PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 2
4ta práctica (tipo b)
(Segundo Semestre 2024)

Indicaciones Generales:
Duración: **110 minutos.**

- No se permite el uso de apuntes de clase, fotocopias ni material impreso.
- No se pueden emplear variables globales, ni objetos (con excepción de los elementos de **iostream**, **iomanip** y **fstream**). No se puede utilizar la clase **string**. Tampoco se podrán emplear las funciones de C que gesten memoria como **malloc**, **realloc**, **memset**, **strdup**, **strtok** o similares, 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. No podrá emplear plantillas en este laboratorio.
- 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 el archivo main.cpp deberá incluir un comentario en el que coloque claramente su nombre y código, de no hacerlo se le descontarán 0.5 puntos en la nota final.
- El código comentado no se calificará. De igual manera no se calificará el código de una función si esta función no es llamada en ninguna parte del proyecto o su llamado está comentado.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no muestren 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 con el reglamento disciplinario de nuestra institución, constituye una falta grave copiar del trabajo realizado por otra persona o cometer plagio.
- No se harán excepciones ante cualquier trasgresión de las indicaciones dadas en la prueba.

Puntaje total: 20 puntos

- La unidad de trabajo será t:\ (Si lo coloca en otra unidad, no se calificará su laboratorio y se le asignará como nota cero). En la unidad de trabajo t:\ colocará el(los) proyecto(s) solicitado(s).
- Cree allí una carpeta con el nombre Lab04_2024_2_CO_PA_PN donde: CO indica: Código del alumno, PA indica: Primer Apellido del alumno y PN indica: primer nombre. De no colocar este requerimiento se le descontarán 3 puntos de la nota final.

Cuestionario

- La finalidad principal de este laboratorio es la de reforzar los conceptos contenidos en el capítulo 3 del curso: Arreglos y punteros. En este laboratorio se trabajará con punteros genéricos, registros de registros y métodos de asignación de memoria.
- Al finalizar la práctica, comprima la carpeta dada en las indicaciones iniciales 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.

Para el diseño de su solución, considere que:

- No podrá emplear arreglos estáticos de más de una dimensión.
- No puede manipular un puntero con más de un índice.
- No puede emplear arreglos auxiliares, estáticos o dinámicos, para guardar los datos de los archivos. Los archivos solo se pueden leer una vez.
- **Puede elegir el tipo de asignación de memoria dinámica (exacta o por incrementos).**
- **Se le proveerá una biblioteca estática con funciones para probar la carga de datos**

(probarCargaInventario y pruebaCargaMenu).

Descripción del caso

Una cafetería necesita un programa en C++ que permita gestionar y actualizar su **menú diario** de manera automatizada, antes de abrir el local al público. Este sistema debe verificar la **disponibilidad de insumos** en el inventario cada mañana y, en función de ello, habilitar o deshabilitar las bebidas que pueden ofrecerse. Si los insumos necesarios para la preparación de una bebida están en cantidad suficiente, dicha bebida se marcará como disponible en el menú. De lo contrario, la bebida deberá quedar inhabilitada temporalmente hasta que el inventario sea restablecido.

El programa deberá ser capaz de:

1. **Leer y procesar** la información del inventario desde un archivo, actualizando las cantidades de cada insumo en el sistema.
2. **Comparar las necesidades de insumos** para cada bebida con las cantidades disponibles en el inventario.
3. **Actualizar el estado del menú**, habilitando o deshabilitando las bebidas en función de la disponibilidad de los insumos.
4. **Mostrar el menú actualizado**, indicando qué bebidas están disponibles para la venta y cuáles no, de manera que el personal de la cafetería pueda ver claramente las opciones antes de iniciar el día.

Este sistema permitirá a la cafetería gestionar de manera eficiente su menú, reduciendo el tiempo de preparación y evitando problemas por falta de insumos a lo largo del día.

