PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

LENGUAJE DE PROGRAMACIÓN 1

4ta práctica (tipo b)
(Segundo Semestre 2019)

Indicaciones Generales:

- Duración: 110 minutos.
- Se podrá usar como material de consulta solo sus apuntes de clase (NO fotocopias, impresos ni hojas sueltas).
- No se pueden emplear **variables globales, cadenas de caracteres**, **objetos** (con excepción de los objetos y clases definidos en la biblioteca **fstream**), tampoco se puede emplear cualquier función contenida en las bibliotecas stdio.h, cstdio, string.h cstring, string 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.
- 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%.
- El código comentado NO SE CALIFICARÁ.
- **La presentación, la ortografía y la gramática de los trabajos influirá en la calificación.**

Puntaje total: 20 puntos

Problema

El manejo de cadenas de caracteres en lenguaje C Estándar se realiza mediante la biblioteca de funciones denominada "string.h" o "cstring.h", sin embargo las funciones definidas en ella tienen el problema que no se fijan en la integridad de los datos, es decir que no controlan errores que se puedan producir, por ejemplo, el desbordamiento de los datos en memoria. Además la forma como se emplean difiere mucho de la forma cómo se manipulan otros tipos de datos.

Este laboratorio busca que usted desarrolle **una biblioteca estática de funciones**, que permita operar cadenas de caracteres de modo que su manipulación sea más segura y parecida a la forma en que se operan otros datos. Se busca que la implementación de estas funciones sea muy básica de modo que no se redunde en el uso de las funciones estándar de C, por eso las restricciones en el uso de las bibliotecas definidas por el lenguaje para ese fin.

Pregunta 1 (13 puntos)

Se solicita que desarrolle una biblioteca estática de funciones denominada **"Bibl-EstructuraCadena"** en la cual se definirá una estructura de datos que contendrá la cadena y una serie de funciones y sobrecargas como se indican a continuación:

(1) Estructura:

La **cadena** se manejará mediante una estructura (struct) denominada "**Cadena**" que contenga tres campos, uno, denominado "**cadena**", que maneje la cadena propiamente dicha mediante un arreglo estático de tamaño fijo, el segundo denominado **longitud,** que almacene la longitud de la cadena (**int**), de modo que no se tenga que contar caracteres cada vez que se le solicite y que se actualice en cada operación con la cadena y el tercero, denominado **overF** (bool), que indica si se produjo un desbordamiento en la capacidad del arreglo (***true*** si hubo desbordamiento y ***false*** si no lo hubo). No podrá agregar más campos, eliminar alguno o modificar los tipos de datos. Tampoco se requerirá el uso de un carácter de terminación en el campo "**cadena**". El tamaño del arreglo se definirá en 100.

(2) Funciones:

La biblioteca implementará las siguientes funciones:

- ✓ **inicializa (cadena, car, cant)** → la función recibirá como parámetro una estructura de tipo ***Cadena,*** un caracter y un valor entero. El campo ***cadena*** se llenará con el carácter contenido en ***car*** hasta completar el valor indicado en ***cant*** y se pondrá el campo ***overF*** en ***false***. Sí, el valor de ***cant*** sobrepasa la capacidad del arreglo, no se colocará el carácter en el arreglo y el campo ***overF*** se tornará en true. Si en la llamada a la función no se coloca el valor para el parámetro ***cant***, se llenará todo el arreglo con el carácter indicado en ***car***, el campo ***longitud*** tomará el valor máximo y el campo ***overF*** se tornará en false. Si tampoco se pone en la llamada el carácter, la función modificará la longitud de la cadena en cero y en false en el desborde.

- ✓ **leer (arch, cadena, car)** → Permite lee desde un archivo de textos uno a uno los caracteres del buffer de entrada hasta encontrar un carácter igual al contenido en ***car***, dando por terminada la lectura y colocando la cantidad de caracteres leídos en la longitud (sin contar el carácter de terminación) y false en el desborde. Si la función es invocada sin el carácter, la lectura se detendrá cuando se encuentre un cambio de línea ('\n'). Si al leer un carácter la cadena ya está llena y el carácter no es el carácter de terminación, se detiene la lectura y se coloca true en el desborde.

- ✓ **imprimir (arch, cadena)** → Permite imprimir todos los caracteres de la cadena en un archivo de textos, siempre y cuando el campo ***overF*** esté en false, de lo contrario se imprimirá solo un mensaje indicando el desborde. La operación de impresión debe realizarse de modo que cualquier otra operación posterior de salida de datos se realice en la misma línea.

- ✓ **longitud(cadena)** → Permite devolver la longitud de la cadena, siempre y cuando el campo ***overF*** esté en false; devolverá -1 en caso contrario.

- ✓ **recortar(cadena)** → Permite eliminar todos los "espacios" sobrantes de la cadena (blancos, tabuladores y cambios de línea), siempre y cuando el campo ***overF*** esté en false. La función eliminará todos los "espacios" al inicio y al final de la cadena y solo debe mantener entre palabra y palabra un espacio en blanco. Esto es, si la cadena fuera:

  " \t \t\n\t Juan \t\t Pérez \n \n\t López\t\t \n\n\t"

  Luego de ejecutar la función quedará como: **"Juan Pérez López"**.

- ✓ **intercambiar(cad01, cad02)** → Al ejecutar la función todos los campos de una estructura se intercambiarán con la otra, siempre y cuando el campo ***overF*** de ***cad01*** y ***cad02*** estén en false.

