Control PRO2 - Torn 2 (Primavera 2017) X38181_ca

Hem decidit estendre la classe `Cjt_estudiants` que heu vist al laboratori amb una nova funcionalitat que selecciona automàticament els estudiants que poden fer el curs de reavaluació (és a dir, que estan admesos) tenint en compte que hi ha un nombre limitat de places disponibles `MAX_PLA` per aquest curs. Un estudiant *pot optar al curs de reavaluació* si té nota i la seva nota és major o igual a 4 i menor que 5. Les places del curs de reavaluació s'assignen per ordre descendent de nota entre els estudiants que poden optar al curs de reavaluació, i en cas d'empat, per ordre descendent de DNI. Això significa que, donats dos estudiants e_1 i e_2 que poden optar al curs de reavaluació, e_1 tindria preferència sobre e_2 per a ser admès al curs de reavaluació si e_1 té millor nota que e_2, o si e_1 i e_2 tenen la mateixa nota i el DNI d'e_1 és més gran que el DNI d'e_2.
Concretament, hem afegit dos mètodes públics a la classe `Cjt_estudiants`: 1) `n_admesos`, que retorna el nombre d'estudiants del conjunt que estan admesos al curs de reavaluació; 2) `pos_min_admes`, que retorna la posició de l'estudiant admès al curs de reavaluació amb nota més petita, i en cas d'empat, amb nota i DNI més petits. Si cap estudiant del conjunt està admès al curs de reavaluació, `pos_min_admes` retorna -1.
En tot moment el nombre d'estudiants admesos `n_admes` serà menor o igual al nombre de places disponibles `MAX_PLA`. A més `n_admes` serà menor o igual al nombre d'estudiants que compleixen les condicions per optar al curs de reavaluació. Finalment, si el nombre d'estudiants que poden optar al curs de reavaluació és major o igual al nombre de places disponibles `MAX_PLA`, el nombre d'estudiants admesos `n_admes` serà igual a `MAX_PLA`.
Per implementar eficientment aquesta funcionalitat hem modificat la representació i l'invariant de la classe `Estudiant` de la manera descrita a l'arxiu `Estudiant.hh`. En particular, representem la informació sobre si un estudiant està admès al curs de reavaluació o no als objectes de la classe `Estudiant`, i no als objectes de classe `Cjt_estudiants`. Això vol dir que per admetre al curs de reavaluació a l'estudiant situat a la posició `pos` de `vest` hem de utilitzar la instrucció `vest[pos].modificar_reaval(true);` i semblantment per no admetre'l `vest[pos].modificar_reaval(false);`, comprovar si està admès `vest[pos].admes_reaval()` o si compleix les condicions per optar al curs de reavaluació `vest[pos].cond_reaval()`. Llegiu amb cura l'arxiu `Estudiant.hh`, especialment les descripcions dels nous atributs, l'invariant i les especificacions de les operacions noves. Les principals novetats de la classe `Estudiant` són:

- hem incorporat un nou atribut `reaval` que permet representar informació sobre si l'estudiant paràmetre implícit està admès al curs de reavaluació o no;
- hem afegit el consultor `admes_reaval` que permet consultar si l'estudiant paràmetre implícit està admès al curs de reavaluació o no;
- hem afegit el modificador `modificar_reaval` que permet modificar la informació sobre si l'estudiant paràmetre implícit està admès al curs de reavaluació o no;
- hem afegit el consultor `cond_reaval` que permet comprovar si l'estudiant paràmetre implícit compleix les condicions per optar al curs de reavaluació;
- hem afegit el mètode `static` i públic `major_nota_dni` que permet comparar dos estudiants per nota i en cas d'empat per DNI;

També hem modificat la representació i l'invariant de la classe `Cjt_estudiants` de la manera descrita a l'arxiu `Cjt_estudiants.hh`. La principal novetat és que emmagatzemem la posició de l'estudiant admès al curs de reavaluació amb nota més petita, i en cas d'empat, amb nota i DNI més petits, en l'atribut `i_min_admes`. D'aquesta manera, quan afegim un estudiant a un conjunt on el nombre d'estudiants admesos al curs de reavaluació és igual al nombre de places disponibles, podrem determinar fàcilment si hem d'admetre el nou estudiant al curs de reavaluació o no. Es a dir, podrem comprovar si el nou estudiant compleix les condicions per optar al curs de reavaluació i si té millor nota o igual nota i DNI més gran que l'estudiant admès al curs de reavaluació amb nota més petita, o amb nota i DNI més petits, del conjunt original. Llegiu amb cura l'arxiu `Cjt_estudiants.hh`, especialment les descripcions dels nous atributs, l'invariant i les especificacions de les operacions noves. Les principals novetats de la classe `Cjt_estudiants.hh` són:

- hem incorporat un nou atribut `static` i constant `MAX_PLA` que especifica el nombre de places disponibles pel curs de reavaluació;
- hem incorporat un nou atribut `n_admes` que conté el nombre d'estudiants del conjunt admesos al curs de reavaluació;
- hem afegit un consultor `n_admesos` que permet consultar el valor de l'atribut `n_admes`, és a dir, el nombre d'estudiants admesos;
- hem incorporat un nou atribut `i_min_admes` que conté la posició a `vest` de l'estudiant admès al curs de reavaluació amb menor preferència. Si hi ha estudiants admesos al curs de reavaluació, l'atribut `i_min_admes` conté la posició de l'estudiant admès amb nota més petita, i en cas d'empat, amb nota i DNI més petits, i el seu valor està dins de l'interval $0 \leq$ `i_min_admes` $<$ `nest`; en cas contrari, si no hi ha estudiants admesos al curs de reavaluació, llavors `i_min_admes` és igual a -1.
- hem afegit el consultor públic `pos_min_admes`, per consultar la posició de l'estudiant admès al curs de reavaluació amb nota més petita i, en cas d'empat, amb nota i DNI més petits. Si hi ha estudiants admesos al curs de reavaluació al conjunt retorna `i_min_admes+1`, i si no hi ha estudiants admesos al conjunt retorna -1.
- i hem afegit el modificador privat `recalcular_pos_min_admes` per recalcular el valor de l'atribut `i_min_admes` quan sigui necessari.

