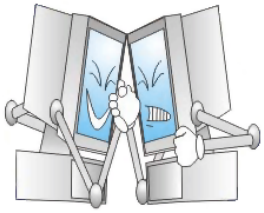

Pripreme za JBOI 2008
02. jul 2008. godine
predavač: Slobodan Mitrović
e-mail: boba5555@gmail.com

Sortiranje i binarno pretraživanje - zadaci

Zadatak 1. [TopCoder MinDifference] Dat je niz od n brojeva ($2 \leq n \leq 100\,000$). Naći najmanju razliku izmedju bilo koja dva broja i ispisati je na ekran.

Ulaz	**Izlaz**
4	1
5 2 8 6	

Zadatak 2. [Regionalno 2004 Jagode] Posle dugoročne saradnje na plantažama Jagoda, farmeri Šomi, Draganče i Djurica su se posvadjali, te su poželeli da podele plantaže. Pošto je Viskonsin (država iz koje su naši farmeri) poznata po tome da ima močvarno zemljište farmeri su pravili plantaže u obliku kvadrata. Da bi se spasili papirologije odlučili su da zemljište podele linijama koje idu od severa ka jugu tako da svako dobije neke od plantaža (i neke delove ostalih). Njihov komšija Zlatević je uzeo plan u ruke i rekao da je problem lako rešiti. Medjutim u medjuvremenu farmer Zlatević se izgubio jureći zlatnu žabu u šumi. Tako da na vama ostaje da pomognete farmerima da reše svoj problem.

U prvom redu ulaza se nalazi ceo broj N ($1 \leq N \leq 200$), koji predstavlja ukupan broj plantaža koje poseduju Šomi, Draganče i Djurica. U svakom od sledećih redova nalaze se tri realna broja X, Y, i A, ($0 \leq X, Y, A \leq 30000$). X, Y predstavljaju koordinate početka plantaže na mapi (Jugo-Zapadni ćošak ili donji levi ugao plantaže), dok A predstavlja dužinu strane plantaže.

Na izlaz, u dva reda treba ispisati dva realna broja, sa tačnošću na dve decimale (tj. sa dve cifre iza decimalne tačke). Oni predstavljaju X koordinatu zamišljenih linija koje dele zemljište.

Ulaz	**Izlaz**
2	67.67
1 1 100	233.33
200 200 100	

Obrazloženje. U datom primeru jednom od farmera će pripasti dve trećine prvog kvadrata (plantaže), drugom trećina prvog i trećina drugog, i najzad trećem će pripasti dve trećine drugog kvadrata. Rešenja su će u svakom test-primeru biti jedinstvena.

Zadatak 3. [Z-Trening takmičenje z-policajac] Mali Z se zaposlio kao policajac, i njegov prvi zadatak je bio da očuva red i mir na velikoj Novogodišnjoj zurci u Beogradskoj Areni. Medjutim, nespretni Z je zaspao čim je došao na posao. Sutradan je morao da podnese izveštaj svom sefu o tome koji je maksimalan broj ljudi koji je u jednom trenutku bio na zurci. Z se bacio na posao i rešio da pozove telefonom svakoga ko je bio na zurci. I svako mu je odgovrio kada je tačno došao, i kada je otišao za zurke. Radi uopštenja vremenski interval od početka do kraja žurke Z je podelio na 1000000 vremenskih intervala (kada je već zabrljao, sada hoće da bude precizan)

Pomozite malom Z-u da sa spiska na kome piše kada je koji gost došao a kada otišao zaključi koliko je najviše ljudi bilo na zurci u istom trenutku.

Sa ulaza se učitava broj N ($0 < N \leq 10000$), N predstavlja broj ljudi koji je bio na žurci. U narednih N linija nalaze se dva broja $A[i]$ i $B[i]$, gde $A[i]$ prestavlja interval u opsegu [1..1000000] kada je osoba i došla na žurku a $B[i]$ interval kada je osoba i otisla za žurke. Što znači da je osoba i bila na žurci u svim intervalima izmedju $A[i]$ i $B[i]$ kao i u intervalima $A[i]$ i $B[i]$.

Na izlaz ispisati jedan broj, maksimalan broj ljudi koji je u istom intervalu (trenutku) bio na žurci.

