LISTAS DE COTEJO

Estructura de Datos y Algoritmos
Facultad de Ingeniería, UNAM, 2022
Elaboró: Dr. Marco Antonio Negrete Villanueva

Las prácticas, tareas y listas de cotejo se diseñaron considerando el marco de acreditación ANECA. Esto para que los instrumentos sirvan también como apoyo en el proceso de acreditación de la carrea de Ingeniería en Computación. La siguiente tabla muestra un resumen de las listas de cotejo de cada práctica y los resultados de aprendizaje que se evalúan en el marco de acreditación ANECA.

Instrumento	Sub Resultados de Aprendizaje evaluados
Práctica 4 “Almacenamiento en tiempo de ejecución” – lista de cotejo	1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.
Práctica 5 “Estructuras de datos lineales: Pila y cola” – lista de cotejo	1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.
Práctica 7 “Estructuras de datos lineales: Lista simple y lista circular” – lista de cotejo	1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.
Práctica 11 “Estrategias para construcción de algoritmos” – lista de cotejo	1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos. 2.2. La capacidad de identificar, formular y resolver problemas de ingeniería en su especialidad; elegir y aplicar de forma adecuada métodos analíticos, de cálculo y experimentales ya establecidos; reconocer la importancia de las restricciones sociales, de salud y seguridad, ambientales, económicas e industriales.
Práctica 12 “Recursividad” – lista de cotejo	1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos. 2.2. La capacidad de identificar, formular y resolver problemas de ingeniería en su especialidad; elegir y aplicar de forma adecuada métodos analíticos, de cálculo y experimentales ya establecidos; reconocer la importancia de las restricciones sociales, de salud y seguridad, ambientales, económicas e industriales.

Práctica 4 – Almacenamiento en tiempo de ejecución (multiplicación de matrices)

Resumen: la práctica consiste en la aplicación de las funciones malloc y free para reserva y liberación de memoria en tiempo de ejecución, en la implementación de un programa que realiza la multiplicación de tres matrices cuyos órdenes son dados por el usuario. Puesto que en tiempo de compilación no se conoce el tamaño de la matriz, el espacio se debe reservar en tiempo de ejecución.

Subresultado de aprendizaje	Comentario
1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.	Para el correcto funcionamiento de la práctica, el alumno debe comprender el manejo de memoria en tiempo de ejecución, debe entender el algoritmo para la multiplicación de matrices y debe ser capaz de escibir en código dicho algoritmo. La comprensión de estos temas permitirá al estudiante adquirir conocimientos posteriores tanto de ciencias básicas como de programación.

Lista de cotejo para la evaluación del código:

Criterio	Calificación
Código incompleto (que sea evidente que no va a resolver el problema).	0.0
El código no compila (este punto se omite para códigos en python).	1.0
El código compila pero no funciona (se considera que no funciona si sólo funciona en la mitad de los casos probados o menos).	3.0
El código compila y funciona sólo en algunos casos (funciona en la mitad de los casos probados o más).	6.0
El código compila y funciona en todos los casos.	8.0
El código compila, funciona en todos los casos y está documentado.	10.0

Al alumno se le entrega una hoja con la calificación obtenida y comentarios.

Práctica 5 – Estructuras de datos lineales: pila y cola (expresiones aritméticas en notación postfix y prefix)

Resumen: la práctica consiste en la implementación de una pila utilizando dos colas y la implementación de una cola utilizando dos pilas. La implementación de cada estructura se usa para procesar una cadena de texto en donde el caracter * indica una eliminicación, ya sea desapilar o desencolar, y cualquier otro caracter indica una inserción, ya sea encolar o apilar.

Subresultado de aprendizaje	Comentario
1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.	Para el correcto funcionamiento de la práctica, el alumno debe comprender el funcionamiento de un pila, el funcionamiento de una cola, el uso de cadenas de texto en lenguaje C y debe ser capaz de integrar todos estos conceptos para la solución del problema. Estos conceptos permitirán al alumno entender futuros conceptos de estructura de datos y también comprender el uso de otros lenguajes de programación.

Lista de cotejo para la evaluación del código:

Criterio	Calificación
Código incompleto (que sea evidente que no va a resolver el problema).	0.0
El código no compila (este punto se omite para códigos en python).	1.0
El código compila pero no funciona (se considera que no funciona si sólo funciona en la mitad de los casos probados o menos).	3.0
El código compila y funciona sólo en algunos casos (funciona en la mitad de los casos probados o más).	6.0
El código compila y funciona en todos los casos.	8.0
El código compila, funciona en todos los casos y está documentado.	10.0

Al alumno se le entrega una hoja con la calificación obtenida y comentarios.

Práctica 7 – Estructuras de datos lineales: Lista simple y lista circular (simulación de lista de reproducción)

Resumen: la práctica consiste en la elaboración de un programa que simule el comportamiento de una lista de reproducción en la que se puende insertar y eliminar títulos, cambiar al título siguiente, cambiar al título previo y función de repetir la lista. El programa se implementa con una lista doblemente ligada circular de cadenas de texto que contienen los títulos de la lista.

