Pregunta **1**

Sin responder aún

Puntúa como 1,00

El presente examen consiste en implementar una versión simplificada del conocido juego "**Buscaminas**". El mismo resulta en un tablero minado, donde el jugador debe ir explorando las celdas siendo su objetivo descubrir todas aquellas que no tienen explosivos. Si el jugador descubre una posición en la que hay una mina, pierde la partida. En el contexto de esta implementación se representará el tablero del Buscaminas con 2 matrices. Una de tipo entero que contendrá la información de las posiciones minadas y otra de tipo carácter que almacenará la representación del tablero que se le mostrará al usuario. En la matriz de tipo entero, almacenaremos el valor 1 cuándo la posición esta minada y 0 en caso contrario. Mientras que en la matriz de tipo carácter almacenaremos '#' para referirnos a posiciones aún no exploradas por el usuario, '_' para aquellas posiciones exploradas y vacías y '*' para las posiciones exploradas que contengan bombas.

Sugerencias:

1. Lea todo el examen para formarse una idea global del problema a resolver.
2. Lea detenidamente cada funcionalidad a implementar y válgase de los ejemplos brindados para generar su solución.
3. Para el desarrollo de la solución utilice la siguiente plantilla C++ como punto de partida:

```
#include <stdlib.h>
#include <iostream>
using namespace std;
const int FIL = 3;
const int COL = 3;
```

**Realizar un programa en C++ que**

**1) (55 puntos) incluya las siguientes funciones:**

**a)** (5 puntos) int contarPosVacias(int tablero[FIL][COL])
Dada una matriz de entero debe contar la cantidad de posiciones vacias, es decir aquellas que tengan almacenado el valor 0 y retornar dicho valor.

Ejemplo:

Sea t={0,0,0,1,0,0,0,1,0}, la invocación contarPosVacias(t) debe retornar 7.

**b)** (5 puntos) void mostrarTablero(char mascara[FIL][COL])

Dada una matriz de caracteres esta función debe imprimirla por pantalla con forma de matriz y dejando dos espacios entre valores de una misma fila. El ejemplo de ejecución muestra el resultado esperado.

Ejemplo:

Sea m = {'_','#','#','#','*','#','#','_','#'}, mostrarTablero(m) deberá imprimir por pantalla:

```
_ # #
# * #
# _ #
```

**c)** (12 puntos) void minarTablero(int tablero[FIL][COL] , int bombas)
Dada una matriz de enteros y un numero entero n, el objetivo de esta función es colorar n bombas en el tablero en celdas elegidas de manera pseudo-aleatoria. Una bomba en la fila i columna j del tablero, se simboliza almacenando el valor 1 en la posición [i,j]. Para obtener una fila y columna al azar debe valerse de la función rand() implementada en la librería stdlib. Para ello, considere que la expresión rand() % 10 resultará en un entero pseudo-aleatorio perteneciente al intervalo [0,10). Para elegir un número de fila(resp. columna) de manera pseudo-aleatoria usted debería utilizar rand() % FIL (resp. COL).
No se requiere validar que la posición a minar no haya sido minada previamente. Es decir el tablero puede quedar minado con menos de n bombas.

Ejemplo:

Sea t = {0,0,0,0,0,0,0,0,0} la invocación minarTablero(t,2) debe colocar dos bombas en posiciones elegidas de manera pseudo-aleatoria. Un posible valor de t luego de la ejecución podría ser: t={0,0,0,1,0,0,0,1,0}.

**d)** (8 puntos) void realizarJugada(int tablero[FIL][COL], char mascara[FIL][COL], int i, int j)
Dada dos matrices y dos enteros simbolizando una posición de fila y columna esta función debe descubrir la posición i,j de la matriz. Para ello debe evaluar el valor de la matriz tablero y asignar el valor correspondiente como se indica a continuación:

- Si tablero en i,j contiene 1 (esta minado) debe almacenar '*' en la matriz mascara.
- Si  tablero en i,j contiene 0 (no esta minado) debe almacenar '_' en la matriz mascara.

Ejemplo:
Sea t=[0,1,0,0,1,0,0,0,0], m = {'_','#','#','#','#','#','#','_','#'}:

- Luego de la invocación realizarJugada(t,m,0,2) el valor de m debería ser  {'_','#','_','#','#','#','#','_','#'}.
- Luego de la invocación realizarJugada(t,m,1,1) el valor de m debería ser {'_','#','#','#','*','#','#','_','#'}.
- Luego de la invocación realizarJugada(t,m,0,0) el valor de m debería ser {'_','#','#','#','#','#','#','_','#'} (el mismo).

Notar que no debe modicarse el valor de t en ningún caso.

**e)** (10 puntos) bool jugadorPerdio(char mascara[FIL][COL])
Dada una matriz de caracteres, debe determinar si el jugador ha perdido la partida. Esto sucede cuándo existe una posición con el valor '*', que simboliza que una posición minada ha sido descubierta. Por tanto este método debe retornar true si efectivamente existe una posición con dicho valor o false en caso contrario.

Ejemplo:

- Sea m={'_','#','#','#','#','#','#','_','#'}, jugadorPerdio(m) debe retornar false.
- Sea m={'_','#','#','#','*','#','#','_','#'} jugadorPerdio(m) debe retornar true.