Ulaz	**Izlaz**
4 1 3 2 3 3 4 5 7	3

Obrazloženje. U vremenskom intervalu 3, osobe 1, 2 i 3 su bile na žurci.

Zadatak 4. [SPOJ ADOMINO] Dominoes have long entertained both game enthusiasts and programmers for quite some time. Many games can be played with dominoes, including multiplayer and single player games. Hari Khan has come up with a single player game. He takes N boxes and arranges them in a row at positions $N1$, $N2$ ... NN. Now he has to place D dominoes ($D \leq N$) in the boxes such that the minimum distance between any two filled boxes is maximized.

The first line of the input contains an integer t, the number of test cases. t test cases follow. The first line of each test case consists of two integers, $N \leq 100000$ and $D \leq N$, separated by a single space. N lines follow, each containing a single integer $Ni \leq 1000000000$, indicating the location of the ith box.

For each test case, output a single line containing a single integer denoting the largest minimum distance achievable between any two boxes with dominoes.

Ulaz	**Izlaz**
1 5 3 1 2 3 4 5	2

Zadatak 5. [Hrvatska izborna Olimpijada 2008 - Glasnici] A long straight road connects two villages. Along the road, N messengers are stationed and, when needed, they exchange messages using mostly their legs, but also their vocal cords and ears. The first messenger (the closest to the first village) has a radio-receiver which he uses to keep track of current ongoings in the country. When he finds out who has been evicted from whichever reality show is currently popular, he starts running as fast as he can to share the unfortunate (or fortunate) news with everyone else. While running, he shouts the name of the evicted person so that any fellow messengers that are close enough can hear him. Meanwhile, the remaining messengers do not merely sit and wait, but also run themselves, all with the selfless goal of sharing the news with everyone as fast as possible. The running and shouting proceeds as follows:

- Each of the messengers may run whenever, in either direction, at a speed of at most 1 unit per second, or may decide not to run at all and stand still.

- All messengers that know the news shout it at all times. One messenger can hear another messenger shouting (and learn the news) if the distance between them is at most K units.

Write a program that, given the initial locations of the messengers, determines the least amount of time (in seconds) needed for all messengers to learn the news. The location of every messenger is given with a positive real number - the distance from the first village. As mentioned above, initially only the first messenger knows the news.

The first line of input contains the real number K ($0 \leq K \leq 1000000$), the largest distance at which two messengers can hear each other, and the integer N ($1 \leq N \leq 100{,}000$), the number of messengers. Each of the following N lines contains one real number D ($0 \leq D \leq 1000000000$), the distance of one messenger from the first village. The distances will be sorted in ascending order. It is possible for multiple messengers to be at the same location.

Output a real number, the least time for all messengers to learn the news. Your output needs to be within 0.01 of the official output.

Ulaz	**Izlaz**
3.000 2 0.000 6.000	1.500
2.000 4 0.000 4.000 4.000 8.000	1.000

Sortiranje i binarno pretraživanje - teorija

Sortiranje

Cilj sortiranja je da dati skup elemenata uredi u neopadajući ili nerastući poredak. (u daljem textu će se podrazumevati da je uvek u pitanju skup/niz uredjen u neopadajućem poretku)

1. **Inferiorni sortovi**

   - **Bubble Sort**
     Neka je dati niz od n elemenata A potrebno urediti u neopadajući poredak. Bubble sort algoritam/pseudo-code

     ```
     dokle god ima razmene
       za svako i od 1 do n-1
         ako je A[i] > A[i + 1]
           razmeni A[i] i A[i + 1]
     ```

     U ovom algoritmu najveći elementi kao mehurići isplivavaju na kraj niza. U najgorem slučaju vremenska složenost $O(n^2)$.

   - **Selection Sort**
     Ovaj algoritam u prvom koraku nadje najmanji medju elementima od pozicije 1 do pozicije n, njih n, i stavi na 1. mesto; u drugom koraku nadje najmanji medju elementima od pozicije 2 do n, njih n-1, i stavi na 2. mesto;..., sve dok ne preostane samo jedan elemenat.

```
za svako k od 1 do n
  nadji najmanji od k do n i postavi ga na mesto k
```

Vremenska složenost $O(n^2)$.