Tenint això en compte heu d'implementar eficientment el següent mètode privat sense utilitzar l'operació `sort` de la biblioteca <algorithm>:

```
void recalcular_pos_min_admes();
/* Pre: cert */
/* Post: Si hi ha estudiants admesos al curs de reavaluació al conjunt
paràmetre implícit, llavors l'atribut i_min_admes conté la posició a
vest[0...nest-1] de l'estudiant admès amb nota més petita, i en cas
d'empat de l'estudiant admès amb nota i DNI més petits; en cas contrari,
si no hi ha estudiants admesos al curs de reavaluació al p.i., l'atribut
i_min_admes és igual a -1. */
```

i el següent mètode públic sense utilitzar l'operació `sort` de la biblioteca <algorithm>. Noteu que quan afegim al conjunt un estudiant que compleix les condicions per optar al curs de reavaluació és possible que hi hagi places disponibles per al curs de reavaluació no assignades a altres estudiants, i en aquest cas hem d'admetre'l. Però també és possible que totes les places disponibles estiguin assignades a altres estudiants, i en aquest cas hem de determinar si el nou estudiant té preferència sobre el estudiant admès amb nota més petita,

o amb nota i DNI més petits, del conjunt original. Perquè si no hi ha places disponibles per ambdós, l'estudiant que tingui major preferència ha de passar a ser admès i l'estudiant amb menor preferència ha de passar a no ser admès.

```
void afegir_estudiant(const Estudiant& est, bool& trobat);
/* Pre: El nombre d'estudiants del paràmetre implícit es més petit
que la mida màxima permesa. */
/* Post: Si el p.i. original no contenia cap estudiant amb el DNI
d'est, trobat és false, s'ha afegit l'estudiant est al p.i., s'han
actualitzat els estudiants admesos al curs de reavaluació al p.i.
si ha estat necessari, i s'ha actualitzat la posició de l'estudiant
admès amb nota més petita, i en cas d'empat amb nota i DNI més petits,
si ha estat necessari. En cas contrari, trobat és true i el p.i. és
igual a l'original. */
```

Observació

Heu de lliurar un fitxer `solucio.cc` amb una implementació eficient de les operacions `recalcular_pos_min_admes` i `afegir_estudiant` que ha de tenir el següent format:

```
#include "Cjt_estudiants.hh"

void Cjt_estudiants::recalcular_pos_min_admes() {
  ... // codi de la implementació
}

void Cjt_estudiants::afegir_estudiant(const Estudiant& est, bool& trobat) {
  ... // codi de la implementació
}
```

Copieu aquesta plantilla en el vostre `solucio.cc` i completeu-la. El vostre `solucio.cc` no pot contenir la implementació d'altres operacions de la classe.
A l'apartat *Public files* del Jutge us proveïm amb material addicional comprimit en un fitxer .tar. Podeu descomprimir aquest fitxer amb la comanda

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

Aquest material addicional consisteix en els següents fitxers:

- `Cjt_estudiants.hh`: l'especificació Pre/Post de totes les operacions públiques i privades d'aquesta nova versió de la classe `Cjt_estudiants`, així como la definició dels camps privats. Fixeu-vos que **hem afegit tres atributs** `n_admes, i_min_admes` i `MAX_PLA`, i que **hem modificat l'invariant** de la representació de `Cjt_estudiants`. **És molt important que la implementació de les operacions que us hem encarregat tingui en compte i preservi l'invariant de la representació**. Fixeu-vos també que hi ha operacions noves: el modificador privat `recalcular_pos_min_admes`, els consultors pùblics `n_admesos, places_disp, pos_min_admes`, i el modificador públic `afegir_estudiant`.

- `Cjt_estudiants.cc`: la implementació de totes les operacions de la nova versió de la classe `Cjt_estudiants` tret de les operacions que us demanem.

- `Estudiant.hh`: l'especificació de la classe `Estudiant` i la definició dels seus atributs. Les principals novetats que presenta són un atribut `reaval` que indica si l'estudiant paràmetre implícit està admès al curs de reavaluació, els consultors públics `admes_reaval, cond_reaval` i `major_nota_dni`, i el modificador públic `modificar_reaval`.

- `Estudiant.cc`: la implementació dels mètodes de la classe `Estudiant`.
- `pro2.cc`: un programa principal que podeu fer servir per provar els mètodes públics d'aquesta versió de la classe `Cjt_estudiants`.
- `llegeixme.txt`: instruccions per a generar l'executable del programa `pro2` i provar-lo.

Valorarem positivament que la solució no contingui instruccions (especialment bucles o crides a operacions costoses) ni objectes (especialment vectors o conjunts) innecessaris, que no faci recorreguts quan hauria de fer cerques, i que usi correctament les operacions més eficients de la classe sempre que sigui possible. No es pot emprar cap estructura de dades que no hagi aparegut a les sessions 1-4 de laboratori.

La utilització de l'operació `sort` de la biblioteca <`algorithm`> a l'arxiu `solucio.cc` comportarà una qualificació de 0 a la correcció manual del control.

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*.

Informació del problema

Autoria: Professors de PRO2

Generació: 2026-01-25T15:46:04.327Z