Control PRO2 - Torn 1 (Tardor 2017) X17042_ca

El sistema jutge.org (d'ara en endavant Jutge) és un entorn d'aprenentatge virtual en què els estudiants poden trobar diferents col.leccions de problemes. Donat un problema, un estudiant pot realitzar diversos enviaments amb possibles solucions. Un enviament es caracteritza per un conjunt d'atributs, com ara la identitat de l'estudiant que va realitzar l'enviament, la data i hora en què es va lliurar la solució, i el veredicte del Jutge respecte als jocs de proves del problema, que s'utilitzen per comprovar la correcció de la solució lliurada.
Alguns exàmens d'assignatures de programació es realitzen i avaluen parcialment utilitzant el Jutge. La qualificació automàtica dels enviaments d'un estudiant es calcula seleccionant en primer lloc **el millor dels enviaments** i multiplicant a continuació el nombre de jocs de proves superats per aquest enviament per una constant donada (per ex., 2.5 si el problema conté quatre jocs de proves). El millor enviament d'un estudiant és l'enviament d'aquest estudiant que supera el màxim nombre de jocs de proves. Si l'estudiant ha fet diversos enviaments que superen el màxim nombre de jocs de proves, el millor enviament d'aquest estudiant és l'últim enviament que supera el màxim nombre de jocs de proves.
En aquest exercici, construirem un programa que llegeixi una seqüència d'enviaments de solucions per a un problema del Jutge; els emmagatzemi en un vector v d'enviaments; els ordeni en ordre creixent pel DNI de l'estudiant que va fer l'enviament i en ordre creixent pel temps de lliurament en el cas en què dos enviaments siguin del mateix estudiant; i els escrigui ordenats d'aquesta manera en la pantalla.
La seqüència d'enviaments és precedida pel nombre d'estudiants que poden realitzar enviaments, és a dir, el nombre d'estudiants matriculats al curs del Jutge a què pertany el problema, i acaba amb un enviament d'un estudiant inexistent amb un DNI igual a 0 (vegeu el fitxer corresponent a l'entrada de l'exemple d'aquest enunciat).
A continuació, el programa rebrà una seqüència d'instruccions, executarà les operacions corresponents a cada instrucció i acabarà quan rebi la instrucció "fi".
La instrucció "consultar" requereix llegir un enter x corresponent al DNI de l'estudiant els enviaments del qual es volen consultar. Si no hi ha cap enviament d'aquest estudiant en el vector v, el programa ho indica mitjançant un missatge; altrament, escriu els enviaments de l'estudiant amb el DNI x que conté el vector v en ordre creixent pel temps de lliurament de l'enviament.
La instrucció "classificar" escriurà el subconjunt de v format pel millor enviament de cada estudiant ordenat de manera que estiguin junts tots els enviaments que superen el mateix nombre de jocs de proves.
La primera vegada que s'introdueixi la instrucció "classificar" caldrà construir una matriu de classificació m amb els millors enviaments del vector v, per poder escriure el contingut d'aquesta matriu de classificació a la pantalla. Se suposa que en tot moment el vector v està ordenat en ordre creixent pel DNI de l'estudiant que realitza l'enviament, i que els enviaments d'un mateix estudiant en el vector v estan ordenats al seu torn en ordre creixent pel temps de lliurament al Jutge.
La matriu de classificació m conté un sol enviament per estudiant, que és, a més, el millor dels enviaments que conté el vector v d'aquest estudiant. Donats dos enviaments e_1 i e_2 d'un mateix estudiant, e_1 és millor que e_2 si e_1 supera més jocs de proves que e_2, o si e_1 i e_2 superen el mateix nombre de jocs de proves però e_1 és més recent que e_2 (és a dir, el temps de lliurament de e_1 és més gran que el de e_2).

Els enviaments de la matriu de classificació m estan classificats, al seu torn, pel nombre de jocs de proves que superen, de manera que cada fila k de la matriu de classificació m ha de contenir únicament enviaments que superin exactament k jocs de proves. Dins de cada fila de la matriu m, els enviaments estan ordenats en ordre creixent pel DNI de l'estudiant que va realitzar l'enviament.
Al fitxer de sortida de l'exemple d'aquest enunciat podeu observar la matriu de classificació corresponent als 29 lliuraments rebuts. Els lliuraments que apareixen a continuació de la frase "0 jocs de proves superats" són els lliuraments de la fila 0 de la matriu de classificació m. Els lliuraments que apareixen a continuació de la frase "1 jocs de proves superats" són els lliuraments de la fila 1 de la matriu de classificació, i així successivament.
Per implementar aquest programa hem construït la classe *Lliurament* (que permet representar un enviament d'un estudiant) i un mòdul funcional *Eines_Vec_Lliu* (que permet realitzar diferents operacions amb vectors d'objectes de la classe `Lliurament`). Podeu consultar l'especificació i la representació del tipus de la classe *Lliurament* al fitxer `Lliurament.hh`, i l'especificació del mòdul funcional *Eines_Vec_Lliu* al fitxer `Eines_Vec_Lliu.hh`.
Tenint en compte tot això, heu d'implementar eficientment el mètode estàtic i públic `millor` de la classe *Lliurament*, que determina si l'enviament representat per $e1$ és millor que l'enviament representat per $e2$.

```
static bool millor(const Lliurament& e1, const Lliurament& e2);
/* Pre: e1 i e2 han estat lliurats pel mateix estudiant.  */
/* Post: Retorna true a algun dels casos següents: 1) e1 ha superat més
 jocs de proves que e2; 2) e1 i e2 han superat el mateix nombre de jocs
 de proves, i el temps de lliurament de e1 és més gran que el temps de
 lliurament de e2. En altres casos, retorna false. */
```