- **Insertion Sort**
  Ovaj algoritam u k-tom koraku garantuje da su prvih k elemenata niza sortirani. Prvih $k+1$ elemenata sortira na taj način sto $(k+1)$. elemenat umetne na pravo mesto medju prethodnih k elemenata.
  U najgorem slučaju vremenska slozenost $O(n^2)$.

2. **Superiorni sortovi**

- **Quick Sort**
  Divide and conquer algoritam. Troši više memorije nego inferiorni sortovi, ali radi brže. Vremenska složenost u srednjem slučaju $O(nlog(n))$. Pseudo-code algoritma:

```
Procedure qsort(idx1, idx2)
  Izaberemo pivot p u intervalu [idx1, idx2]
  Vece elemente od p prebacimo desno, a manje ili jednake levo
  qsort(idx1, pozicija(p) - 1)
  qsort(pozicija(p) + 1, idx2)
```

  Mana: ako se pivot loše izabere, vremenska složenost moze da postane $O(n^2)$.

- **Merge Sort**
  Ideja algoritma je da u svakom koraku podeli niz na dva podniza sa razlikom u broju elemenata ne većom od jedan, rekurzivno sortira dobijene podnizove i kasnije ih spoji u jedan sortirani. Podela se odvija dok se ne dobije podniz sa jednim elementom jer je takav podniz jednoznačno sortiran.
  Uvek daje vremensku složenost $O(nlog(n))$.

3. **Counting sort**
   Ako različitih dopustivih elemenata u skupu ima relativno malo, onda se svaki element može prebrojati, a zatim počevši od najmanjeg elementa, redom svaki elemenat vratiti onoliko puta koliko se ponavlja.
   Primer: Dat je niz dužine n ($1 \leq n \leq 1\ 000\ 000$) koji se sastoji od brojeva 0, , 9. Sortirati dati niz.

Pretraga

Cilj pretrage je da proveri da li je dati elemenat sadržan u datom skupu podataka i ako se nalazi da vrati poziciju elementa.

- **Prolazak kroz sve elemente skupa**
  Podrazumeva jednostavnu iteraciju kroz sve elemente i proveravajući da li se dati elemenat nalazi u nizu.
  U najgorem slučaju vremenska složenost $O(n)$.

- **Pretraga sa graničnikom (santinelom)**
  Na kraj niza se dodaje graničnik, elemenat koji se traži, pa smo sigurni da cemo naići na traženi elemenat, a shodno poziciji na kojoj smo našli traženi elemenat odredjujemo da li se on nalazi u nizu ili ne. Na ovaj način može da se izbegne dodatno proveravanje da li se elemenat uopšte nalazi u nizu.

- **Binarna pretraga**
  Ako je niz u kojem tražimo elemenat sortiran, to nam daje dodatnu pogodnost i možemo u tom slučaju koristiti binarnu pretragu. Ideja binarne pretrage je da u svakom koraku proveri srednji elemenat podniza u kojem pretražujemo. Ako je elemenat koji proveravamo jednak elementu koji tražimo, onda se pretraga završila. Ako je elemenat koji proveravamo veći od elementa koji tražimo, pošto je podniz sortiran, onda nastavljamo traženje u podnizu koji je levo od elementa kojeg proveravamo, inače nastavljamo pretragu u podnizu desno. Ovo daje vremensku složenost pretrage $O(log(n))$.

  ```
  low = 1, high = n
  dok low <= high
    mid = (low + high) / 2
    if (A[mid] = trazeni)
      kraj
    else if (trazeni < A[mid])
      high = mid - 1
    else
      low = mid + 1
  ```

  Primer: Koristeći binarnu pretragu, naći *arcos* za datu vrednost na intervalu $[0, pi]$ i na intervalu $[pi, 2*pi]$. Da li je moguće koristeći binarnu pretragu naći *arcos* na intervalu $[pi/2, 3*pi/2]$?

Napomena. Deo sa teorijom je korišćen kao podsetnik predavaču, stoga može koristiti kao mali podsetnik učenicima koji su prisustvovali predavanju, a ne kao odgovarajući materijal za učenje.