Segundo examen parcial
Computacional II
Nombre alumno:

Resuelva el siguiente ejercicio:

a) Diseñe una clase llamada "Circular" con los datos para el radio del circulo en el plano xy, la frecuencia angular de la particula que se mueve en el circulo, un pequeño intervalo de tiempo, el tiempo total que durara el movimiento del cuerpo, el numero de intervalos de tiempo tenidos en cuenta y una fase. Tambien deben aparecer el constructor y el destructor de la clase, las funciones miembros "xpos" y "ypos" que calculan las posiciones xy dado el tiempo. Las siguientes son las ecuaciones que debe codificar para xpos y ypos:

$xpos = Rcos(wt + \alpha)$
$ypos = Rsin(wt + \alpha)$

b) Ahora diseñe la clase "Expiral", que sera derivada de la clase Circular, para que herede el movimiento circular y adicione el movimiento en la direcion del eje z, de acuerdo con la ecuacion:

$z = z_0 + v_z * t$

Resuelva almenos uno de los siguientes ejercicios:

1. El problema del caballo en el ajedrez. ¿puede la pieza de ajedrez, conocida como caballo, moverse alrededor de un tablero de ajedrez vacío y tocar cada una de las 64 posiciones una y sólo una vez? Por favor desarrolle un algoritmo en c++ para estudiar este problema.

2. El problema del las reinas en el ajedrez.
   ¿Es posible colocar ocho reinas en un tablero de ajedrez vacío, de tal manera que ninguna reina “ataque" a cualquier otra (es decir, que no haya dos reinas en la misma fila, en la misma columna o a lo largo de la misma diagonal)?

   Considerando la “jerga" del curso: resuleva este problema, por favor desarrolle un algoritmo en c++ para estudiar este problema. Dividalo en una definicion de clase (file.h), una interfaz(implementación) y el codigo principal (main).

3. Laberinto.

   a) La cuadrícula que contiene caracteres # y puntos (.) en la Figura 1 es una representación de un laberinto mediante un arreglo bidimensional. En este arreglo bidimensional, los caracteres # representan las paredes del laberinto, y los puntos representan las ubicaciones en las posibles rutas a través del laberinto. Sólo pueden realizarse movimientos hacia una ubicación en el arreglo que contenga un punto.

      Hay un algoritmo simple para recorrer un laberinto, que garantiza encontrar la salida (suponiendo que la haya). Si no hay salida, el algoritmo lo llevará a la ubicación inicial de nuevo. Coloque su mano derecha en la pared a su derecha y empiece a caminar hacia adelante. Nunca quite su mano de la pared. Si el laberinto gira a la derecha, siga la pared

a la derecha. Mientras que no quite su mano de la pared, en un momento dado llegará a la salida del laberinto, siempre y cuando el laberinto no tenga caminos circulares y empiece en uno de ellos . Puede haber una ruta más corta que la que usted haya tomado, pero se garantiza que saldrá del laberinto si sigue el algoritmo. No es muy util en laberintos reales demasiado grandes, por la cuestion del tiempo, pero en una computadora es mas que aceptable.

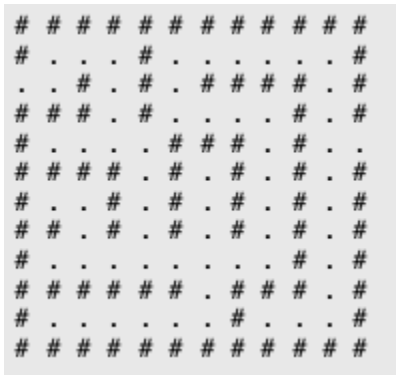

Figure 1: Representación de un laberinto mediante un arreglo bidimensional.

Escriba una función recorrerLaberinto para avanzar a través del laberinto. La función debe recibir como argumentos un arreglo de caracteres de 12 por 12 que representa el laberinto, y la posición inicial en el mismo. A medida que recorrerLaberinto trate de localizar la salida, debe colocar el carácter x en cada posición en la ruta. La fun- ción debe mostrar el laberinto después de cada movimiento, de manera que el usuario pueda observar a medida que se va resolviendo.

b) (Generación de laberintos al azar, opcional). Escriba una función llamada generarLaberintos , que reciba como argumento un arreglo bidimensional de 12 por 12 caracteres, y que produzca un laberinto al azar. Esta función también deberá proporcionar las posiciones inicial y fi nal del laberinto. Pruebe su función recorrerLaberinto del ejercicio 8.25, usando varios laberintos generados al azar.