PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

LENGUAJE DE PROGRAMACIÓN 1

2da práctica (tipo b)
(Primer Semestre 2020)

Indicaciones Generales:

- Duración: 110 minutos.

Obligatoriamente los alumnos deberán mantener en todo momento el audio de sus computadores abierto de modo que puedan recibir los comunicados que se hagan durante el examen. De tener algún problema deberán hacérselo saber de inmediato al profesor de su horario por correo. De no hacerlo, no se aceptarán reclamos alegando que no oyeron las indicaciones.

- No se pueden emplear **variables globales, estructuras, ni la clase string.** Tampoco se podrán emplear las funciones malloc, realloc, strdup o strtok, igualmente no se puede emplear cualquier función contenida en las bibliotecas stdio.h, cstdio o similares y que puedan estar también definidas en otras bibliotecas.
- Deberá modular correctamente el proyecto en archivos independientes. LAS SOLUCIONES DEBERÁN DESARROLLARSE BAJO UN ESTRICTO DISEÑO DESCENDENTE. Cada módulo **NO** debe sobrepasar las 20 líneas de código aproximadamente. El archivo **main.cpp** solo podrá contener la función **main** de cada proyecto y el código contenido en él solo podrá estar conformado por tareas implementadas como funciones. En cada archivo que implemente en los proyectos (.h y .cpp) deberá colocar un comentario en el que coloque claramente su nombre y código, de no hacerlo se le **descontará 0.5 puntos por archivo**.
- NO SE CALIFICARÁN AQUELLAS FUNCIONES DESARROLLADAS EN EL MISMO ARCHIVO QUE LA FUNCIÓN main.
- El código comentado NO SE CALIFICARÁ.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no den resultados coherentes en base al 60%.
  Se tomará en cuenta en la calificación el uso de comentarios relevantes.

SE LES RECUERDA QUE, DE ACUERDO AL REGLAMENTO DISCIPLINARIO DE NUESTRA INSTITUCIÓN, CONSTITUYE UNA FALTA GRAVE COPIAR DEL TRABAJO REALIZADO POR OTRA PERSONA O COMETER PLAGIO. ESTO Y EL HECHO DE ENCONTRAR CUALQUIER ARCHIVO YA SEA .cpp O .h CON FECHA U HORA DE CREACIÓN ANTERIOR AL EXAMEN SERÁ CONSIDERADO UNA FALTA DE PROBIDAD Y POR LO TANTO AMERITARÁ LA ANULACIÓN DE LA PRUEBA.

NO SE HARÁN EXCEPCIONES ANTE CUALQUIER TRASGRESIÓN DE LAS INDICACIONES DADAS EN LA PRUEBA

Puntaje total: 20 puntos

Cree una carpeta con el nombre "LABORATORIO02" y en él desarrolle los proyectos que a continuación se solicitan.

Problema

Este laboratorio busca que usted desarrolle clases e implemente operadores sobrecargados, que le permita manejar las calificaciones obtenidas por un alumno en diferentes ciclos y cursos.

DEBERÁ LEER Y ENTENDER TODAS LAS TAREAS SOLICITADAS ANTES DE EMPEZAR A ESCRIBIR EL CÓDIGO DE PROYECTO PARA EVITAR REPETIR CÓDIGO INNECESARIAMENTE.

PARTE 1: Clase Alumno (15 puntos)

Se solicita que desarrolle un proyecto, denominado **LAB02_PREG01_ALUMNO**, en la cual se defina tres clases, la primera clase denominada "Curso" (en archivos **Curso.cpp** y **Curso.h**), la segunda "**Alumno**" (en archivos **Alumno.cpp** y **Alumno.h**), la tercera clase "**Acurso**" (en archivos **Acurso.cpp** y **Acurso.h**). Las clases deben contar con sus los atributos y métodos (datos y funciones miembro) necesarios que cumplan con los siguientes requerimientos:

a. Atributos (3.0 puntos):

(1) Clase Curso:
Contendrá la información de un curso y se manejara mediante clase denominada "**Curso**" que contenga los siguientes atributos:

- "**ccurso"** es una cadena que almacena el código del curso.
- "**ciclo"** es un entero que almacena el ciclo donde se llevó el curso.
- "**nota"** es un entero que almacena la nota final obtenida en el curso respectivo.
- "**credito"** es un número de punto flotante que representa los créditos del curso.

