Estructuras de Datos y Algoritmos

Grados en Ingeniería Informática. Grupos C y F

Examen Primer Cuatrimestre, 15 de Enero de 2019.

Nombre: _____ **Grupo:** _____

Laboratorio: _____ **Puesto:** _____ **Usuario de DOMjudge:** _____

Normas de realización del examen

1. Debes programar soluciones para cada uno de los tres ejercicios, probarlas y entregarlas en el juez automático accesible en la dirección `http://exacrc/domjudge/team`.
2. Escribe comentarios que expliquen tu solución, justifiquen por qué se ha hecho así y ayuden a entenderla. Calcula la complejidad de todas las funciones que implementes.
3. En el juez te identificarás con el nombre de usuario y contraseña que has recibido al comienzo del examen. El nombre de usuario y contraseña que has estado utilizando durante la evaluación continua **no** son válidos.
4. Escribe tu **nombre y apellidos** en un comentario en la primera línea de cada fichero que subas al juez.
5. Tus soluciones serán evaluadas por el profesor independientemente del veredicto del juez automático. Para ello, el profesor tendrá en cuenta **exclusivamente** el último envío que hayas realizado de cada ejercicio.

1. **(3.5 puntos)** Dado un vector de enteros, se desea encontrar el segmento más largo de números consecutivos. Se pide:

   1. (0.75 puntos) Especifica una función que dado un vector de enteros de longitud ≥ 0 devuelva la longitud del segmento más largo del vector formado por una secuencia de números consecutivos.
   2. (1.5 puntos) Diseña e implementa un algoritmo iterativo que resuelva el problema propuesto.
   3. (0.75 puntos) Escribe el invariante del bucle que permite demostrar la corrección del mismo y proporciona una función de cota.
   4. (0.5 puntos) Indica el coste asintótico del algoritmo en el caso peor y justifica adecuadamente tu respuesta.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá la longitud del vector, y a continuación los valores de tipo `int` que contiene el vector.

Salida

Por cada caso de prueba el programa escribirá una línea con la longitud del segmento más largo solicitado.

Entrada de ejemplo

```
6
0
1
5
2
3 2
10
2 3 2 3 4 5 6 5 4 3
10
1 1 2 3 4 8 7 6 5 6
10
-2 2 2 2 2 2 2 2 2 2
```

Salida de ejemplo

```
0
1
2
10
5
1
```

2.(2.5 puntos) Dado un vector formado por una secuencia de 1 seguido de una secuencia de 0, se desea averiguar el número de 0s que contiene. Se pide:

1. (1.5 puntos) Escribe un algoritmo recursivo eficiente (mejor que lineal) que permita resolver el problema para un vector dado.
2. (1 punto) Escribe la recurrencia que corresponde al coste de la función recursiva e indica a qué orden de complejidad asintótica pertenece dicho coste.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá en una línea el tamaño del vector y a continuación en la siguiente línea los valores que contiene el vector.

Recuerda: el número de 0s no se ha de calcular durante la lectura de los datos.

Salida

Por cada caso de prueba el programa escribirá el número de 0s.

Entrada de ejemplo

```
6
0
1
1
2
1 0
2
1 1
10
1 1 1 1 1 1 1 0 0 0
10
0 0 0 0 0 0 0 0 0 0
```

Salida de ejemplo

```
0
0
1
0
3
10
```

3. (4 puntos) El 13 de Julio de 2019 se va a celebrar un concierto benéfico de rock como el que tuvo lugar hace 34 años en Wembley. Los artistas participantes ya están confirmados y solamente falta decidir el orden de actuación de los mismos. Los promotores del concierto han realizado una estimación de la cantidad de donaciones que se pueden recibir durante la actuación de cada uno de los n artistas dependiendo del momento 0 a $n-1$ en el que actúan. También disponen de una tabla de "vetos" en la que cada artista ha reflejado si admite tocar o no inmediatamente después de cada uno de los demás. Por ejemplo Queene no acepta tocar después de nadie mientras que U3 acepta tocar solamente después de Chimpanzeez. Ayuda a los promotores a determinar el orden en que han de tocar los artistas para obtener la máxima donación posible según la estimación realizada.

- (2.5 puntos) Implementa un algoritmo de vuelta atrás que resuelva el problema. Explica claramente los marcadores que has utilizado.
- (1.5 puntos) Plantea dos posibles funciones de poda de optimalidad, razona sobre cual de ellas es mejor e impleméntala en tu algoritmo.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor del número de artistas n. A continuación figuran las estimaciones de las donaciones: una fila para cada artista. Después los vetos de los artistas: una fila para cada artista i indicando si admite (1) o no (0) tocar después del artista j (habrá un 0 en la posición i).

Salida

Por cada caso de prueba el programa escribirá una línea con la donación máxima estimada (suma de las donaciones obtenidas por cada artista en el momento que le corresponde tocar). En caso de que no sea posible satisfacer los vetos se escribirá `NEGOCIA CON LOS ARTISTAS`.

Entrada de ejemplo

```
2
3
10 20 30
140 20 10
160 10 20
0 1 1
0 0 1
0 0 0
3
10 20 30
140 20 10
160 10 20
0 0 1
0 0 1
0 0 0
```

Salida de ejemplo

```
210
NEGOCIA CON LOS ARTISTAS
```