Para ello se cuenta con los siguientes archivos de texto: **inventario.csv** (código, nombre, cantidad y unidad de medida), **menu.csv** (código, nombre, descripción, tipo de bebida, precio, cantidad de ventas diarias estimadas) e **insumos-bebidas.csv** (código de la bebida, código del insumo, cantidad, unidad de medida).

inventario.csv

```
I001,Café Arábica,50,kg
I002,Café Robusta,30,kg
```

Código, Nombre, Cantidad, Unidad de medida

menu.csv

```
B001,Espresso Doble,Un espresso intenso...,C,2.50,120
B002,Café Latte,Suave y cremoso elaborado...,C,3.20,150
```

Código, Nombre, Descripción, Tipo de bebida, precio, cant. vendidas est.

insumos-bebidas.csv

```
B001,I001,10,g
B001,I002,8,g
B002,I001,12,g
B002,I002,6,g
B002,I003,200,g
```

Código de bebida, Codigo del Insumo, Cantidad, Unidad de medida

Con esta información se solicita elaborar un proyecto en Netbeans cuya función “main” estará compuesta por el siguiente código:

```
#include <cstdlib>
#include "cafeteria.h"
#include "pruebas.h"

using namespace std;

int main(int argc, char** argv) {
    void *inventario, *menu;

    cargarInventario("inventario.csv", inventario);
    probarCargaInventario("prueba-inventario.txt", inventario);

    cargarMenu("menu.csv", menu);
    pruebaCargaMenu("menu-inicial.txt", menu);
    actualizarMenu("insumos-bebidas.csv", inventario, menu);
    reporteMenu("menu.txt", menu);

    return 0;
}
```

Se les ha hecho entrega de una biblioteca estática **liblab04pruebas.a** así como de los archivos de cabecera pruebas.h y enums.h los cuales serán usados en su proyecto para verificar que la carga de datos se realizó de forma correcta. No es necesario que implemente las funciones probarCargaInventario ni pruebaCargaMenu ya que estas se encuentran en la biblioteca estática.

Parte 1 (6 puntos)

Implementación de las funciones **cargarInventario** y **cargarMenu**

Desarrolle las funciones **cargarInventario** y **cargarMenu**, las cuales se encargarán de leer los datos del inventario y el menú de la cafetería desde los archivos **inventario.csv** y **menu.csv**, respectivamente.

- **cargarInventario (3 puntos)**: Esta función procesará el archivo **inventario.csv** y cargará la información en la estructura **Inventario**, almacenando cada insumo con sus respectivos detalles (código, nombre, cantidad disponible y unidad de medida). Estos insumos representan los ingredientes necesarios para preparar las bebidas ofrecidas en la cafetería.
- **cargarMenu (3 puntos)**: A continuación, **cargarMenu** leerá el archivo **menu.csv** y organizará los datos en la estructura **Menú**, que contendrá la lista de bebidas disponibles, con su código, nombre, tipo de bebida, precio, cantidad estimada de ventas y una descripción.

Las estructuras de datos **Inventario** y **Menú** a las que hacen referencia estas funciones están ilustradas en las figuras 1 y 2, donde se muestran cómo se almacenan y organizan los datos dentro del sistema.