i l'acció `classifica` del mòdul funcional *Eines_Vec_Lliu*, que construeix la matriu de classificació m descrita anteriorment a partir d'un vector v d'objectes de la classe `Lliurament`, que en el rang $v[0, \ldots, n\_lliu - 1]$ està ordenat en ordre creixent pel DNI de l'estudiant que va realitzar l'enviament i en ordre creixent pel temps de lliurament en el cas en què dos enviaments siguin del mateix estudiant.

```
void classifica(int n_lliu, const vector<Lliurament>& v,
     vector<vector<Lliurament> >& m);
  /* Pre: 0 <= n_lliu <= v.size(), v[0 ... n_lliu -1] està ordenat en ordre
  creixent per número de DNI i, en cas d'empat, per temps de lliurament.
  m=M, M.size() = 1 + Lliurament::nombre_jps() i M[j].size()=0 per a tot j */
  /* Post: Per cada x tal que v[0, ..., n_lliu - 1] conté com a mínim un
  lliurament amb DNI = x, m conté el millor lliurament amb DNI = x de
  v[0, ..., n_lliu - 1]. El millor lliurament d'un estudiant és el que
  supera més jocs de proves i, en cas d'empat, el que té el temps de
  lliurament més gran. La matriu m no conté més d'un lliurament amb el
  mateix DNI. A més, els lliuraments de m estan organitzats de la manera
  següent: 1) cada fila k només conté lliuraments que superen exactament
  k jocs de proves; 2) dins d'una fila concreta, els lliuraments estan
  ordenats en ordre creixent per número de DNI. */
```

Heu de lliurar un fitxer `solucio.cc` amb una implementació eficient del mètode `millor` de la classe *Lliurament* i de l'acció `classifica` del mòdul funcional *Eines_Vec_Lliu*. Trobareu la plantilla del fitxer `solucio.cc` dins del material addicional de què us proveïm a l'apartat *Public files* del problema del Jutge. Aquesta plantilla es troba en el fitxer `plantilla.txt`: n'heu de canviar el nom de manera que es digui `solucio.cc`, completar-lo i lliurar-lo al Jutge.
El vostre fitxer `solucio.cc` no pot contenir la implementació d'altres operacions de la classe `Lliurament` ni del mòdul funcional `Eines_Vec_Lliu`.

Observació

A l'apartat *Public files* del Jutge us proveïm de material addicional en un fitxer .tar. Podeu extreure el contingut d'aquest fitxer amb la instrucció

```
tar -xvf nom_fitxer.tar
```

Aquest material addicional consisteix en els fitxers següents:

- `plantilla.txt`: és la plantilla del fitxer solucio.cc; heu de canviar el nom d'aquest fitxer de manera que es digui `solucio.cc`, completar-lo i lliurar-lo al Jutge
- `Lliurament.hh`: l'especificació i la representació del tipus de la classe `Lliurament`
- `Lliurament.cc`: la implementació dels mètodes de la classe `Lliurament`, tret de la del mètode estàtic i públic `millor`, que heu de completar al fitxer `solucio.cc`
- `Eines_Vec_Lliu.hh`: l'especificació Pre/Post de totes les operacions del mòdul funcional `Eines_Vec_Lliu`
- `Eines_Vec_Lliu.cc`: la implementació de totes les operacions del mòdul funcional `Eines_Vec_Lliu`, tret de la de l'acció `classifica`, que heu de completar al fitxer `solucio.cc`
- `pro2.cc`: un programa principal que podeu fer servir per provar els mètodes públics de la classe `Lliurament` i les operacions del mòdul funcional `Eines_Vec_Lliu`
- `llegeixme.txt`: instruccions per a generar l'executable del programa `pro2` i provar-lo

Valorarem positivament que la solució no contingui instruccions innecessàries (especialment bucles o crides a operacions costoses), ni objectes innecessaris (especialment vectors o matrius), que no faci recorreguts quan hauria de fer cerques i que usi correctament les operacions més adients de la classe `Lliurament` i del mòdul funcional `Eines_Vec_Lliu` sempre que sigui possible. No es pot emprar cap estructura de dades que no hagi aparegut a les sessions 1 a 4 de laboratori. Es permet un ús justificat de l'operació `push_back` de la classe `vector`.

