Control PRO2 - Turno 1 (Primavera 2018) X37831_es

Hemos decidido extender la clase `Cjt_estudiants` que habéis visto en el laboratorio con dos nuevas funcionalidades.
La primera funcionalidad calcula la nota global de un estudiante en una asignatura a partir de sus notas en los exámenes parciales de dicha asignatura, y asigna la nota global calculada a dicho estudiante. La nota global de un estudiante puede ser -1, si se considera que la calificación global del estudiante en la asignatura debe ser NP (No Presentado), o un valor real en el rango `[0 ...Estudiant::nota_maxima()]`, que corresponde a la suma ponderada de las notas del estudiante en los exámenes parciales de la asignatura. Para realizar la evaluación global de los estudiantes de una asignatura, la clase `Cjt_estudiants` necesita representar cierta información sobre dicha asignatura, en particular, el número de exámenes parciales que se realizan en la asignatura y el peso de cada examen parcial en la nota global de la asignatura. Por este motivo, hemos añadido dos atributos nuevos a la clase `Cjt_estudiants`: (1) `nombre_parcials` de tipo entero, que representa el número de exámenes parciales; y (2) `pesos_parcials` de tipo `vector<double>`, que permite almacenar los pesos de los distintos exámenes parciales en la nota global. En particular, el peso del examen parcial j-ésimo es `pesos_parcials[j-1]` para todo j tal que $1 \leq j \leq$ nombre_parcials.
Concretamente, la evaluación global de un estudiante se realiza de la siguiente manera. Si la suma de los pesos de los exámenes parciales a los que no se ha presentado un estudiante es mayor que 0.5, la nota global del estudiante debe ser -1 (que es el valor elegido para representar la calificación "No Presentado" en nuestra implementación del tipo `Cjt_estudiants`).
En otro caso, la nota global del estudiante debe ser la suma ponderada de sus notas en los exámenes parciales de la asignatura, que es igual a la suma de las contribuciones a la nota global de los exámenes parciales a los que el estudiante se ha presentado, donde la contribución del examen parcial j-ésimo se obtiene multiplicando la nota del estudiante en el examen parcial j-ésimo por el peso de dicho examen parcial.
Esta funcionalidad se implementa en la clase `Cjt_estudiants` mediante el método privado

```
void avaluacio_global_iesim(int i);
/* Pre:  1 <= i <= mida()
   El estudiante i-ésimo tiene notas asignadas para todos los exámenes
   parciales de la asignatura. La nota de cada examen parcial puede ser
   -1, si el estudiante no se ha presentado a dicho examen parcial, o
   una nota válida en el rango [0...Estudiant::nota_maxima()]. */
/* Post: El estudiante i-ésimo pasa a tener asignada su nota global en
   la asignatura, que puede ser -1, si se considera que su calificación
   global debe ser NP (No Presentado), o una nota válida en el rango
   [0...Estudiant::nota_maxima()] que corresponde a la suma ponderada
   de sus notas en los exámenes parciales de la asignatura a los que
   se ha presentado. */
```

Por ejemplo, si c es un objeto de la clase `Cjt_estudiants` que representa los estudiantes de una asignatura con 5 exámenes parciales cuyos pesos son `[0.1, 0.15, 0.25, 0.25, 0.25]` respectivamente, y las notas en los exámenes parciales del tercer estudiante de c son `{6.5, -1, 10, 6, 4}`, después de la llamada `c.avaluacio_global_iesim(3)`, la nota global del tercer estudiante de c debe ser 5.65. Del mismo modo, si las notas en los exámenes parciales del quinto estudiante de c son `{-1, 10, -1, 5, -1}`, después de la

llamada `c.avaluacio_global_iesim(5)`, la nota global del quinto estudiante de c debe ser -1 (i.e. No Presentado).
La segunda funcionalidad añadida a la clase `Cjt_estudiants` calcula y escribe en el canal estándar de salida el subconjunto de exámenes parciales a los que se han presentado todos los estudiantes que han aprobado la asignatura, es decir, todos los estudiantes cuya nota global en la asignatura es mayor o igual que 5. Esta funcionalidad se implementa en la clase `Cjt_estudiants` mediante el método público

```
void parcials_presentats_aprovats() const;
/* Pre: Todos los estudiantes del parámetro implícito tienen notas
   asignadas para todos los exámenes parciales y también tienen
   asignada su nota global en la asignatura. La nota de cada examen
   parcial puede ser -1, si el estudiante no se ha presentado a dicho
   examen parcial, o una nota válida en el rango
   [0...Estudiant::nota_maxima()]. */
/* Post: En el canal de salida estándar se han escrito los identificadores
   de los exámenes parciales a los que se han presentado todos los
   estudiantes que tienen una nota global en la asignatura mayor o igual
   que 5. Los identificadores de estos exámenes parciales están ordendados
   crecientemente.  */
```