Subresultado de aprendizaje	Comentario
1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.	Para el correcto funcionamiento de la práctica, el alumno debe comprender el comportamiento de una lista doblemente ligada circular, apuntadores, manejo de cadenas de texto y manejo de memoria en tiempo de ejecución. Estos conceptos permitirán al alumno entender futuros conceptos de estructura de datos y también comprender el uso de otros lenguajes de programación.

Lista de cotejo para la evaluación del código:

Criterio	Calificación
Código incompleto (que sea evidente que no va a resolver el problema).	0.0
El código corre pero no funciona (se considera que no funciona si sólo funciona en la mitad de los casos probados o menos).	3.0
El código corre y funciona sólo en algunos casos (funciona en la mitad de los casos probados o más).	6.0
El código compila y funciona en todos los casos.	8.0
El código compila, funciona en todos los casos y está documentado.	10.0

Al alumno se le entrega una hoja con la calificación obtenida y comentarios.

Práctica 11 – Estrategias para construir algoritmos (comparación de algoritmos de ordenamiento)

Resumen: la práctica consiste en la programación del algoritmo de ordenamiento QuickSort para ordenar un conjunto de enteros. El usuario pasa los enteros a ordenar como argumentos de línea de comandos y el programa imprime los valores ordenados.

Subresultado de aprendizaje	Comentario
1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.	Para esta práctica el alumno debe comprender el problema del ordenamiento y debe entender la estrategia para construir algoritmos 'divide y vencerás'. Puesto que en otras tareas se implentó el algoritmo MergeSort, con esta práctica el alumno dispone de más ejemplos para comprender el uso de la estrategia divide y vencerás, para poder aplicarla en futuros problemas.
2.2. La capacidad de identificar, formular y resolver problemas de ingeniería en su especialidad; elegir y aplicar de forma adecuada métodos analíticos, de cálculo y experimentales ya establecidos; reconocer la importancia de las restricciones sociales, de salud y seguridad, ambientales, económicas e industriales.	Con esta práctica el alumno compara el desempeño de QuickSort con el desempeño de otros dos algoritmos vistos previamente: Bubble Sort y MergeSort, con lo que aprende a elegir la mejor herramienta para el caso particular del problema del ordenamiento. Además, debe ser capaz de utilizar el lenguaje Python para realizar esta implentación.

Lista de cotejo para la evaluación del código:

Criterio	Calificación
Código incompleto (que sea evidente que no va a resolver el problema).	0.0
El código corre pero no funciona (se considera que no funciona si sólo funciona en la mitad de los casos probados o menos).	3.0
El código corre y funciona sólo en algunos casos (funciona en la mitad de los casos probados o más).	6.0
El código compila y funciona en todos los casos.	8.0
El código compila, funciona en todos los casos y está documentado.	10.0

Al alumno se le entrega una hoja con la calificación obtenida y comentarios.

Práctica 12 – Recursividad (el corte de la viga)

Resumen: La práctica consiste en realizar un programa que resuelva el problema del corte de la viga.

Subresultado de aprendizaje	Comentario
1.2. Conocimiento y comprensión de las disciplinas de ingeniería propias de su especialidad, en el nivel necesario para adquirir el resto de competencias del título, incluyendo nociones de los últimos adelantos.	Para el correcto funcionamiento de la práctica, el alumno debe comprender el concepto de recursividad, la estrategia para construir algoritmos de programación dinámica, y debe manejar el lenguaje Python a un nivel suficiente para poder implementar la solución del corte de la viga.
2.2. La capacidad de identificar, formular y resolver problemas de ingeniería en su especialidad; elegir y aplicar de forma adecuada métodos analíticos, de cálculo y experimentales ya establecidos; reconocer la importancia de las restricciones sociales, de salud y seguridad, ambientales, económicas e industriales.	Para resolver este problema, el alumno debe ser capaz de expresar en forma de algoritmo la solución a un problema básico de las ciencias de la computación: el problema del corte de la viga. Además, debe ser capaz de escribir en código el algoritmo desarrollado. Por otra parte, para poder diseñar el algoritmo, debe comprender la estrategia de Programación Dinámica y plantear correctamente la recursión y las funciones para determinar una solución que además debe ser rápida en tiempo de ejecución.

Lista de cotejo para la evaluación del código:

Criterio	Calificación
Código incompleto (que sea evidente que no va a resolver el problema).	0.0
El código corre pero no funciona (se considera que no funciona si sólo funciona en la mitad de los casos probados o menos).	3.0
El código corre y funciona sólo en algunos casos (funciona en la mitad de los casos probados o más).	6.0
El código compila y funciona en todos los casos.	8.0
El código compila, funciona en todos los casos y está documentado.	10.0

Al alumno se le entrega una hoja con la calificación obtenida y comentarios.