(2) Clase Alumno:
La información de alumnos y notas se manejarán mediante clase denominada "**Alumno**" que contenga los siguientes atributos:

- "**codigo**" tipo entero almacena el código del alumno.
- "**tipo**" es un carácter para servirá para guardar el tipo de alumno.
- "**nombre**" cadena de caracteres que guarda el nombre del alumno.
- "**lcurso"** es un arreglo de objetos tipo **Curso**. Aquí se guardaran las notas de cada curso llevado por el alumno. Considere el arreglo de un tamaño adecuado.
- "**numcur"** es un número entero que representa los cursos que ha llevado un alumno.

(3) Clase ACurso:
La nota de un curso en un ciclo determinado se manejará mediante una clase denominada "**ACurso**" que contenga los siguientes atributos:

- "**codigo**" es de tipo entero y sirve para almacenar el código del alumno.
- "**clcurso**" es un objeto tipo **Curso**.
- "**operacion**" es un carácter, que indica que transacción se debe realizar con el registro, si es "**N"** se debe añadir, si es "**A**" sirve para actualizar y si es "**E**" se debe eliminar.

Se deben inicializar las clases de tal forma que soporten el ingreso de los diferentes cursos llevados por los alumnos. De igual forma se deben desarrollar los constructores copia que ayuden a las diversas tareas.

b. Asignación (2 puntos):

- ✓ **curso01.asignar(curso02);** asigna el objeto curso01 desde un objeto tipo class **Curso** enviado como parámetro.
- ✓ **curso01 = curso02;** Sobrecargando **operador =** a partir de un objeto tipo class **Curso**.

c. Comparación (2 puntos):

Se podrá comparar dos objetos de la clase **Curso** de las formas siguientes:

- ✓ **curso01.compare(curso02);** → Compara el código del curso y ciclo del objeto ingresado como parámetro con la del objeto **curso01**. Devuelve **true** si son iguales y **false** si son diferentes.
- ✓ **curso01 == curso02** → Se comparan el código del curso y ciclo de dos objetos tipo class **Curso** mediante la sobrecarga del operador. Se devuelve **true** si se cumple la condición y **false** si no. Esta sobrecarga deben implementarse como método de la misma clase (no como una función externa).

d. Operadores Sobrecargados (6 puntos):

- ✓ **Agregación: Alumno + ACurso;** Sobrecargando el operador **+**. Agrega los valores del objeto de la clase ***ACurso*** al objeto de la clase ***Alumno,*** ingresando los datos obtenidos en un nuevo curso para el alumno. Además incrementa el valor de ***numcur*** en el objeto de la clase ***Alumno***.

- ✓ **Actualización: Alumno * ACurso;** Sobrecargando el operador *. Actualiza la nota de un curso en un ciclo determinado del objeto ***Alumno,*** con los valores contenidos en el objeto ***ACurso***. Si el curso y ciclo no se pueden ubicar simplemente no se realiza la acción.
- ✓ **Eliminación (Alumno - *ACurso;)*** Sobrecargando el operador -. Se eliminan los valores de un curso en un ciclo determinado en el objeto ***Alumno,*** de acuerdo al curso y ciclo que contenga el objeto ***ACurso***. Además se actualiza el valor de ***numcur*** en el objeto ***Alumno***. Si el curso y ciclo no se pueden ubicar simplemente, no se realiza la acción.
- ✓ **Promedio (Alumno / int;)** Sobrecargando el operador /. Calcula el promedio de notas para un ciclo determinado del objeto **Alumno.** El ciclo que se brinda es un número entero.
- ✓ **Suma de Créditos (--Alumno;)** Sobrecargando el operador --. Se calcula el número de créditos que ha llevado un alumno del contenido en el objeto **Alumno.**
- ✓ **Suma de Créditos aprobados (++Alumno;)** Sobrecargando el operador ++. Se calcula el número de créditos que ha aprobado un alumno del objeto **Alumno.**

e. <u>Otros métodos</u> (2 puntos):

- ✓ **Lectura: arch >>Alumno.** Sobrecargando operador >>. Permite leer, desde el archivo ***arch***, una fila y almacena los datos en ***Alumno***.
- ✓ **Lectura: arch >> ACurso.** Sobrecargando operador >>. Permite leer, desde el archivo ***arch***, una fila y almacena los datos en **ACurso**.
- ✓ **Impresión: arch << Alumno.** Sobrecargando operador <<. Permite imprimir, en el archivo ***arch***, los datos de **Alumno**, de acuerdo al formato solicitado.
- ✓ **Destructor:** Debe liberar los espacios de memoria dinámica asignados al objeto.

<u>(*) Pruebas individuales</u>:

Deberá elaborar un conjunto de pruebas (en el archivo **main.cpp** del proyecto) que permita probar de manera sencilla cada uno de las clases, los métodos y sobrecargas desarrolladas. Los métodos u operadores sobrecargados que no sean parte de esta prueba no serán calificados.