(3) Sobrecarga de operadores:

La biblioteca implementará las siguientes sobrecargas:

- ✓ **Asignación (cad01 & cad02;)** → Sobrecargando el operador **&**. Se asignarán todos los campos de ***cad02*** en ***cad01***, siempre y cuando el campo ***overF*** de ***cad02*** esté en false.
- ✓ **Agregación (cad01 += cad02;)** Sobrecargando operador **+=**. Agrega la cadena de caracteres contenida en la estructura ***cad02*** al final de la cadena de la estructura ***cad01,*** siempre y cuando el campo ***overF*** de ***cad01*** y ***cad02*** esté en false. Si la cadena tiene capacidad para contener las dos cadenas se le asigna y modifica el valor de ***long***; si no es así, no se realiza la asignación y el campo ***overF*** de ***cad01*** se pone en ***true***.
- ✓ **Concatenación (cad01 + cad02;)** Sobrecargando operador **+**. La función debe devolver una estructura ***Cadena*** con la concatenación de las cadenas ***cad01*** y ***cad02***, de manera similar al del operador **+=** pero en este caso no se deben modificar ***cad01*** ni ***cad02***.
- ✓ **Comparación (cad01 == cad02, cad01 > cad02 y cad01 < cad02)** → Sobrecargando los operadores **==, >** y **<**. Se comparan los dos operandos de tipo ***Cadena,*** siempre y cuando el campo ***overF*** de ***cad01*** y ***cad02*** esté en false**.** Se devuelve true si se cumple la condición y false si no.
- ✓ **Lectura (arch>>cadena)** Sobrecargando operador **>>**. Permite leer, desde el archivo ***arch***, uno a uno los caracteres del buffer de entrada hasta encontrar el carácter de cambio de línea.
- ✓ **Impresión (arch<<cadena)** Sobrecargando operador **<<**. Permite imprimir, en el archivo ***arch***, la cadena contenida en la estructura, siempre y cuando el campo ***overF*** esté en false.

CONSIDERACIONES:

- La solución debe contemplar la elaboración de: un proyecto de implementación y prueba de las funciones (deben probarse todas las funciones y sobrecargas), un proyecto que genere la biblioteca estática y un proyecto donde se pruebe la biblioteca ya compilada. La prueba de las funciones debe ser hecha lo más simple posible pero que muestre claramente los resultados correctos. Los proyectos se denominarán: "**EstructuraCadena-ImplementacionYPrueba**", "**EstructuraCadena**", "**EstructuraCadenaPruebaCompilada**" respectivamente. Los tres proyectos deberán colocarse en una carpeta denominada "**Pregunta01**"

PARTE 2: Reutilización de la Biblioteca estática (7 puntos)

Se solicita que desarrolle un proyecto denominado "**Pregunta02"** en el cual se utilizará la biblioteca estática **"EstructuraCadena"** (compilada). El proyecto solucionará el problema planteado a continuación:

Se tiene un archivo de textos que contiene un registro de los cursos de una institución educativa. El archivo es similar al que se muestra a continuación:

```
    INF263  ,    ALGORITMIA   ,    3.75,     INGENIERIA     INFORMATICA
MEC270,        PROCESOS          DE         MANUFACTURA,           4.00,       INGENIERIA INDUSTRIAL
  MEC289,  TURBOMAQUINAS  Y MAQUINAS       DE DESPLAZAMIENTO    POSITIVO,  4.25,  INGENIERIA    MECÁNICA
MEC206,TERMODINAMICA 1,4.50,INGENIERIA MECÁNICA
…
```

En cada línea se encuentra el código de un curso, el nombre del curso, el número de créditos y la especialidad a la que pertenece. Cada uno de estos campos estará separado por el carácter ',' (coma). Cada palabra que conforman los campos estará separada por uno o más "espacios" (espacios en blanco o tabuladores), pudiendo haber "espacios" al inicio o al final de los campos.

Empleando las funciones y sobrecargas de la biblioteca estática "**EstructuraCadena**" (**compilada**), deberá leer los datos de cada línea y colocarlos en cuatro arreglos, uno de ellos contendrá solo los códigos de los cursos, otro los nombres de los cursos, el siguiente los créditos y el último las especialidades (*). El arreglo con los créditos deberá ser de tipo "doublé", mientras que los otros serán del tipo "struct Cadena". Luego empleando estos cuatro arreglos deberá emitir el siguiente reporte:

```
        ESPECIALIDAD          CODIGO      CUURSO                                          CREDITOS
=============================================================================================================
  1)   Física (*)             FIS220      Técnicas computacionales en física (*)            4.50
  2)   Física                 FIS247      Óptica Y Fotónica                                 2.25

  …    …                      …           …                                                  …
 35)   Física                 FIS203      Física 4                                          5.00
-------------------------------------------------------------------------------------------------------------
                                                                          Total:           235.75
=============================================================================================================
481)   Ingeniería Mecánica    MEC229      Mecánica de fluidos                               4.00
482)   Ingeniería Mecánica    MEC23       Motores de combustión                             2.50
  …    …                      …           …                                                  …
-------------------------------------------------------------------------------------------------------------
                                                                           Total            …
=============================================================================================================
```

El reporte debe estar perfectamente tabulado (sin usar el carácter '\t'). Los datos deben aparecer ordenados ascendentemente por especialidad. La ordenación la debe hacer usando QuickSort o algún otro algoritmo con similar eficiencia (n log n). (*)**Las palabras de cada campo solo estarán separadas solo por un espacio en blanco y no habrán "espacios" al inicio o final del campo**.

Anotaciones finales

Al finalizar el laboratorio, comprima[1] las dos carpetas creadas (Pregunta01 y Pregunta02) 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 \Laboratorio4\<código del horario>\<aula>.

Profesores del curso: Rony Cueva
Miguel Guanira E.

San Miguel, 27 de septiembre 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.