Universidad CENFOTEC

Escuela de Software

Código del curso: BISOFT-11.

Nombre del curso: Estructuras de Datos 1.

Sección: SCV1.

Periodo: C3-2025.

Docente facilitador: Romario Salas Cerdas.

CONSIGNA DE LA SEGUNDA PRÁCTICA

1. Datos generales de la actividad

Tipo de actividad:	Práctica.		
Fecha de entrega:	09 de noviembre del 2025, 11:59pm.	**Valor porcentual:**	10%.
Formato de entrega:	Archivo comprimido.	**Puntaje total:**	100.
Individual: Sí.	**Grupal:** No.		

2. Instrucciones generales

1. Lea cuidadosamente las instrucciones de la actividad. En caso de tener alguna duda, puede consultar con su docente.
2. Esta actividad se desarrolla de manera individual. Cualquier intento de plagio será sancionado de acuerdo con el reglamento académico vigente.
3. Al completar la actividad, el estudiante debe subir un archivo comprimido con los archivos de código fuente a la plataforma Moodle en el tiempo y espacio indicado por el docente.

3. Objetivos o competencias del curso que se evaluarán en la actividad de aprendizaje

Objetivo general del curso	Construir aplicaciones de software pequeñas y medianas, usando las estructuras de datos lineales, las estructuras de datos jerárquicas y los algoritmos de ordenamiento y búsqueda, para implementar aplicaciones de software más eficientes en tiempo y en recursos de computadora.
Objetivos específicos o	• Comprender el funcionamiento y la algorítmica de

resultados de aprendizaje que se evalúan	las estructuras de datos lineales y jerárquicas y de los algoritmos de ordenamiento y búsqueda, mediante el estudio de ejemplos y prácticas en clase, a fin de que el estudiante pueda conocer y distinguir las diferentes estructuras de datos. • Usar las estructuras de datos lineales y jerárquicas, y los algoritmos de ordenamiento y búsqueda, mediante el uso de Tipos Abstractos de Datos y de lenguajes de programación, para resolver un problema con una aplicación de software pequeña o mediana. • Analizar la eficiencia en tiempo y en espacio, de las estructuras de datos y de los algoritmos, usando la notación O, a fin de aplicar este criterio y escoger las estructuras y los algoritmos de menor complejidad. • Diseñar estructuras de datos lineales y jerárquicas por medio de Tipos Abstractos de Datos, para construir aplicaciones de software fáciles de entender, corregir y cambiar. • Elaborar aplicaciones de software eficientes, implementando estructuras de datos lineales y jerárquicas, y algoritmos de búsqueda y ordenamiento, que garanticen el menor uso de recursos y de tiempo.

4. Descripción de la actividad

Instrucciones:

El objetivo de esta tarea es implementar una cola de **Tareas** en C++, que permita almacenar y manipular una colección lineal de **Tareas**, cada uno de los cuales estará asociado a una lista enlazada simple de **Requerimientos**. Durante la implementación de la cola, el estudiante trabajará con punteros, clases basadas en tipos de datos abstractos y las operaciones fundamentales asociadas con colas y listas enlazadas de tipo simple.

Los archivos de código fuente que implementan el ADT **Tarea**, cuyos atributos propios son su **descripcion**, su **prioridad** (clasificada del 1 al 3), así como su **encargado** y su **ListaRequerimientos**, definen la clase cuyos objetos funcionarán como los nodos de la **ColaTareas**. Dichos archivos de código fuente deben incluir como mínimo los métodos necesarios para la construcción de las **Tareas** y para la inserción de **Requerimientos** en su lista interna. La implementación del ADT **Requerimiento** debe incluir su **descripcion** y su **complejidad** (valorada del 1 al 10). El llenado de la **ListaRequerimientos** de cada **Tarea** debe realizarse al momento de su inserción a la **ColaTareas**. Debido a que cada **Tarea**

almacenará internamente una referencia a una estructura de datos con memoria reservada de forma dinámica, su clase debe incluir un método destructor.