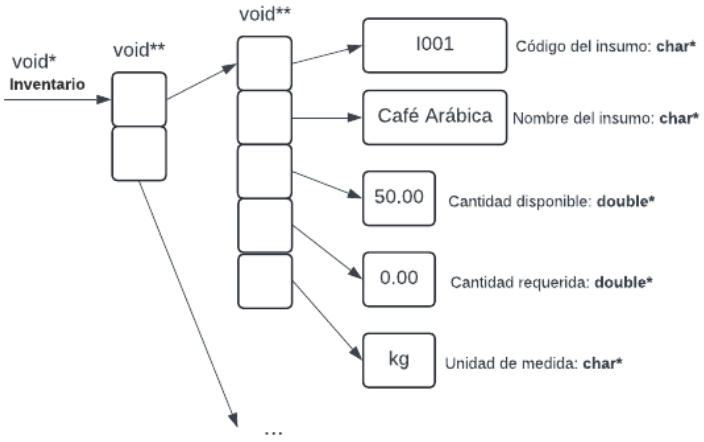

Figura 1 - Estructura Inventario

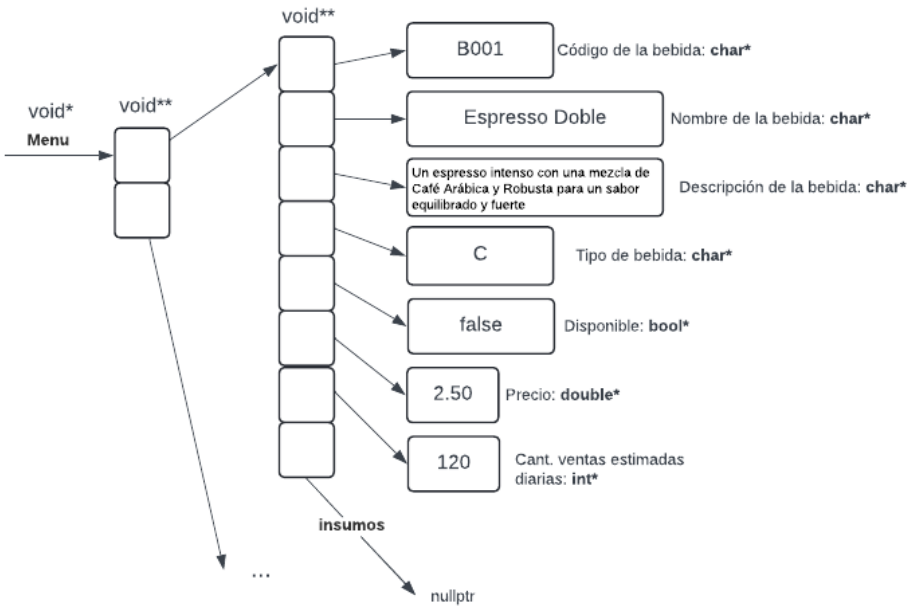

Figura 2 - Estructura Menú

En la estructura **Inventario**, el campo correspondiente a la **cantidad requerida** debe ser inicializado en **cero** durante esta fase del laboratorio. La **unidad de medida** se utiliza para representar la cantidad disponible de cada insumo en el inventario. Por ejemplo, "50 kg" indica la cantidad de un insumo disponible.

En cuanto a la estructura **Menú**, el campo **tipo de bebida** admite los siguientes valores:

- C: Café
- T: Té
- I: Infusión
- H: Chocolate.

Además, el campo **disponible** de cada bebida en el menú debe ser inicializado en **true**, lo que indica que el menú aún no ha sido actualizado (la actualización se realizará en la siguiente fase del laboratorio). Finalmente, el último campo de cada bebida, que representa los **insumos necesarios** para su preparación, debe ser inicializado en **nullptr** en esta etapa.

Parte 2 (10 puntos)

Implemente la función **actualizarMenu**. Esta función recibirá como parámetros las estructuras Inventario y Menú, previamente cargadas, y utilizará el archivo **insumos-bebidas.csv** para cargar los insumos necesarios para la elaboración de las bebidas.

La función **actualizarMenu** creará una estructura dinámica, la cual será referenciada por el campo insumos de la estructura Menú como se muestra en la Figura 3, para cada bebida. La estructura de insumos estará compuesta por un puntero que apuntará a un insumo en la estructura **Inventario**, el cual deberá ser identificado mediante el campo código inventario. Para realizar esta búsqueda, no se podrá utilizar la función bsearch; en su lugar, se deberá recorrer manualmente la estructura **Inventario** hasta encontrar el insumo correspondiente por su código.

Además, se deberán cargar la cantidad requerida del insumo y su unidad de medida (por ejemplo, "10 g"). Durante este proceso, se actualizará la **cantidad requerida** de cada insumo a través del **puntero al insumo en la estructura Inventario**. La actualización se realizará aplicando la **conversión adecuada** de unidades de medida (por ejemplo, 1 kg equivale a 1000 g) y multiplicando con la **cantidad de ventas diarias estimada de la bebida**, ajustando la cantidad necesaria para preparar la bebida. Esto garantizará que el insumo de la bebida refleje correctamente la cantidad requerida en gramos, mientras que el insumo en el inventario mantiene las cantidades en kilogramos.

Si la cantidad disponible en el inventario es **no es suficiente** (es decir, menor a la cantidad solicitada) para **alguno de los insumos** de una bebida, el campo disponible de dicha bebida se actualizará a **false,** indicando que la bebida no puede ser elaborada debido a la falta de insumos. De lo contrario, permanecerá en **true**.

