Primer Parcial 2024

Tener en cuenta que:

- La duración máxima del examen es de 3 horas y 30 minutos.
- Para el Ejercicio 1 (Parte Teórica): NO se puede usar material, debe realizarse **con lapicera en papel**, y debe entregarse al docente **antes de comenzar la parte práctica**.
- Para el Ejercicio 2 (Parte Práctica): SI se puede usar material y debe realizar en computadora y debe entregarse por Moodle al finalizar. Para este fin se deberá seguir los siguientes pasos:
  1. Crear una carpeta dentro de su computadora de nombre primerparcial2024. En caso de realizar el parcial en una computadora del laboratorio usar la carpeta Z.
  2. Descargar del Moodle el archivo primer_parcial_2024.py .
  3. Mover archivo a carpeta "primerparcial2024" creado en paso 1.
  4. Abrir el Visual Studio Code y colocar en el menú la opción File->Open ir a la carpeta creada primerparcial2024 y poner Open. Esto debería abrir la carpeta y mostrarle el módulo primer_parcial_2024.py a la izquierda para seleccionar.
- **La prueba es individual y no se permiten entregas fuera de hora.**

Ejercicio teórico

1. Convierta el número binario 010111 a base 16 y explique cuál fue el procedimiento utilizado.
2. Escribir una función en pseudocódigo que, dada una matriz de números y un número, devuelva la posición de dicho número en la matriz. Por ejemplo, dada la matriz [[3,6,4], [12,1,8], [9,13,15]] y el número 9, la función debería retornar (2, 0). Si no se ecuentra dicho número, devolver (-1,-1).
3. Escribir una función en pseudocódigo que, dado un arreglo de números, retorne otro arreglo con todos los números que se repitan. Por ejemplo, dado el arreglo [1,5,7,5,2,1], la función debería retornar [1,5].
4. Escriba las diferencias entre la memoria RAM y la memoria ROM.

Ejercicio Práctico

La Academia de Música, consciente de la importancia de la formación musical integral y deseando apoyar a estudiantes con talento pero de recursos limitados, ha establecido un programa de becas. Este programa busca garantizar que estudiantes calificados puedan acceder a cursos de música sin preocuparse por las restricciones económicas.

El software de gestión de la academia debe permitir la administración eficaz de este programa de becas, contemplando las siguientes necesidades:

- Registro de becas ofrecidas: Permitir el registro de distintos tipos de becas, especificando el nombre, el instrumento, la cantidad de becas disponibles de ese tipo, y el nivel de dificultad (siendo 1: Principiante, 2: Intermedio, y 3: Avanzado).
- Aplicación de estudiantes: Los estudiantes deben poder solicitar becas disponibles, indicando su nombre, la beca a la que aplican y su nivel actual (1: Principiante, 2: Intermedio, y 3: Avanzado).
- Evaluación y asignación de becas: La plataforma facilitará que un comité revise las solicitudes para asignar becas basándose en el nivel del estudiante y el de la beca. Solo se otorgan becas cuando los niveles coinciden y hay disponibilidad.
- Reporte: Generación de informes sobre el número de solicitudes por beca.

En este contexto, se solicita desarrollar un prototipo de aplicación que permita gestionar eficientemente el programa de becas, siguiendo estas consideraciones:

- Tipos de Becas: Utilizar una matriz para registrar los diferentes tipos de becas disponibles, donde cada fila represente un tipo de beca y las columnas contengan el nombre, nivel, instrumento, y el número de becas disponibles. Un ejemplo sería: [[“Beca piano clásico para principiantes”, 1,”Piano”, 2]]
- Registro de Aplicaciones: Mantener una matriz de aplicaciones de estudiantes, donde cada fila represente una aplicación y las columnas contengan nombre del estudiante, nombre de beca aplicada, el nivel actual del alumno. Un ejemplo sería: [[“Juan Perez”, “Beca piano clásico para principiantes”, 1]]

Las operaciones a implementar en el módulo gestion_becas_2024.py incluyen:

- registrar_beca(tipos_becas, nombre, nivel, instrumento, cantidad_disponible):
  - Agrega una nueva beca al sistema, agregandolo a la matriz tipos_becas.
  - En caso de que algunos de los parámetros sea inválido (nombre de la beca o instrumento vacíos, el nivel de la beca no es 1,2 o 3, cantidad disponibles es menor a 1) retorna -1.
  - Se debe verificar que no haya una beca con el mismo nombre en el sistema, en caso de que exista retorna -1.
  - En caso de que se haya agregado exitosamente, se retorna 1.
- aplicar_beca(mat_aplicaciones, tipos_becas, nombre_estudiante, nombre_beca, nivel_estudiante):
  - Registra la aplicación de un estudiante a una beca específica en la matriz mat_aplicaciones.
  - En caso de que algunos de los parámetros sea inválido (nombre del estudiante vacío, no existe la beca en la matriz tipos_becas o el nivel del estudiante no es 1,2 o 3) retorna -1.
  - En caso de se haya agregado exitosamente retorna 1.
- asignar_becas(tipos_becas, mat_aplicaciones, nombre_alumno):
  - Evalúa y asigna la beca al alumno. Para ello, evalúa el nivel del alumno y el nivel de la beca. Si el nivel del alumno es igual al nivel de la beca, entónces se otorga. Además, se debe considerar la cantidad de becas disponibles. Si es 0, no se otorga.

  - En caso de que se otorgue la beca, se debe eliminar la aplicación de dicho alumno de la matriz mat_aplicaciones y descontar la cantidad de becas disponibles de dicho tipo.
  - Retorna 1 en caso de que se haya otorgado con éxito.
  - En caso de que no se encuentre al alumno en la matriz de aplicaciones o no se otorgue la beca, se retorna -1.
- Cantidad_aplicaciones_a_beca(mat_aplicaciones, nombre_beca):
  - Retorna la cantidad de aplicaciones que hay para dicha beca.
  - Retorna -1 si no se encontró la beca en la matriz.