Competitive Programming Workshop: Algoritmos voraces

Giovanny Alfonso Chávez Ceniceros

Universidad Autónoma de Chihuahua
Facultad de Ingeniería

gchavezceniceros@acm.org

21 de marzo de 2019

Contenido

¿Qué es algoritmo *greedy*?

- Se dice que un algoritmo es codicioso si toma la decisión óptima a nivel local en cada paso con la esperanza de llegar a la solución global óptima.
- En algunos casos, la codicia funciona: la solución es corta y se ejecuta de manera eficiente. Para muchos otros, sin embargo, no lo hace.
- Cuando una estrategia voraz falla al producir resultados óptimos en todas las entradas, en lugar de algoritmo suele denominarse heurística.

Devolviendo el cambio

Descripción

El objetivo de este problema es encontrar el número mínimo de monedas necesarias para devolver el cambio del valor de entrada (un número entero) en monedas con denominaciones 1, 5 y 10.

Entrada

Un solo entero m.

Salida

El número mínimo de monedas con denominaciones 1, 5, 10 que se devuelven m.

Devolviendo el cambio

Restricciones

$0 \leq m \leq 10^3$.

Casos de prueba

Entrada	**Salida**
2	2 $(2 = 1 + 1\ )$
28	6 $(28 = 10 + 10 + 5 + 1 + 1 + 1)$

Devolviendo el cambio

```
1  int greedyChange( int money )
2  {
3      int i = 0, change = 0;
4      vector<int> coins{10, 5, 1};
5
6      while(money > 0)
7          while(i < coins.size( ))
8          {
9              if(coins[i] > money)
10                 ++i;
11             else
12             {
13                 money -= coins[i];
14                 change += 1;
15             }
16         }
17
18     return change;
19 }
```

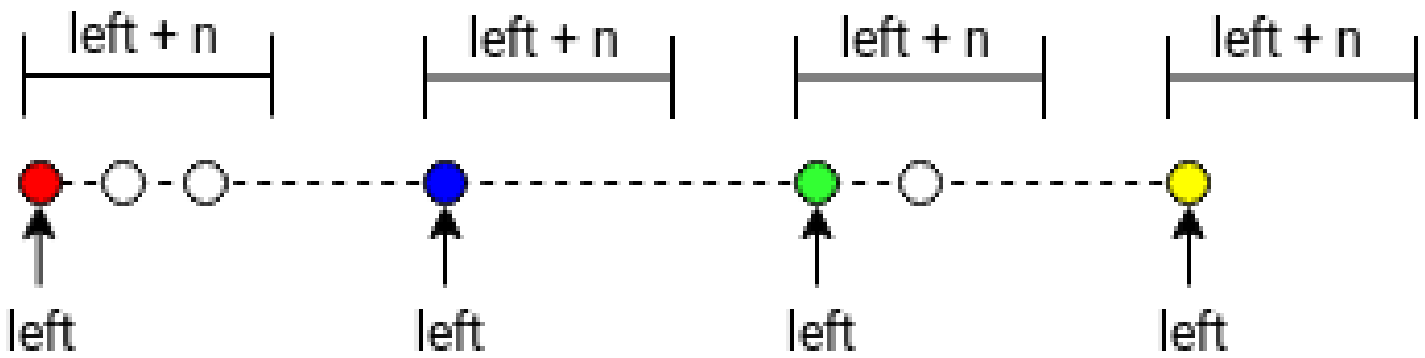
Fiesta de celebración

- Muchos niños vinieron a una fiesta de celebración. Necesitas organizarlos en grupos pero quieres que jueguen bien unos con otros, por lo que los grupos deben ser bastante homogéneos y para eso se desea que dos niños cualquiera en el mismo grupo debe diferir por lo más dos años de edad.
- Por lo tanto, se desea organizarlos en el número mínimo posible de grupos porque para cada grupo necesitas un adulto como supervisor, y quieres minimizar el número de adultos.