Quan feu els enviaments, el Jutge us indicarà quants jocs de proves passeu i de quin tipus (públic o privat). El joc de proves anomenat `públic` correspon als fitxers `entrada.txt` i `sortida_correcta.txt` de l'apartat *Public files*.

Exemple d'entrada

```
13
100 1 1 0 0 2500
102 1 0 0 0 1600
102 1 1 0 0 2400
102 1 1 1 0 4000
105 1 1 1 1 3000
105 1 1 1 0 4000
106 1 1 1 1 3000
120 1 0 1 0 2000
120 1 1 1 0 4000
132 1 1 1 0 2500
132 1 1 1 0 3000
130 1 1 1 1 2600
130 1 1 1 1 2700
135 1 0 0 0 1800
135 1 0 1 0 2700
100 1 0 0 0 2600
135 1 1 1 0 3600
135 1 0 1 0 4000
136 1 0 1 0 3000
145 1 0 1 0 4000
145 1 1 1 1 4500
152 1 0 1 0 2000
152 1 1 1 0 3000
152 1 1 0 0 4000
156 1 0 0 0 1000
```

```
156 1 1 0 0 2000
156 1 1 1 0 3000
156 1 1 1 1 4000
158 1 0 0 0 2000
0
classificar
consultar 156
consultar 103
fi
```

Exemple de sortida

```
29 LLIURAMENTS REBUTS

DNI 100: [1,1,0,0];  temps: 2500
DNI 100: [1,0,0,0];  temps: 2600
DNI 102: [1,0,0,0];  temps: 1600
DNI 102: [1,1,0,0];  temps: 2400
DNI 102: [1,1,1,0];  temps: 4000
DNI 105: [1,1,1,1];  temps: 3000
DNI 105: [1,1,1,0];  temps: 4000
DNI 106: [1,1,1,1];  temps: 3000
DNI 120: [1,0,1,0];  temps: 2000
DNI 120: [1,1,1,0];  temps: 4000
DNI 130: [1,1,1,1];  temps: 2600
DNI 130: [1,1,1,1];  temps: 2700
DNI 132: [1,1,1,0];  temps: 2500
DNI 132: [1,1,1,0];  temps: 3000
DNI 135: [1,0,0,0];  temps: 1800
DNI 135: [1,0,1,0];  temps: 2700
DNI 135: [1,1,1,0];  temps: 3600
DNI 135: [1,0,1,0];  temps: 4000
DNI 136: [1,0,1,0];  temps: 3000
DNI 145: [1,0,1,0];  temps: 4000
DNI 145: [1,1,1,1];  temps: 4500
DNI 152: [1,0,1,0];  temps: 2000
DNI 152: [1,1,1,0];  temps: 3000
DNI 152: [1,1,0,0];  temps: 4000
DNI 156: [1,0,0,0];  temps: 1000
DNI 156: [1,1,0,0];  temps: 2000
DNI 156: [1,1,1,0];  temps: 3000
DNI 156: [1,1,1,1];  temps: 4000
DNI 158: [1,0,0,0];  temps: 2000

CLASSIFICACIO

0 jocs de proves superats

1 jocs de proves superats
DNI 158: [1,0,0,0];  temps: 2000

2 jocs de proves superats
DNI 100: [1,1,0,0];  temps: 2500
DNI 136: [1,0,1,0];  temps: 3000

3 jocs de proves superats
DNI 102: [1,1,1,0];  temps: 4000
DNI 120: [1,1,1,0];  temps: 4000
DNI 132: [1,1,1,0];  temps: 3000
DNI 135: [1,1,1,0];  temps: 3600
DNI 152: [1,1,1,0];  temps: 3000

4 jocs de proves superats
DNI 105: [1,1,1,1];  temps: 3000
DNI 106: [1,1,1,1];  temps: 3000
DNI 130: [1,1,1,1];  temps: 2700
DNI 145: [1,1,1,1];  temps: 4500
DNI 156: [1,1,1,1];  temps: 4000

LLIURAMENTS DE 156
DNI 156: [1,0,0,0];  temps: 1000
```

```
DNI 156: [1,1,0,0];  temps: 2000
DNI 156: [1,1,1,0];  temps: 3000
DNI 156: [1,1,1,1];  temps: 4000
```

```
NO HI HA LLIURAMENTS DE 103
```

Informació del problema

Autor : Professors de PRO2
Generació : 2017-10-26 00:20:02