Por ejemplo, si c es un objeto de la clase `Cjt_estudiants` que representa los estudiantes de una asignatura con 4 exámenes parciales en el que hay exactamente 10 estudiantes $\{e_{i_1}, \ldots, e_{i_{10}}\}$ con nota global mayor o igual que 5, y sabemos que de esos 10 estudiantes e_{i_4} no se ha presentado al primer examen parcial, y e_{i_1} y e_{i_6} no se han presentado al tercer examen parcial, después de la llamada `c.parcials_presentats_aprovats()` se escribirá en la pantalla el subconjunto $\{2, 4\}$, ya que los únicos exámenes parciales a los que se han presentado todos los estudiantes aprobados son el segundo y el cuarto.
Para implementar estas funcionalidades hemos modificado también la representación de la clase `Estudiant` de la manera descrita en el archivo `Estudiant.hh`. En particular, representamos las notas de un estudiante en dos atributos nuevos: (1) `nota_global` de tipo `double`; y (2) `notes_parcials` de tipo `vector<double>`. Concretament, la nota del examen parcial j-ésimo del estudiante parámetro implícito es `notes_parcials[j-1]` para todo j en el rango $1 \leq j \leq$ notes_parcials.size()

Observación

Teniendo esto en cuenta debéis implementar eficientemente el método privado `avaluacio_global_iesim` y el método público `parcials_presentats_aprovats`. Debéis entregar un archivo `solucio.cc` con una implementación eficiente de estos dos métodos.
En el caso del método público `parcials_presentats_aprovats` podéis completar la implementación parcial que os proponemos, o implementar este método sin utilizar la implementación parcial propuesta.
Es posible superar algunos juegos de prueba (pero no todos) **implementando únicamente el método privado** `avaluacio_global_iesim`, siempre y cuando no modifiquéis la implementación parcial del método público `parcials_presentats_aprovats` que os proporcionamos.
Dentro del material adicional que os proporcionamos en el apartado *Public files* del problema del jutge encontraréis el archivo `plantilla.txt` con las cabeceras de ambos métodos y una implementación incompleta del método `parcials_presentats_aprovats`: debéis renombrar el archivo `plantilla.txt` de manera que se llame `solucio.cc`, completarlo y enviarlo al jutge.

Vuestro archivo `solucio.cc` no puede contener la implementación de otros métodos de las clases `Estudiant` o `Cjt_estudiants`.
En el apartado *Public files* del jutge os proporcionamos material adicional en un fichero .tar. Podéis extraer el contenido de este fichero con la instrucción

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

Este material adicional contiene los siguientes archivos:

- `plantilla.txt`: es la plantilla del archivo solucio.cc; debéis renombrar este archivo de manera que se llame `solucio.cc`, completarlo y enviarlo al jutge
- `Cjt_estudiants.hh`: la especificación Pre/Post de todos los métodos públicos y privados de esta nueva versión de la clase `Cjt_estudiants`, así como la definición de los atributos privados.
- `Cjt_estudiants.cc`: la implementación de todos los métodos de la clase `Cjt_estudiants`, excepto la de los métodos que os pedimos.
- `Estudiant.hh`: la especificación de la nueva versión de la clase `Estudiant` y la definición de sus atributos privados.
- `Estudiant.cc`: la implementación de todos los métodos de la clase `Estudiant`.
- `pro2.cc`: un programa principal que podéis utilizar para probar los métodos públicos de esta nueva versión de la clase `Cjt_estudiants`.
- `entrada.txt` y `sortida_correcta.txt`: archivo de entrada del juego de pruebas público, y salida correcta para dicho juego.
- `llegeixme.txt`: instrucciones para generar el ejecutable del programa `pro2.cc` y probarlo.

Valoraremos positivamente que la solución no contenga instrucciones innecesarias (especialmente bucles o llamadas a operaciones costosas), ni objetos (especialmente vectores o matrices) innecesarios, que no haga recorridos cuando debería hacer búsquedas, y que use correctamente las operaciones más adecuadas de las clases `Estudiant` y `Cjt_estudiants` siempre que sea posible. No se puede usar ninguna estructura de datos que no haya aparecido en las sesiones 1 a 4 de laboratorio.

Cuando hagáis envíos, el jutge os indicará cuantos juegos de pruebas supera vuestro programa y de qué tipo (público o privado). **Tened en cuenta que es posible superar algunos juegos de prueba (pero no todos) implementando únicamente el método** `avaluacio_global_iesim`**, si no modificáis la implementación parcial del método** `parcials_presentats_aprovats` **que os proporcionamos**.

Información del problema

Autor : Professors de PRO2
Traductor : Professors de PRO2
Generación : 2018-03-18 00:37:50