| ***UTN – FR Mar del Plata – TUP***  **Laboratorio de Computación II**  **Segundo Parcial** Noviembre 2023 | Nombre y Apellido: | Nota: |
| --- | --- | --- |

**IMPORTANTE PARA APROBAR**

1. El programa debe compilar sin errores.

2. Agregue comentarios identificando cada inciso realizado.

3. El usuario debe poder ingresar información e interactuar con el sistema.

La organización de TEDx Mar del Plata nos solicita un aplicativo para gestionar el manejo de sus entradas y la concurrencia el día del evento.

| **Estructura compuesta** | | |
| --- | --- | --- |
| typedef struct {  char categoria[20];  nodoEntrada \* entradas;  }stBoleteria; | typedef struct {  stEntrada unaEntrada;  struct nodoEntrada \*sig;  }nodoEntrada; | typedef struct {  int idTicket;  int isVIP;  char sector[50];  }stEntrada; |

| **Fila** | |
| --- | --- |
| typedef struct  {  nodoFila \* ultimo;  nodoFila \* primero;  }fila; | typedef struct {  char nombreParticipante[50];  stEntrada entradaAsignada;  struct nodoFila \*sig;  }nodoFila; |

| Valor | Inciso |
| --- | --- |
| 30 | 1. En la primera etapa del evento los organizadores van a cargar las boleterías con sus respectivas entradas disponibles. Codificar las funciones necesarias para que puedan dejar el arreglo de listas cargado. El punto de control para ellos es poder visualizar toda la estructura compuesta. Realizar las funciones necesarias para mostrar todas las boleterías con sus entradas. El idTicket es un super contador general automático.   Ej de categorías para la carga: clásica, premium, deluxe, estudiante.  Ej de sectores para la carga: palco, pullman, campo, platea. |
| 40 | 1. Una vez que todas las boleterías se cargaron, es el turno de empezar a vender las mismas. A un participante se le pregunta primero el nombre como para decirle “hola don pedro pascal, como anda usted el dia hoy?. luego de que categoría desea la entrada, el sector, si es vip o no. Verificamos que haya una entrada disponible con una función buena, bonita y barata. Si la función nos da el ok debemos agregar a la fila (habrá una super fila para todo el mundo) la entrada correspondiente junto con el nombre del participante. Además, como la entrada se asignó, debemos borrar el nodo del arreglo de listas del punto uno. Entonces para resumir    1. FN que pregunta todo al participante    2. FN que verifica si hay una entrada disponible    3. FNs para agregar a la fila    4. FN para borrar un nodo de una lista |
| 10 | 1. Una vez que las entradas dejan de venderse (esto es decisión del usuario, no una cuestión de disponibilidad), los organizadores desean ver como quedo la fila. Seguramente la función ya debería estar codificada para probar el punto anterior. Si no está hecha, mal ahí. Hay que hacerla. |
| 20 | 1. ¡Llegó el día del evento! La gente empieza a aplaudir porque las puertas del estadio están cerradas y cae en la ciudad granizo del tamaño de pelotas de golf. Las personas que no sufrieron lesiones entran finalmente al evento. Al ingresar se eliminan de la fila no si antes pasar por los controles. Cuando ya no hay gente en la fila, los organizadores nos preguntan ¿che, cuanta gente entró?    1. FN para vaciar vila    2. FN para eliminar un elemento (ustedes sabrán cual)    3. Al finalizar el proceso del punto A, deben saber cuántas personas ingresaron efectivamente al estadio. NO vale contar al momento de ingresarlas a la fila. Repito, no contar al ingresar sino al egresar. recuerden que vinieron los paramédicos y se llevaron a un par\* |

**\*** ninguna persona o ser sintiente fue lastimado en la realización de este enunciado. todas las personas involucradas estan sanas y salvas.