PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

LENGUAJES DE PROGRAMACIÓN 1
3ra práctica (tipo b)
Primer Semestre 2021

Indicaciones Generales:

- Duración: 110 minutos.

Obligatoriamente los alumnos deberán mantener en todo momento el AUDIO Y VIDEO de sus computadores abierto de modo que puedan recibir los comunicados que se hagan durante el examen y la revisión de los trabajos que estén desarrollando. 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**, **ni objetos**(con excepción de los elementos de iostream, iomanip y fstream). 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 función 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 proyectos deben obligatoriamente desarrollarse en NetBeans bajo el sistema operativo Windows. No se revisarán los proyectos desarrollados en otros sistemas operativos o IDEs.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no muestres resultados o que estos no sean 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 ALAEVALUACIÓN 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.

Cuestionario:

La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en el capítulo 4 del curso: "Funciones y alcance de variables". En este laboratorio se desarrollará **una biblioteca estática de funciones** en la que se implementen sobrecargas de operadores y funciones que le permitan manejar las infracciones de tránsito registradas por una entidad reguladora.

Cree una **carpeta** denominada "Infracciones2021-1Lab03", dentro de ella cree dos **carpetas** denominadas "Lab03-Parte01" y "Lab03-Parte02", en estas últimas colocará los proyectos solicitados en las preguntas 1 y 2 respectivamente. DE NO COLOCAR ALGUNO DE ESTOS REQUERIMIENTO SE LE DESCONTARÁ 3 PUNTOS DE LA NOTA FINAL.

PARTE 01 (14 puntos): CREACIÓN DE LA BIBLIOTECA ESTÁTICA

Se solicita que desarrolle una biblioteca estática denominada **"CompilacionBibInfrac"** en la cual se definirán las estructuras de datos que describen a continuación, además definirá una serie de operadores sobrecargados que permitirá manejar las estructuras. Las estructuras se definen a continuación:

➢ **Para manejar los conductores**: La estructura (struct) se denominará "**ConductorSt**" y deberá contener lo siguiente: 1) un campo denominado **licencia**, este campo almacenará la licencia del conductor (int), 2) un campo denominado **nombre** definido por una cadena de caracteres, 3) un campo denominado **faltas**, definido por un arreglo de estructuras de tipo struct FaltaSt, considerar un máximo de 20 infraciones, 4) un campo denominado numFaltas (int) que llevará la cuenta de las faltas cometidas por el conductor, 5) tres campos de tipo int denominados: numFaltasLeves, numFaltasGraves y numFaltasMuyGraves que llevaran la cuenta del número de faltas cometidas por tipo, 6) tres campos de tipo double denominados: montoFaltasLeves, montoFaltasGraves, montoFaltasMuyGraves, que acumularán los motos a pagar por tipo de falta.

➢ **Para manejar las faltas de un conductor**: La estructura (struct) se llamará "FaltaSt" deberá contener lo siguiente: 1) un campo denominado placa definido por una cadena de caracteres, 2) un campo denominado fecha (int) que almacenará una fecha en el formato AAAAMMDD, 3) un campo denominado codInf (int) que almacenará el código de la infracción, 4) un campo denominado multa (double) que almacenará el monto a pagar por la multa, 5) un campo denominado gravedad, cadena de caracteres que almacenará: "LEVE", "GRAVE" o "MUY GRAVE",.

➢ **Para manejar las infracciones cometidas**: La estructura (struct) se denominará "InfraccionSt" deberá contener lo siguiente: 1) un campo denominado licencia, este campo almacenar una licencia del conductor que ha cometido una falta (int), 2) un campo denominado placa definido por una cadena de caracteres, 3) un campo denominado codInf con el código de la infracción cometida (int), 4) un campo denominado fecha (int) que almacenará una fecha en el formato AAAAMMDD.

➢ **Para manejar las infracciones establecidas que se pueden cometer**: estructura (struct) se denominará "InfraccionEstabSt" deberá contener lo siguiente: 1) un campo denominado codigo con el código de la infracción (int), 2) un campo denominado gravedad definido por una cadena de caracteres que indicará la gravedad de la falta ("LEVE", "GRAVE" o "MUY GRAVE"), 3) un campo denominado multa (double) que almacenará el pago que debe hacerse por la multa.

"**DEBE EMPLEAR OBLIGATORIAMENTE LOS NOMBRES DE LAS ESTRUCTURAS Y SUS CAMPOS**"

Las operaciones que la biblioteca estática permitirá realizar a través de sobrecargas de operadores se definen a continuación:

➢ **Lectura**:

- Sobrecargando el operador **>>** de modo que permita leer un conductor de un archivo de textos. La operación **(arch >> conductor;)** involucrará un archivo de textos y una variable de tipo "**ConductorSt**". La sobrecarga deberá devolver **true** si se pudo leer la información y **false** si se llegó al final de archivo. Una línea de archivo tendrá la siguiente forma:

```
30018447  FUENTES/QUISPE/CINTHIA_DELIA          (licencia  nombre)
```

  Además deberá inicializar los campos restantes de la estructura.
- Sobrecargando el operador **>>** de modo que permita leer una falta cometida de un archivo de textos. La operación **(arch >> infraccion;)** involucrará un archivo de textos y una variable de tipo "InfraccionSt". La sobrecarga deberá devolver **true** si se pudo leer la información y **false** si se llegó al final de archivo. Una línea de archivo tendrá la siguiente forma:

```
30018447   01C-880  25/8/2020  102       (licencia  placa fecha  código  de  infracción)
```

