UNIVERSIDAD NEBRIJA

PROGRAMACIÓN I - GRUPO C

PRÁCTICA Nº4

5 de diciembre 2023

INSTRUCCIONES

1. Se debe acceder a la **SALA DE GRUPO** del campus virtual de vuestro equipo de trabajo, hay una sala para cada integrante.

2. Durante el desarrollo de la sesión práctica se debe seguir el procedimiento de trabajo en la sala asignada para realizar las actividades evaluables. Recordad que es **imprescindible GRABAR toda la sesión** y **COMPARTIR la pantalla completa**. Esto debe hacerlo cada integrante del equipo

3. La entrega de la práctica sólo se admitirá a través de la **actividad** disponible en el campus virtual de la asignatura de **Prácticas de Programación I antes de la hora de finalización de la sesión de prácticas**.

4. Se debe entregar **un único fichero** en **formato .txt o .cpp, sin comprimir**, con el nombre de ***P4GC.***

5. El fichero entregado debe **incluir el nombre de los integrantes del equipo**.

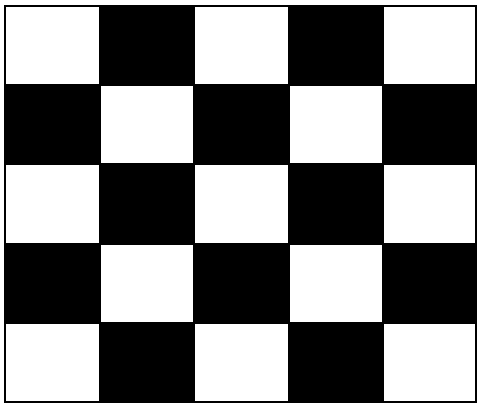
6. Aunque las prácticas se realizan en grupos de dos integrantes, para su evaluación, **ambos deberán hacer la entrega a través de su campus virtual**. En otro caso, la práctica quedará **sin evaluar** y supondrá un **0 en su calificación**.

7. **El incumplimiento de alguna de las instrucciones sobre la realización/entrega de la Práctica supondrá su descalificación.**

A. EJERCICIO PRÁCTICO [70% puntuación] – 160 min

Escriba un programa en C++11, ***P3GC***, utilizando los **tipos de datos, las operaciones de entrada y salida, las estructuras de control** vistas en clase y **las funciones** adecuadas para estructurar bien el simulador del ***juego Saltarranas*** en un tablero (charca) de **5x5**. El juego consiste en eliminar todas las ranas situadas en la *charca* (tablero) hasta que **sólo quede una**.

El tablero tendrá posiciones no utilizables que siempre serán las mismas y corresponden a posiciones alternas en cada fila del tablero. La figura representa el tablero y los recuadros **negros** son los que **nunca se pueden utilizar**:



La ranas o fichas se representarán con la letra "***R***", las posiciones libres y utilizables como un espacio en blanco y las posiciones no utilizables con una "***X***".

Existirán 5 retos para jugar. Los posibles retos se tienen que almacenar en una variable tipo contenedor donde se indica las posiciones que deben ocupar las ranas en el tablero. Para comenzar a jugar se realizará la **elección del reto de manera aleatoria.** Los retos que se deben de cargar para realizar la selección son los siguientes:

Reto nº1: (0, 2), (0, 4), (2, 2), (2, 4)

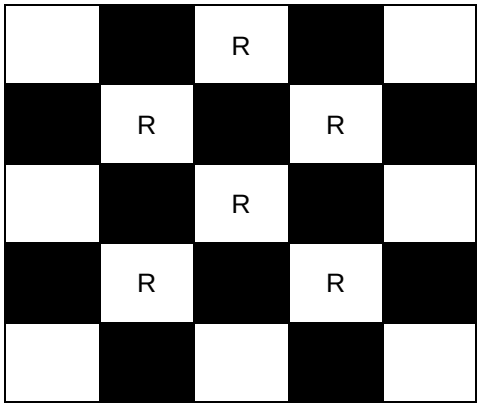
Reto nº2: (0, 2), (1, 1), (1, 3), (2, 2), (3, 1), (3, 3)

Reto nº3: (0, 2), (0, 4), (2, 2), (2, 4), (4, 2)

Reto nº4: (0, 2), (2, 0), (2, 4), (4, 0), (4, 2)

Reto nº5: (0, 4), (1, 1), (1, 3), (2, 4), (3, 1)

Por ejemplo, si se selecciona el reto nº1, el tablero sobre el que se juega sería:



Ayuda para la selección aleatoria del reto:

1. Incluir las siguientes librerías:
   ```
   #include <cstdlib>
   #include <ctime>
   ```
2. Inicializar la semilla del generador de números aleatorios con:
   ```
   srand (time(NULL));
   ```
3. Utilizar la función `rand()%(n+1)` para generar números aleatorios entre 0 y n.

Reglas del juego:

1. Seleccionar de manera aleatoria uno de los cinco retos propuestos.
2. Situar las ranas sobre la charca (tablero de juego) como se indica en la propuesta del reto seleccionada.
3. Solicitar al jugador las dos posiciones de su jugada:

   (1) **Posición de la rana que va a saltar**. El jugador debe indicar **fila y columna** para comprobar que **existe una rana** en dicha posición. En caso contrario, se debe mostrar un **mensaje informativo** al usuario.

   (2) **Nueva situación para la rana que salta**. El jugador debe indicar **fila y columna** de la nueva ubicación para comprobar que es correcta según las reglas del juego que se indican. En caso contrario, se debe mostrar un **mensaje** informativo al usuario.

   Cualquier rana puede comer a su vecina saltando por encima de ella en diagonal, vertical u horizontalmente, situándose inmediatamente en la siguiente posición libre según la dirección del salto. Esto hace que la rana sobre la que se salta se **elimine** del tablero de juego.