Los archivos de código fuente que implementan el ADT **ColaTareas**, cuya instancia dará estructura funcional a la aplicación descrita, deben incluir como mínimo los métodos necesarios para construir una nueva cola, destruirla (i.e., liberar la memoria utilizada de forma dinámica por la estructura), insertar una nueva **Tarea** al **final** de la cola y para eliminar la **Tarea** con la prioridad más alta de la cola (en caso de empate, la **Tarea** más cercana al **frente** de la cola será la eliminada). Eliminar una **Tarea** de la **ColaTareas** implica generar un pequeño reporte en la consola, donde se enumere su **descripcion**, su **prioridad**, los **Requerimientos** de su lista y la **complejidad** acumulada de estos últimos.

Además de todos los archivos de código fuente necesarios para el funcionamiento de la aplicación descrita, el estudiante debe desarrollar un archivo **.cpp** adicional, incluyendo una rutina **main()** en la que se construya una instancia de **ColaTareas** y se evidencie el funcionamiento de todos sus métodos solicitados dentro de un **menu()** de consola plenamente funcional, que le dé al usuario acceso a todos los comportamientos descritos.

5. Rúbrica

Esta actividad de aprendizaje será evaluada mediante la siguiente rúbrica:

1. Implementación completa y correcta de **Tarea**: **15 puntos.**
2. Implementación completa y correcta de **Requerimiento**: **15 puntos.**
3. Implementación completa y correcta de **ListaRequerimientos**: **20 puntos.**
4. Implementación completa y correcta de **ColaTareas**: **20 puntos.**
5. Evidencia del funcionamiento de la aplicación dentro de una rutina **main()** mediante un **menu()** de consola plenamente funcional: **30 puntos.**

Total: **100 puntos.**

Criterio	Deficiente (1 punto)	Regular (2 puntos)	Bueno (3 puntos)	Excelente (4 puntos)
Requerimientos de la práctica	Analiza los requerimientos presentados en la práctica de manera inadecuada	Analiza los requerimientos presentados en la práctica de manera básica	Analiza los requerimientos presentados en la práctica de manera aceptable	Analiza los requerimientos presentados en la práctica de manera adecuada
Estructuras de datos	Selecciona las estructuras de datos que se utilizan en el desarrollo de la práctica de manera inadecuada y con múltiples errores	Selecciona las estructuras de datos que se utilizan en el desarrollo de la práctica de manera básica y con varios errores	Selecciona las estructuras de datos que se utilizan en el desarrollo de la práctica de manera regular y con algún error	Selecciona las estructuras de datos que se utilizan en el desarrollo de la práctica de manera adecuada y sin errores
Codificación de las funcionalidades	Desarrolla la codificación de las funcionalidad que permiten la solución del problema de manera inadecuada y con múltiples errores	Desarrolla la codificación de las funcionalidad que permiten la solución del problema de manera elemental y con varios errores	Desarrolla la codificación de las funcionalidad que permiten la solución del problema de manera aceptable y con algún error	Desarrolla la codificación de las funcionalidad que permiten la solución del problema de manera adecuada y sin errores
Análisis de eficiencia	Emplea casos de prueba para la comprobación de la funcionalidad y eficiencia de los algoritmos de manera inadecuada y con múltiples errores	Emplea casos de prueba para la comprobación de la funcionalidad y eficiencia de los algoritmos de manera elemental y con varios errores	Emplea casos de prueba para la comprobación de la funcionalidad y eficiencia de los algoritmos de manera aceptables y con algún error	Emplea casos de prueba para la comprobación de la funcionalidad y eficiencia de los algoritmos de manera adecuada y sin errores
Entrega del informe de la solución de la práctica	Realiza la entrega de las secciones de la práctica de forma inadecuada			Realiza la entrega de la práctica con todas las secciones de forma adecuada