  El resto de los campos quedará sin alteración
- Sobrecargando el operador **>>** de modo que permita leer una falta establecida de un archivo de textos. La operación **(arch >> infraccionEst;)** involucrará un archivo de textos y una variable de tipo "InfraccionEstabSt". La sobrecarga deberá devolver **true** si se pudo leer la información y **false** si se llegó al final de archivo. Una línea de archivo tendrá la siguiente forma:

```
103   Detener_el_vehículo_bruscamente_sin_motivo  Grave   316.00  (código descripción gravedad multa)
```

➢ **Agregación**:

- Sobrecargando el operador + de modo que permita agregar **una** infracción a **un** conductor. La operación (**conductor + infracción;**) colocará los datos de la infracción al final de su lista de faltas, modificando el número de faltas (numFaltas). No debe modificar los campos que no están involucrados en la operación.

- Sobrecargando el operador + de modo que permita completar los datos de algunas faltas cometidas por un conductor. La operación (conductor + infracciónEst;) colocará la gravedad y multa en todas las faltas que coincidan con la infracción dada (infracción Est). No debe modificar los campos que no están involucrados en la operación.

- **Cálculo de las infracciones cometidas**: sobrecargando el operador ++ de modo que permita calcular los montos y cantidades según el tipo de infracción de un conductor.
- **Amnistía de infracciones**: sobrecargando el operador * de modo que permite aplicar una serie de beneficios a un conductor. La operación (**conductor * fecha;**), donde la fecha estará dada por un valor entero de la forma AAAAMMDD. La amnistía consistirá en reducir en un 25% sus infracciones graves, en un 8% sus infracciones muy graves y eliminar las infracciones leves de aquellas infracciones que cometió antes de la fecha dada. Por eliminar se entiende que sacará del arreglo el dato completo, desplazando el resto de modo que no queden "huecos" en el arreglo.
- **Impresión**: sobrecargando el operador << de modo que permita imprimir la información de un conductor. La operación (**arch << conductor;**) permitirá imprimir en un archivo de textos los datos contenidos en una variable de tipo "**ConductorSt**". El formato será el siguiente:

```
Conductor   : Arca/Amezquita/Edric-Ronald
Licencia No.: 54738291
============================================================
Infracciones cometidas:
------------------------------------------------------------
No.   Fecha        Placa     Infracción    Gravedad    Multa
1)    23/06/2018   G4S-021       503       Muy grave   504.50
2)    03/09/2020   K5Q-737       801       Leve        735.00
…     …            …              …                   …
============================================================
                              Cantidad      Total
Infracciones leves:              3          5843.25
Infracciones graves:             1         33111.75
Infracciones muy graves:         0             0.00
```

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

Consideraciones:

La solución debe contemplar la elaboración de: 1) un proyecto de implementación y prueba de las sobrecargas, 2) un proyecto que genere la biblioteca estática y 3) un proyecto donde se pruebe la biblioteca ya compilada.

La prueba de las sobrecargas para el primer y tercer proyecto deben ser hecha lo más simple posible pero que muestre claramente son correctas (**No debe solucionar aquí el problema de la pregunta 2**). Los proyectos se denominarán: "**ImplementacionYPruebaBibInfrac**", "**CompilacionBibInfrac**", "**CompilacionPruebaBibInfrac**" respectivamente. Los tres proyectos deberán colocarse en una carpeta denominada "**Lab03-Parte01**".

PARTE 2(6 puntos): REUTILIZACIÓN DE LA BIBLIOTECA ESTÁTICA.

Desarrolle en la carpeta "Lab03-Parte02" un proyecto denominado "**PruebaCompiladaConductores**" en el cual se utilizará obligatoriamente las sobrecargas de la biblioteca estática (compilada) "**CompilacionBibInfrac**". **La pregunta no se evaluará si no se usa la biblioteca compilada (.a).** El proyecto ejecutará las tareas descritas a continuación utilizando las sobrecargas definidas en la biblioteca:

a) Leer los dato de los conductores contenidos en un archivo de textos como se muestra a continuación y los coloque en un arreglo de estructuras del tipo "**ConductorSt**":

```
30018447  FUENTES/QUISPE/CINTHIA_DELIA
32517791  ZORRILLA/LARA/ARTURO
…
```

b) Leer los datos de las faltas cometidas y asignárselas a cada conductor del arreglo anterior. El archivo que es similar al siguiente:

```
30018447   O1C-880  25/8/2020  102
78153392   A7R-205  18/12/2017 672
…
```

c) Leer una a una las infracciones establecidas y completar las multas y gravedad de las infracciones cometidas por los conductores. del El archivo que es similar al siguiente:

```
103   Detener_el_vehículo_bruscamente_sin_motivo  Grave   316.00
207   Utilizar_la_bocina_para_llamar_la_atención_en_forma_inncesaria    Leve    158.002
…
```

d) Calcular los montos y cantidades según el tipo de infracción de los conductores del arreglo.

e) Emitir un reporte en el que se muestren los datos de cada conductor.

f) Aplicar una amnistía a los conductores para una fecha ingresada por teclado.

g) Emitir un reporte en el que se muestren los datos de cada conductor con la amnistía aplicada.

Al finalizar la práctica, comprima la carpeta de su proyecto empleando el programa Zip que viene por defecto en el Windows, **no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares**. Luego súbalo a la tarea programa en Paideia para estaeste laboratorio.

Profesores del curso: Miguel Guanira
Rony Cueva

San Miguel, 30 de abril del 2021.