Fiesta de celebración

```
1  typedef vector<int>  vi;
2  typedef vector<pair<int, int>>  vii;
3
4  vii pointsCoverSorted( vi x, int n = 2 )
5  {
6      vii segments;
7      int left = 0, l = 0, r = 0;
8
9      while(left < x.size( ))
10     {
11         l = x[left];
12         r = x[left] + n;
13         segments.push_back(make_pair(l,r));
14         left += 1;
15         while(left < x.size( ) && x[left] <= r)
16             left += 1;
17     }
18     return segments;
19 }
```

Fiesta de celebración



Recolectando firmas

- Eres el responsable de recopilar firmas de todos los inquilinos de un edificio. Para cada inquilino, conoces el período de tiempo cuando él o ella está en casa. Te gustaría reunir todas las firmas visitando el edificio tan pocas veces como sea posible.
- El modelo matemático para este problema es el siguiente. Te dan un conjunto de segmentos en una línea y tu objetivo es marcar el menor número de puntos posible en una línea. de modo que cada segmento contenga al menos un punto marcado.

Recolectando firmas

Descripción

Dado un conjunto de n segmentos $[a_0, b_0], [a_1, b_1], ..., [a_{n1}, b_{n1}]$ con coordenadas de enteros en una línea, encuentre el número mínimo de puntos de tal manera que cada segmento contenga al menos un punto. Es decir, encuentre un conjunto de enteros X del tamaño mínimo tal que para cualquier segmento $[a_i, b_i]$ haya un punto $x \in X$ tal que $a_i \leq x \leq b_i$.

Recolectando firmas

Entrada

La primera línea contiene el número n de segmentos. Cada una de las siguientes n líneas contiene dos enteros a_i y b_i (separados por un espacio) que definen las coordenadas de los puntos finales del i-esimo segmento.

Salida

El número mínimo m de puntos y las coordenadas enteras de m puntos (separados por espacios).

Recolectando firmas

Restricciones

$1 \leq n \leq 100$ $0 \leq a_i \leq b_i \leq 10^9$ para toda $0 \leq i \leq n-1$.