<u>PARTE 2: Prueba final</u> (5 puntos)

Se solicita que desarrolle un proyecto denominado **LAB02_PREG02_CARGA** en el cual se utilizaran las clases, desarrolladas en la pregunta anterior. Este proyecto solucionará el problema planteado a continuación:

Se cuenta con el primer archivo denominado Alumnos.txt que es similar al que se muestra a continuación:

```
R  20160658  Arca/Amezquita/Edric-Ronald  IngenieriaMecanica  FCI
R  20119778  Morales/Valverde/Ines-Martha  IngenieriaInformatica  FCI
I  5258      Gavidia/Mendoza/Ronald-Johnny IngenieriaElectronica  FEI
R  20150564  Auris/Zimic/Javier-Daniel  Matematicas  FCI
R  20080667  Lozada/Yino/Martha  Matematicas  FCI
I  5395      Alamo/Pairazaman/Miguel-Roberto Quimica  FEI
R  20160119  Paredes/Patino/Christian  IngenieriaInformatica  FCI
R  20109738  Caro/Polo/Sandro  IngenieriaMecanica  FCI
…
```

En el archivo aparecen los datos de cada alumno, el primer campo indica el tipo de alumno (R: Regular, I: Intercambio), su código que siempre es un número, el nombre del alumno concatenado, la especialidad concatenada y la facultad abreviada.

El segundo archivo denominado Cursos.txt es similar a la muestra que se observa a continuación:

```
20110165 FIS111 19 2018-0 3.5 N
20110165 MAT111 8 2017-1 4 N
4450 FIS208 7 2015-2 3.5 N
20127352 MAT218 15 2017-0 4 N
5395 FIS218 11 2015-2 3.5 N
20119778 MAT218 16 2016-1 4 N
20170596 MAT218 14 2015-1 4 N
4450 FIS203 18 2018-1 3.5 N
6965 MAT111 13 2017-2 4 N
20109738 MEC206 12 2017-1 4.5 N
…
```

En el archivo se registran los cursos que los alumnos han llevado, en los diferentes ciclos y la nota obtenida, por tal motivo en cada línea se registra el código del alumno, el código del curso, la nota, el ciclo, la cantidad de créditos del curso en el ciclo respectivo y la operación que se realiza con el registro en este caso siempre es N (nuevo registro).

El tercer archivo denominado Correcciones.txt es similar a la muestra que se observa a continuación:

```
20119778 MAT218 18 2016-1 4 A
20170596 MAT218 11 2015-1 4 A
1994 FIS208 0 2015-2 0 E
6702 FIS219 16 2018-2 3.5 A
6702 FIS220 0 2016-1 0 E
1346 FIS220 12 2017-2 3.5 A
6553 FIS111 11 2015-2 3.5 A
5258 MAT219 0 2017-2 0 E
…
```

En el archivo se registran las posibles correcciones en la nota de un curso en un determinado ciclo, de acuerdo al código del alumno. Al final de cada línea se aprecia la operación que se realiza con el registro, el cual puede ser "A" (se debe modificar la nota de la estructura principal) o "E" (se debe eliminar la nota y los datos correspondientes de la estructura principal).

Empleando las clases, métodos y sobrecargas, deberá realizar: leer los datos de cada línea del archivo Alumnos.txt y colocarlos en un **arreglo estático** de la clase **Alumno**. Luego debe leer cada línea del archivo Cursos.txt utilizando la clase **ACurso** y completar los datos de la clase **Alumno**. Con los datos cargados en el arreglo se deben realizar las actualizaciones registradas en el archivo Correcciones.txt empleando la estructura auxiliar. Con el arreglo principal cargado y utilizando las sobrecargas, genere un archivo consolidad por alumno, con el siguiente formato:

```
Nombre:
Código:
=====================================
       CICLO                  PROMEDIO
  1)   2012-1                    14.00
  2)   2013-0                    13.50

   …   …                           …
  6)   2018-1                    14.50

Créditos Cursados:  60.0
Créditos Aprobados: 45.5
```

El reporte debe estar perfectamente tabulado (sin usar el carácter '\t').

Anotaciones finales

Al finalizar la práctica, comprima[1] la carpeta Laboratorio02 y súbalo a Paideia.

Profesores del curso: Miguel Guanira
Rony Cueva

San Miguel, 26 de junio del 2020.

[1] Para evitar problemas en la corrección de la prueba, utilice el programa de compresión que viene por defecto en Windows (Zip).