|  | **Técnico Universitario en Programación**  **Laboratorio de Computación II** |
| --- | --- |

Segundo Parcial

Duración aproximada: 3 horas (recuerde participar del zoom por consultas)

Puntos para Aprobar: 60

**Problema 1 - (80 puntos)**

Haciendo uso del archivo SegundoParcial.zip que se encuentra en el campus virtual, realizar una clase llamada SegundoParcial con los métodos que resuelvan los siguientes puntos:

| Para la resolución se pueden hacer uso de todas las funciones o métodos adicionales o atributos que consideres necesarios. Pero todo el examen debe resolverse dentro de la clase SegundoParcial. |
| --- |

1- Crear un método llamado **alumnosDesaprobados** que muestre el curso cuya cantidad de alumnos que desaprobaron el examen final sean la más alta, en caso de haber más de uno, mostrar todos los cursos que cumplan dicha condición.

**(20- Puntos)**

2- Crear un método llamado **peorCurso** que muestre los cursos donde todos sus alumnos desaprobaron.

**(20- Puntos)**

3- Crear un método llamado **verGeneraciones** que muestre para cada curso la generación en la que más alumnos tenga inscriptos. Se clasifican según el año de nacimiento de la siguiente manera:

* Generación silenciosa <= 1945
* Baby boomers 1946 - 1964
* Generación X 1965 - 1980
* Millennials 1981 - 1995
* Generación [Z](https://elvisitantedigital.com/caracteristicas-generacionz-adolescentes/) 1996 - 2012
* Generación Alpha >= 2013

**(20- Puntos)**

4- Crear un método llamado **generarReporte** que genere un archivo llamado reportes.dat donde se guarde para cada curso el porcentaje de aprobados y desaprobados de todos los cursos que se hayan terminado.

Un curso está terminado cuando todos sus alumnos realizaron el examen final.

Crear un método llamado  **mostrarReporte** que muestre el reporte leyéndolo del archivo de reportes.dat

**(20- Puntos)**

**ACLARACIÓN:**

Se aprueba con una nota de examen final mayor o igual a 6

Si la nota tiene un cero, es que aun no realizó el examen final.

Solo existen 10 cursos, y los ids van del 1 al 10

**Problema 2 - (20 puntos)**

Hacer una clase llamada CuentaBancaria que registre un ID y un saldo.

Luego, crear una clase llamada CajaAhorro que herede de CuentaBancaria y tenga los siguientes métodos:

* Hacer un constructor que reciba el ID como parámetro y establezca el saldo en 0.
* Debitar(float importe): Reste el importe al saldo si el importe es positivo. El saldo resultante no puede ser menor a 0.
* Acreditar(float importe): Incrementa el importe al saldo si el importe es positivo.
* getSaldo(): Devuelva el saldo de la cuenta.

**(10 Puntos)**

Además, crear una clase llamada CuentaCorriente que herede de CuentaBancaria y agregue una propiedad llamada límite. Debe tener los siguientes métodos:

* Hacer un constructor que reciba el ID y el límite como parámetros y establezca el saldo en 0.
* Debitar(float importe): Reste el importe al saldo si el importe es positivo. El saldo resultante no puede ser menor al límite.
* Acreditar(float importe): Incrementa el importe al saldo si el importe es positivo.
* getSaldo(): Devuelva el saldo de la cuenta.

**(10 Puntos)**