Casos de prueba

Entrada	**Salida**
3 1 3 2 5 3 6	1 3

Recolectando firmas

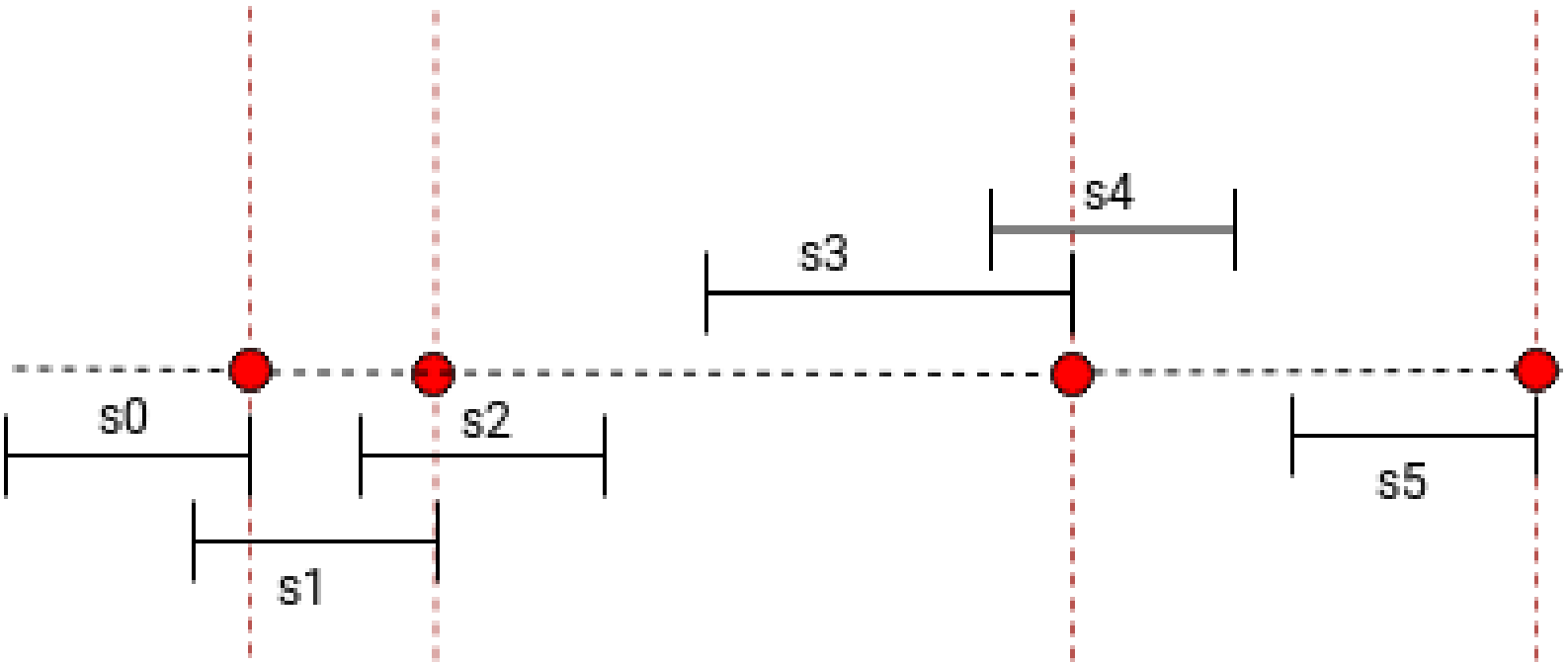
Casos de prueba

Entrada	Salida
4	2 3 6
4 7	
1 3	
2 5	
5 6	

Recolectando firmas

```
1  vi optimalPoints( vii &segments )
2  {
3      // Ordenar los segmentos (x,y) segun yi < yj
4
5      vi points;
6      int point = segments[0].second;
7      points.push_back(point);
8
9      for(int i = 1; i < segments.size( ); ++i)
10         if(point<segments[i].first || point>segments[i].second)
11         {
12             point = segments[i].second;
13             points.push_back(point);
14         }
15
16     return points;
17 }
```

Recolectando firmas



UVa Online Judge

11264 Coin Collector

Our dear Sultan is visiting a country where there are n different types of coin. He wants to collect as many different types of coin as you can. Now if he wants to withdraw X amount of money from a Bank, the Bank will give him this money using following algorithm.

```
withdraw(X){
    if( X == 0) return;
    Let Y be the highest valued coin that does not exceed X.
    Give the customer Y valued coin.
    withdraw(X-Y);
}
```

Now Sultan can withdraw any amount of money from the Bank. He should maximize the number of different coins that he can collect in a single withdrawal.

Input

First line of the input contains T the number of test cases. Each of the test cases starts with n ($1 \leq n \leq 1000$), the number of different types of coin. Next line contains n integers C_1, C_2, ..., C_n the value of each coin type. $C_1 < C_2 < C_3 < \ldots < C_n < 1000000000$. C_1 equals to 1.

Output

For each test case output one line denoting the maximum number of coins that Sultan can collect in a single withdrawal. He can withdraw infinite amount of money from the Bank.