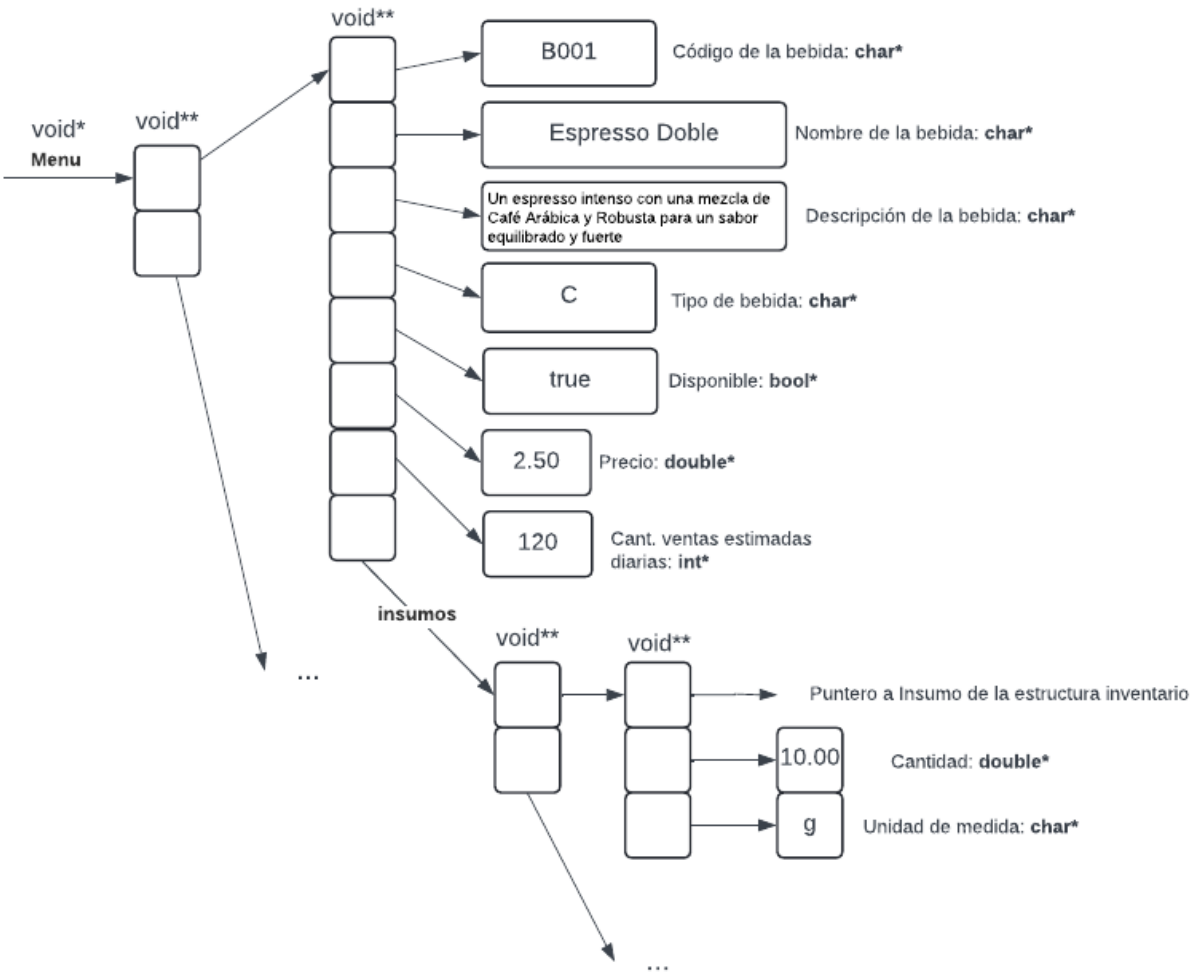

Figura 3 - Estructura Menú Actualizada

Parte 3 (4 puntos)

Es importante destacar que esta sección del laboratorio dependerá directamente de la correcta implementación de las dos primeras partes. Por lo tanto, asegúrese de que esas fases estén completadas antes de continuar. A continuación, implemente la función **reporteMenu**, la cual recibirá como parámetro el menú actualizado, el reporte solo mostrará si las bebidas se encuentran **disponibles**. Los datos mostrados en el reporte son: **Tipo de Bebida (Café, T: Té, I: Infusión, H: Chocolate)**, **Nombre de la bebida**, **descripción de la bebida** y el **precio**. Esta función será responsable de generar un informe que contenga los detalles del menú del día, el cual deberá ser guardado en un archivo denominado **menu.txt**.

========== Menú del día ==========

Café: Espresso Doble

Un espresso intenso con una mezcla de Café Arábica y Robusta para un sabor equilibrado y fuerte

Precio: S/ 2.50

Disponible: Si

Té: Chai Latte

Mezcla especiada de té negro leche vaporizada y especias como canela y cardamomo

Precio: S/ 3.20

Disponible: No

Café: Café Latte

Suave y cremoso elaborado con una mezcla de cafés y leche vaporizada para una textura delicada

Precio: S/ 3.20

Disponible: No

Profesores del curso: Miguel Guanira, Andrés Melgar, Rony Cueva, Eric Huiza, Erasmo Gómez

Pando, 27 de septiembre de 2024