i) el valor de C_1 en la i-ésima consulta.

Salida

La salida debe constar de Q líneas, una para cada consulta, indicando la respuesta a dicha consulta con el siguiente formato:

```
R1
R2
...
RQ
```

siendo R_i la respuesta a la i-ésima consulta.

Entrada de ejemplo 1	**Salida para la entrada de ejemplo 1**
5 5 4 1 2 3 4 5 2 3 4 5 6 3 4 5 6 7 4 5 6 7 8 5 6 7 8 9 2 2 4 4 3 3 3 5 1 1 5 5 1 2 3 4	24 0 96 20

Pesos mínimo y máximo: 6 y 8.

Cotas recomendadas para testear: Se recomienda testear el problema con valores de $AC \leq 10^5$ y $Q \leq 10^5$.

Problema D: Diversión asegurada

Con el correr del cuatrimestre nos fuimos dando cuenta que la materia podría tener menos clases de las que tiene y por eso para el año que viene decidimos tomarnos algunos días de vacaciones, durante los cuales suspenderemos las clases de la materia, para retomarlas al volver de las vacaciones.

La idea es viajar por el país en esos días para poder disfrutar de distintos eventos que se realizarán a lo largo y a lo ancho de la Argentina.

Tenemos una idea de las actividades y eventos que habrá en cada lugar del país cada día del año que viene (en el nuevo calendario los años tienen D días y no necesariamente $D = 365$), habiendo exactamente un evento por día, y sabemos qué tanto nos divierte cada uno de esos eventos.

El departamento nos pide que no nos vayamos en momentos críticos y por eso nos va ofreciendo distintos rangos de días del año para poder irnos en algunos de esos rangos. La decisión la tomaremos, basándonos en qué tan divertidos son (sumando sus diversiones) los dos eventos más divertidos dentro de dicho rango (por ejemplo, si hay un evento de diversión 4, otro de divessión 2 y otro de diversión 9 en un rango de tres días, la diversión de ese rango será 4+9 = 13 porque sólo consideramos los dos de mayor diversión). Cada una de estos rangos estará descripto por dos enteros P_i y U_i describiendo el primer y el último día del rango (que por supuesto, es [cerrado,abierto)) y lo que debemos saber para ver si ese rango nos convence o no, es la suma de los dos eventos más divertidos en el rango de días $[P_i, U_i)$.

El algoritmo debe tener una complejidad temporal $\mathbf{O}(D + R \log D)$ siendo D la cantidad de días que tiene el año calendario y R la cantidad de rangos que nos propone el departamento.

Entrada

La entrada de cada caso de prueba constará en cada caso de varias líneas.

La primera línea constará de un entero D indicando la cantidad de días del año, y un entero R indicando la cantidad de rangos ofrecidos por el departamento

A dicha línea le seguirán $R + 1$ líneas. La primera de ellas contendrá N enteros no negativos, indicando el i-ésimo de ellos la diversión del evento del i-ésimo día (indexando desde 0).

Luego, cada una de las restantes R líneas contendrá la descripción de un rango, con dos enteros $0 \leq P_i < U_i \leq D$ siendo $U_i - P_i \geq 2$.

La entrada contará con el siguiente formato:

```
D R
E1 E2 ... ED
P1 U1
P2 U2
...
PR UR
```

Siendo E_i la diversión del evento del i-ésimo día, P_i el primer día (inclusive) del i-ésimo rango, y U_i el último día (no inclusive) del i-ésimo rango.

Salida

La salida debe constar de R enteros, uno para cada rango, indicando la suma de las dos mayores diversiones del rango dado, con el siguiente formato:

```
Q_1
Q_2
...
Q_R
```

siendo Q_i la respuesta de la i-ésima query (asociada al i-ésimo rango).

Entrada de ejemplo 1	**Salida para la entrada de ejemplo 1**
6 4 1 2 3 4 5 6 0 6 1 5 2 4 0 3	11 9 7 5

Entrada de ejemplo 2	**Salida para la entrada de ejemplo 2**
8 2 6 2 4 1 3 5 8 2 2 5 3 8	7 13

Entrada de ejemplo 3	**Salida para la entrada de ejemplo 3**
10 1 0 0 0 0 0 0 0 0 0 0 0 10	0

Pesos mínimo y máximo: 9 y 12.

Cotas recomendadas para testear: Se recomienda testear el problema con valores de $D \leq 10^5$, $R \leq 10^5$.