**f)** (15 puntos) bool jugadorGano(char mascara[FIL][COL], int aDescubrir)
Dada una matriz de caracteres y un entero que indica la cantidad de posiciones que se deben descubir para que el jugador gane la

partida, este método debe determinar si el jugador ganó. El jugador gana la partida cuándo el número de posiciones vacías descubiertas(aquellas que tienen almacenado el valor '_') es igual al valor aDescubrir, por lo tanto si ese es el caso el método debe retornar true. En caso contrario false.

Ejemplo:

- Sea m={'_','#','_','_','_','#','#','_','#'}, jugadorGano(m,7) debe retornar false.
- Sea m={'_','#','#','#','*','#','#','_','#'} jugadorGano(m,7) debe retornar false.
- Sea m={'_','_','_','_','#','_','_','_','#'} jugadorGano(m,7) debe retornar true.

**2) (45 puntos) ejecute las siguientes acciones dentro de la función principal (main)**

**a**) (1 punto) Defina una matriz de enteros de FIL filas y COL columnas con identificador tab.
**b**) (1 punto) Defina una matriz de caracteres de FIL filas y COL columnas con identificador masc.
**c**) (5 puntos) Inicialice todos las posiciones de tab con el valor 0 y las de masc con '#'.
**d**) (3 puntos) Defina una variable entera con identificador nBombas, solicite al usuario que ingrese un valor y almacenelo en nBombas.
**e)** (32 puntos) Si el valor almacenado en nBombas es mayor a 0 y menor que FIL*COL ejecute los siguientes pasos:
e1) (3 puntos) Invoque la función minarTablero con tab y nBombas.
e2) (3 puntos) Defina una variable de tipo entero con identificador posADescubir y asignele el valor resultante de la invocación a contarPosVacias con tab.
e2) (3 puntos) Invoque a la función mostarTablero con masc.
e3) (17 puntos) Implemente el ciclo del juego. Mientras el fin del juego no haya sido alcanzado(*), el programa deberá realizar las siguientes acciones:
i) (8 puntos) Solicite que se ingrese la posición a jugar, es decir solicitar al usuario dos números enteros que representarán la fila y columna a descubrir. Deberá validar que sean posiciones válidas, es decir mayores que 0 y menores que FIL (resp.COL) y volver a solicitar si es incorrecta.
ii) (3 puntos) invoque a la función realizarJugada con tab, masc y los valores obtenidos en el paso anterior.
iii) (3 puntos) invoque a la función mostarTablero con masc.
+3 puntos por implementar todo el ciclo del juego de manera correcta
(*) Note que el final del juego será alcanzado cuándo valga la siguiente condición: jugadorGano(masc,posADescubir) || jugadorPerdio(masc)
e4) (1 punto) Imprima el mensaje "Juego Terminado!"
e5) (5 puntos) Si el jugador ganó imprima "Usted gano la partida." en caso contrario imprima "Usted ha perdido la partida.". Recuerde que para averiguar si el jugador gano o perdio puede recurrir a las funciones jugadorGano y/o jugadorPerdio.

**f)** (3 puntos) En caso contrario imprima el mensaje: "El número de bombas debe ser mayor a 0 y menor que A" donde A debe ser remplazado por el valor de FIL*COL.

**EJEMPLOS:**

**1)**
Ingrese la cantidad de bombas:
-1
El número de bombas debe ser mayor a 0 y menor que 9

**2)**
Ingrese la cantidad de bombas:
9
El número de bombas debe ser mayor a 0 y menor que 9

**3)**
Ingrese la cantidad de bombas:
2
# # #
# # #
# # #
Ingrese Fila y Columna: -1 2
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 2 -1
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: -2 -2
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 0 0
_ # #
# # #
# # #
Ingrese Fila y Columna: 0 -1
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 2 2
_ # #
# # #
# # _
Ingrese Fila y Columna: 2 3
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 1 1
_ # #
# * #
# # _
Juego Terminado!
Usted ha perdido la partida

Usted ha perdido la partida.

**4)**

Ingrese la cantidad de bombas:
2
# # #
# # #
# # #
Ingrese Fila y Columna: 2 3

Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 3 2
Jugada no permitida. Vuelva a ingresar!
Ingrese Fila y Columna: 0 0
_ # #
# # #
# # #
Ingrese Fila y Columna: 0 2

_ # _
# # #
# # #
Ingrese Fila y Columna: 1 0

_ # _
_ # #
# # #
Ingrese Fila y Columna: 2 2

_ # _
_ # #
# # _
Ingrese Fila y Columna: 2 0

_ # _
_ # #
_ # _
Ingrese Fila y Columna: 2 1

_ # _
_ # #
_ _ _
Ingrese Fila y Columna: 1 2

_ # _
_ # _
_ _ _
Juego Terminado!
Usted gano la partida.

Tamaño máximo para nuevos archivos: 1GB, número máximo de archivos adjuntos: 1

Archivos

Puede arrastrar y soltar archivos aquí para añadirlos

## Historial de respuestas

| Paso | Hora | Acción | Estado | Puntos |
|---|---|---|---|---|
| 1 | 24/10/2019 08:37 | Iniciado/a | Sin responder aún | |