PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

LENGUAJE DE PROGRAMACIÓN 1

8va práctica (tipo b)
(Primer Semestre 2019)

Indicaciones Generales:

- Duración: 110 minutos.
- Se podrá usar como material de consulta solo sus apuntes de clase.
- No se pueden emplear **variables globales, ni estructuras**. No se podrá emplear la **clase string** ni las funciones malloc, realloc, strdup, strtok, tampoco las funciones incluidas en las bibliotecas cstdio, stdio.h o similares.
- 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. 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**. Se tomará en cuenta en la calificación el uso de comentarios relevantes.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% del puntaje de la pregunta. Los que no den resultados coherentes en base al 60%.
- **La presentación, la ortografía y la gramática de los trabajos influirá en la calificación.**

Puntaje total: 20 puntos

Problema

Se desea elaborara una aplicación orientada a objetos que permita ayudar a una institución en sus procesos. Para realizar esta tarea, se le pide lo siguiente:

Pregunta 1 (8 puntos)

Elabore un proyecto denominado Pregunta01 que defina dos clases como se muestra a continuación:

```
class Curso {
   prívate:
      char *codigo;
      char *nombre;
      double creditos;
      char **requisites;
      int numRequisitos;
   public:
      …
}
```

```
class Alumno {
   prívate:
      int codigo;
      char *nombre;
      char **cursos;
      int numCurso;
   public:
      …
}
```

La clase Curso maneja las características de un curso de la institución, el atributo **requisitos** guardará los códigos de los cursos que se requieren aprobar para poder llevar el curso. La clase Alumno maneja las características de un alumno de la institución, el atributo **cursos** guardará los códigos de los cursos en los que está matriculado el alumno. Ambas clase tendrán su constructor por defecto y su destructor. También definirán métodos selectores en aquellos atributos que así lo requieran. Además ambas clases deben definir un método que permita agregar un código de curso a sus respectivos atributos, requisitos y cursos. En ambos casos los espacios de memoria deberán quedar exactos. La clase Curso deberá implementar un método que permita devolver el código de uno de sus requisitos, el método recibirá la posición en el arreglo del curso solicitado. La clase Alumno deberá implementar un método que devuelva verdadero si está matriculado en un curso cuyo código es entregado como dato. Podrá agregar todos los métodos que crea conveniente.

La aplicación deberá probar todos los métodos de las dos clases.

Pregunta 2 (12 puntos)

Elabore un proyecto denominado "**Pregunta02**", copie en él las clases (.h y .cpp) elaboradas en la pregunta 1 e incorpórelas al proyecto. El proyecto debe definir una clase denominada ListadeClases, la cual manejará las listas de clase de los diferentes cursos de la institución. Esta se describe a continuación:

```
class ListasDeClases {
   prívate:
      Curso *curso;
      int numCursos;
      Alumno *alumno;
      int numAlumnos;
   public:
      …
}
```

La clase definirá su constructor por defecto y su destructor. Además la clase debe definir los métodos: **cargarCursos**, **cargarAlumnos** y **consultarListasDeClases**, los tres recibirán como datos una cadena de caracteres con el nombre de un archivo. El primer método deberá poder leer los datos de un archivo similar al indicado como Archivo1 (ver más adelante) y colocarlos en el atributo **curso**. El segundo método deberá poder leer los datos de un archivo similar al indicado como Archivo2 y colocarlos en el atributo **alumno**. En ambos casos los espacios de memoria deberán ser exactos. El tercer método deberá permitir leer desde la entrada estándar de datos una serie de códigos de cursos, si el código se encuentra en la lista de cursos se deberá enviar a la salida estándar de datos un mensaje que indique que la tarea, que se describe a continuación, ha sido satisfactoria, sino se indicará con un mensaje de error. El fin de los datos se dará con el ingreso de la palabra "FIN".

La tarea a realizar consiste en imprimir en el archivo de textos la lista de clases de cada uno de los cursos que se ingresan, esta deberá ser similar a lo que se muestra a continuación:

```
                                        Listas de clases:
================================================================================================
 Clave: INF281     Curso: Lenguajes de programación 1          Créditos: 5.0     Requisitos: 1INF06,INF263
------------------------------------------------------------------------------------------------
                                        Alumnos matriculados
No.      Código          Nombre                                              Nota
------------------------------------------------------------------------------------------------
1.-      20090051        Fernández/Quispe/Mario-Luís-José                    ____
2.-      20103043        LI/Chong/Kevin                                      ____
3.- …
…
================================================================================================
                                        Número de alumnos: 29
------------------------------------------------------------------------------------------------
```

Los archivos, que son de tipo CSV, son similares a los siguientes:

```
Archivo1
MAT102,Matemáticas básicas 2, 3.5,MAT100,MAT101,FIS101
INF101,Sistemas de Información 1,4.0,INF203,INF301
….…
…
```

```
Archivo2
20141005,PÉREZ/GONZÁLEZ/MARÍA ROSA,INF386,INF409,MAT286,IND333…
20090044,RONCAL/NEYRA/ANA CECILIA,CIV212,CON547…
20101234,…
…
```

El primer archivo registra el código, nombre y el número créditos del curso y los códigos de los cursos requisitos para poder matricularse. El segundo archivo contiene un listado de matrícula, en cada línea aparece el código y el nombre de un alumno y las claves de los cursos en los que se ha matriculado. Un alumno puede llevar varios cursos, no necesariamente los alumnos llevan la misma cantidad de cursos.

CONSIDERACIONES:

1. Los nombres de los archivos se ingresarán en los argumentos de la función main.
2. Cada clase deberá definirse en dos módulos independientes (archivo .h y .cpp). No se podrá definir la implementación de los métodos en los archivos .h.
3. Deberá respetar estrictamente el encapsulamiento de datos a todo nivel.
4. NO se podrán definir funciones (ni plantillas) independientes que no estén ligadas como métodos a alguna de las clases planteadas, salvo que requiera definir la sobrecarga de los operadores << y >>.
5. NO PODRÁ AGREGAR ATRIBUTOS A LAS CLASES NI MODIFICAR SUS TIPOS DE DATOS, LA ASIGNACIÓN DINÁMICA DE LAS CADENAS DEBE SER EXACTA.

Anotaciones finales

Al finalizar el laboratorio, comprima[1] la carpeta "Laboratorio8" en un archivo con nombre <código del alumno con 8 dígitos>.zip y súbalo a la intranet del curso, en el enlace Documentos, en la carpeta \Laboratorio8\<código del horario>\<aula>.

Profesor del curso: Miguel Guanira E. San Miguel, 7 de junio del 2019.

[1] Para evitar problemas en la corrección de la prueba, utilice el programa de compresión que viene por defecto en el Windows (Zip) no 7z.