

---

**Gestión integral de un conjunto de estudiantes****X22862\_es**

---

Escribe una miniaplicación que lea un conjunto de estudiantes y ofrezca la posibilidad de aplicarle los siguientes tratamientos.

- añadir un nuevo estudiante (opción -1); si no es posible, escribir un mensaje informando del motivo
- consultar la nota de un estudiante a partir de su DNI (opción -2); si no es posible, escribir un mensaje informando del motivo
- modificar la nota de un estudiante (opción -3); si no es posible, escribir un mensaje informando del motivo
- redondear la nota, a la décima más cercana, a todos los estudiantes del conjunto (opción -4)
- escribir el conjunto de estudiantes (opción -5)

Organiza el programa principal como un proceso iterativo que, tras leer el conjunto, lea un valor entre -1 y -6 indicativo de la opción que queremos aplicar (la opción -6 será la de terminar el programa), después lea los datos necesarios para ella y proceda a aplicarla.

Observa que las opciones -1, -2, -3 y -5 corresponden a operaciones que ya existen en la clase `Cjt_estudiants`, por lo que solo necesitarás comprobar la precondition de las mismas antes de realizar las correspondientes llamadas. En caso de no cumplirse la precondition no debe realizarse la llamada, simplemente ha de escribirse un mensaje informando de la situación. En algunos casos necesitarás también operaciones de la clase `Estudiant`.

La opción -4 requiere programar una operación nueva en el propio fichero `solution.cc` (o como se llame tu solución). Dicha operación estará basada en operaciones de las clases `Estudiant` y `Cjt_estudiants`. Para evitar problemas de precisión con el tipo `double` al realizar los redondeos, copia y usa el siguiente código en tu solución

```
double redondear(double r)
{
    return int(10.*(r + 0.05)) / 10.0;
}
```

**Entrada**

Una secuencia de instrucciones, definidas mediante los valores de -1 a -6, cada una con datos en el formato adecuado, que se puede deducir del juego de pruebas público.

**Salida**

Los resultados o los mensajes producidos por las instrucciones de la entrada. El formato exacto se puede deducir del juego de pruebas público.

## Observación

Solo has de entregar el programa principal, que deberá contener los `#include` necesarios para poder usar las clases `Estudiant` y `Cjt_estudiants`. **Importante:** usa las versiones de dichas clases que te proporcionamos con los “public files” (icono del gatito).

Como siempre, valida tu solución con el juego de pruebas público y prueba otros casos interesantes antes de subirla al Jutge, como por ejemplo que el conjunto esté vacío.

### Ejemplo de entrada 1

```
3
444 4.44
222 2.22
111 323

-5

-2 111

-3 111 1.11

-5

-2 111

-2 555

-1 111 4

-3 555 5

-5

-4

-5

-1 777 777

-5

-1 333 3.22

-3 333 3.33

-4

-5

-1 555 11
-1 222 11

-5

-6
```

### Ejemplo de salida 1

```
111 NP
222 2.22
444 4.44

el estudiante 111 no tiene nota

111 1.11
222 2.22
444 4.44

el estudiante 111 tiene nota 1.11

el estudiante 555 no esta

el estudiante 111 ya estaba

el estudiante 555 no esta

111 1.11
222 2.22
444 4.44

111 1.1
222 2.2
444 4.4

111 1.1
222 2.2
444 4.4
777 NP

111 1.1
222 2.2
333 3.3
444 4.4
777 NP

el estudiante 222 ya estaba

111 1.1
222 2.2
333 3.3
444 4.4
555 NP
777 NP
```

## **Información del problema**

Autoría: profesores de PRO2

Generación: 2026-01-25T14:25:50.121Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>