Si cualquiera de las posiciones indicadas por el jugador corresponde a una posición marcada con "***X***" (negra), se debe de mostrar un **mensaje** informativo al usuario.

4. Sólo se pueden saltar una rana cada vez.
5. La dificultad del juego consiste en no dejar ranas aisladas, pues éstas, se convierten en inalcanzables y la solución al desafío es imposible.
6. El **reto se resuelve o termina** cuando únicamente te queda **una rana sobre la charca o tablero**.
7. Se preguntará al jugador si desea enfrentarse a un nuevo reto o finalizar el juego.

Sugerencias de elementos a utilizar para la implementación del juego:

1. **Retos:** será un **array de vectores** para los 5 restos. Los vectores incluirán las posiciones que deben ocupar las ranas que estarán guardadas en elementos de tipo ***struct*** para almacenar tanto la fila y como la columna de cada posición.
2. Función que almacene los retos en el array de retos que se declarará en la función principal (***main***). No tendrá parámetros y devolverá el array de vectores con los retos indicados en el enunciado.
3. Función que inicialice el tablero. Esta función no tendrá parámetros y devolverá el tablero de juego con las posiciones de los cuadrados negros ('***X***') y posiciones libres donde pueden estar o saltar las ranas.
4. Función que seleccione el número del reto con el que se jugará de manera aleatoria y prepare el tablero de juego colocando las ranas según se indique en el reto. Esta función recibirá el tablero de juego inicializado y el array de vectores con los retos, además devolverá el número de ranas que se colocarán en el tablero según el reto seleccionado. Una vez elegido el reto se tendrán que colocar las ranas en el tablero de juego ('***R***').
5. Función mostrar tablero que permitirá mostrar el estado actual del tablero de juego. Esta función recibirá el tablero de juego y lo mostrará con los caracteres elegidos para representar el contenido del tablero: ' ', '***X***', '***R***'.
6. Función jugar que solicitará al jugador las dos posiciones necesarias, la primera para saber con qué rana se juega y, la segunda, para indicar dónde se quiere colocar la rana después del salto.

Se deben hacer las validaciones adecuadas para cada una de ellas y se pueden hacer a través de otras funciones:

- La posición de partida de la rana que salta sólo puede contener el carácter 'R'. Habrá que validar que está dentro de los límites del tablero y que no se trata de un espacio vacío (' ') ni de un cuadrado negro ('***X***').
- La posición de destino de la rana que salta siempre debe ser un espacio libre del tablero (' '). Habrá que validar que está dentro de los límites del tablero y que no se trata de una posición ocupada por una rana ('***R***') ni de un cuadrado negro ('***X***'). Además, en este caso, hay que realizar las validaciones correspondientes al posible salto, excluyentes entre ellas:
    - Comprobar si se trata de un salto horizontal y si es válido.
    - Comprobar si se trata de un salto vertical y si es válido.
    - Comprobar si se trata de un salto diagonal (hacia la derecha o hacia la izquierda) y si es válido.

Si el salto es correcto, se habrá eliminado una de las ranas del tablero y estaremos más cerca del final.

RUBRICA DE CALIFICACIÓN

- El 30% de la nota de las prácticas, se obtendrá del trabajo en clase (resolución de preguntas, dominio de la materia, etc.). **Esta nota será individual**. Se consideran aspectos como:
  - Defensa **individual** a cuestiones que el/la profesor/a plantea a cada integrante del equipo.
  - Debate en **equipo** sobre la estrategia a seguir, realizando un **boceto de la estructura** que tendrá el programa y cada una de las opciones (a papel, **dicho boceto debe entregarse al final de la práctica**).
- El 70% restante será de la práctica entregada (**misma nota para cada integrante del equipo**), siguiendo la rúbrica de evaluación general y la siguiente rúbrica para cada apartado:

Concepto	Puntos
Escritura correcta del código con las **tabulaciones** adecuadas, **nombres significativos** para las variables, **comentarios**, etc.	0,5 puntos
Creación del **array** de **vectores** con los retos incluidos en el enunciado del ejercicio con elementos de tipo **struct**.	1 punto
Creación del tablero de juego como un array de dos dimensiones (filas y columnas) inicializado correctamente con posiciones libres (“ “) y cuadrados negros (“X”).	0,75 puntos
Inicialización del tablero de juego con de las ranas (“R”) del reto seleccionado.	0,75 puntos
Mostrar el tablero de juego con de las ranas (“R”) del reto seleccionado, las posiciones libres (“ “) y cuadrados negros (“X”).	0,5 puntos
Solicitar la posición de partida de la rana que va a saltar, incluyendo sus validaciones y mensajes.	0,5 puntos
Solicitar la posición destino de la rana que salta, incluyendo sus validaciones y mensajes: - Validaciones de límites de tablero y contenido de posición. - Validación de salto horizontal. - Validación de salto vertical. - Validación salto diagonal derecho. - Validación salto diagonal izquierdo.	2 puntos
Función ***main*** con la lógica adecuada para el juego (llamadas a las funciones implementadas), incluyendo el mensaje que señale el fin del juego y permitiendo al usuario comenzar un nuevo reto o terminar.	1 punto

Además, cada apartado de la parte práctica, se evaluará considerando el siguiente baremo:

	% máx. (*)
El elemento evaluable no compila o no se asemeja a lo que se pide	0%
El elemento evaluable no se aproxima suficientemente a lo pedido	40%
El elemento evaluable se aproxima suficientemente a lo pedido	60%
El elemento evaluable funciona correctamente y las estrategias y elementos de código elegidos son adecuados.	100%

(*) El porcentaje (% max.) representa el valor máximo sobre el que se evalúa el elemento indicado.