Universidad de San Carlos de Guatemala
Centro Universitario de Occidente
División de Ciencias de la Ingeniería

Estructuras De Datos
Laboratorio

Practica 1
Solitario con Pilas, Colas y Listas
Manual Técnico

Jorge Anibal Bravo Rodríguez
202131782

1. HERRAMIENTAS USADAS

Visual Studio Code: Visual Studio Code es un editor de código fuente que proporciona un entorno de desarrollo integrado (IDE) ligero y altamente personalizable. Visual Studio Code es compatible con C++ y ofrece extensiones que facilitan el desarrollo en este lenguaje, lo que incluye resaltado de sintaxis, depuración y administración de proyectos.

G++: G++ es un compilador de línea de comandos que compila y enlaza programas en C++, generando el correspondiente archivo ejecutable. Es ampliamente utilizado para compilar programas en C++ y está disponible en sistemas Linux de forma predeterminada. En sistemas Windows, se puede utilizar con MinGW. G++ es una herramienta fundamental para desarrollar aplicaciones en C++ y permite compilar y enlazar programas de manera eficiente.

C++: es un lenguaje de programación que evolucionó a partir del lenguaje C para permitir la manipulación de objetos. Es conocido por su potencia y versatilidad, y sigue siendo ampliamente utilizado en la actualidad para desarrollar programas de alto rendimiento, como sistemas operativos, videojuegos y aplicaciones en la nube. C++ es un lenguaje compilado, multiparadigma, principalmente imperativo y orientado a objetos.

GitHub: es una plataforma de alojamiento de código basada en la nube que ofrece un sistema de control de versiones llamado Git.

Make: es una herramienta de construcción que automatiza el proceso de compilación y enlazado de programas. Utiliza un archivo de configuración llamado makefile para definir las reglas y dependencias necesarias para compilar un proyecto. El makefile contiene instrucciones sobre cómo compilar y vincular los archivos fuente, así como las dependencias entre ellos. Al ejecutar el comando make en el directorio que contiene el makefile, se inicia el proceso de compilación según las reglas definidas en el makefile, lo que facilita la gestión de proyectos de software complejos.

2. ARCHIVO MAKEFILE

El código c++ escrito para esta práctica fue hecho en un editor de código (Visual Studio Code) por lo que prescindí de herramientas que automatizaran la compilación desde un primer momento.

Para generar el ejecutable final utilicé la herramienta make propuesta en clase junto al archivo de compilación Makefile, tomando como ejemplo el código alojado en github por parte del auxiliar. Dicho archivo es el siguiente:

En donde se importan todos los archivos utilizados en el proyecto.
Para lo compilación se debe ejecutar el siguiente comando:

make

El programa make buscará el archivo makefile y realizará la compilación.
Cabe aclarar que se usaron librerías estándar, sin emplear librerías propias de un sistema operativo en particular

3. ESTRUCTURAS DE DATOS

La práctica requería la implementación por nuestra cuenta de ciertas estructuras, tales como pilas, colas y listas.
En mi caso, realicé estas estructuras usando nodos que trabajaban con clases de tipo Carta. Los nodos son muy sencillos, por lo que no los detallaré. Estos eran los siguientes:

- NodoDoble: Nodo con siguiente, anterior y carta
- Nodo: Nodo con siguiente y carta

3.1 PILA

- Se incluyen las librerías necesarias y se define el espacio de nombres std.
- Se define la clase Pila que representa una pila de cartas. El constructor inicializa la pila con un símbolo dado y establece la cima de la pila como nulo.
- La función obtenerLongitud devuelve la longitud actual de la pila.
- La función apilar agrega una nueva carta a la cima de la pila.
- La función desapilar elimina y devuelve la carta en la cima de la pila.
- La función verCima devuelve una representación de la carta en la cima de la pila.
- La función obtenerCima devuelve la carta en la cima de la pila.
- La función apilarCartaJuego verifica si una carta puede ser apilada en la pila según ciertas reglas del juego, y la apila si es posible.

3.2 COLA

- Constructor Cola::Cola(): Inicializa la cola estableciendo los punteros primero y último en nulo, y la longitud en 0.
-
- Método Cola::encolar(Carta carta): Crea un nuevo nodo con la carta proporcionada y lo agrega al final de la cola. Actualiza el puntero último y aumenta la longitud de la cola.

- Método Cola::desencolar(): Elimina el primer elemento de la cola y devuelve la carta que estaba en ese nodo. Actualiza el puntero primero y disminuye la longitud de la cola.

- Método Cola::obtenerCartaADesencolar(): Devuelve la carta que se desencolará, pero no la elimina de la cola. Si la cola está vacía, muestra un mensaje de advertencia y devuelve un puntero nulo.

- Método Cola::verSuperior(): Devuelve la carta en la parte superior de la cola (la primera en ser desencolada) sin eliminarla. Si la cola está vacía, devuelve "NLL".

- Método Cola::verLongitud(): Devuelve la longitud actual de la cola.

3.3 LISTA DOBLEMENTE ENLAZADA

- Constructor ListaDoble::ListaDoble(): Inicializa la lista estableciendo los punteros primero y último en nulo, y la longitud en 0.

- Método ListaDoble::obtenerLongitud(): Devuelve la longitud actual de la lista.

- Método ListaDoble::insertarAlInicio(Carta carta):* Inserta un nuevo nodo con la carta proporcionada al inicio de la lista.

- Método ListaDoble::insertarAlFinal(Carta carta):* Inserta un nuevo nodo con la carta proporcionada al final de la lista.

- Método ListaDoble::insertarEnIndice(Carta carta, int indice):* Inserta un nuevo nodo con la carta proporcionada en el índice especificado de la lista.

- Método ListaDoble::eliminarEnIndice(int indice): Elimina el nodo en el índice especificado de la lista.

- Método ListaDoble::obtenerEnIndice(int indice): Devuelve la carta en el índice especificado de la lista.

- Método ListaDoble::insertarAlFinalJuego(Carta car):* Inserta una carta al final de la lista si cumple con ciertas condiciones de juego.

- Método ListaDoble::imprimir(int nivel): Imprime la carta en el nivel especificado de la lista, considerando si está boca arriba o boca abajo.