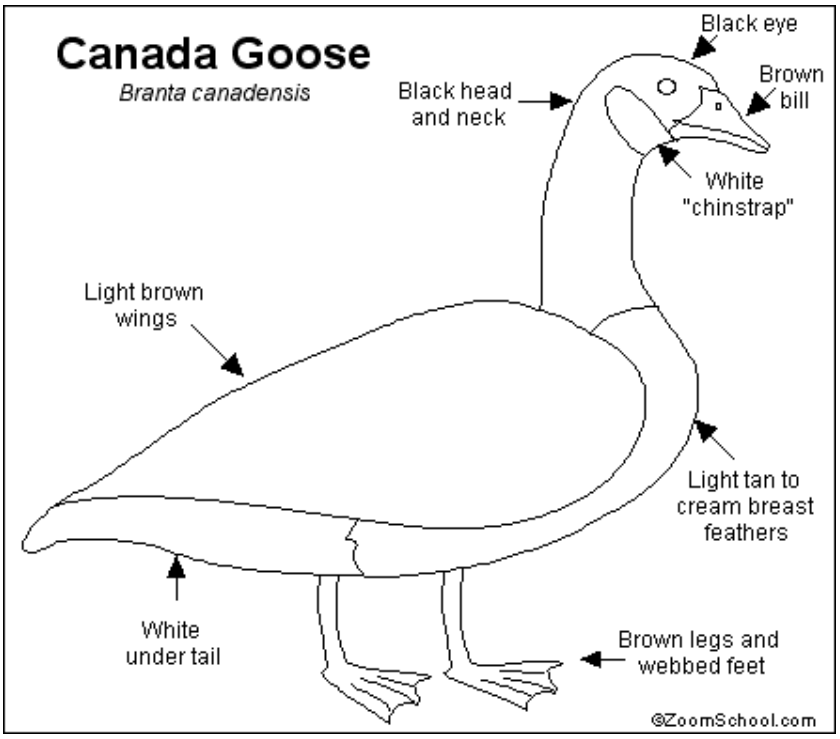
Sample Input

```
2
6
1 2 4 8 16 32
6
1 3 6 8 15 20
```

Sample Output

```
6
4
```

11292 The Dragon of Loowater



Once upon a time, in the Kingdom of Loowater, a minor nuisance turned into a major problem.

The shores of Rellau Creek in central Loowater had always been a prime breeding ground for geese. Due to the lack of predators, the geese population was out of control. The people of Loowater mostly kept clear of the geese. Occasionally, a goose would attack one of the people, and perhaps bite off a finger or two, but in general, the people tolerated the geese as a minor nuisance.

One day, a freak mutation occurred, and one of the geese spawned a multi-headed fire-breathing dragon. When the dragon grew up, he threatened to burn the Kingdom of Loowater to a crisp. Loowater had a major problem. The king was alarmed, and called on his knights to slay the dragon and save the kingdom.

The knights explained: "To slay the dragon, we must chop off all its heads. Each knight can chop off one of the dragon's heads. The heads of the dragon are of different sizes. In order to chop off a head, a knight must be at least as tall as the diameter of the head. The knights' union demands that for chopping off a head, a knight must be paid a wage equal to one gold coin for each centimetre of the knight's height."

Would there be enough knights to defeat the dragon? The king called on his advisors to help him decide how many and which knights to hire. After having lost a lot of money building Mir Park, the king wanted to minimize the expense of slaying the dragon. As one of the advisors, your job was to help the king. You took it very seriously: if you failed, you and the whole kingdom would be burnt to a crisp!

Input

The input contains several test cases. The first line of each test case contains two integers between 1 and 20000 inclusive, indicating the number n of heads that the dragon has, and the number m of knights in the kingdom. The next n lines each contain an integer, and give the diameters of the dragon's heads, in centimetres. The following m lines each contain an integer, and specify the heights of the knights of Loowater, also in centimetres.

The last test case is followed by a line containing '`0 0`'.

Output

For each test case, output a line containing the minimum number of gold coins that the king needs to pay to slay the dragon. If it is not possible for the knights of Loowater to slay the dragon, output the line '`Loowater is doomed!`'.

Sample Input

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

Sample Output

```
11
Loowater is doomed!
```

Referencias


Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2009)
Introduction to algorithms


Skiena, S. S. (2003)
Programing Challenges. The Programming Contest Training Manual