EIE
Escuela de Ingeniería Eléctrica

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

UNIVERSIDAD DE COSTA RICA

`IE-0217` Estructuras abstractas de datos y algoritmos para ingeniería

Laboratorio 1: Juego de ecología en C++ (I)

M. Sc. Ricardo Román Brenes - `ricardo.roman@ucr.ac.cr`

II-2018

Tabla de contenidos

---

1. Enunciado

Diseñe, implemente y documente los objetos con sus métodos y atributos para una simulación simple de ecológica.

La simulación se llevará acabo en una arreglo doble de `Celdas` de $N \times M$. En `Celda` habrá un objeto de tipo `Pasto` y puede o no haber un objeto de alguno de los tipos de animal.

Los objetos animal pueden ser uno de cuatro clases: `Lobos`, `Ovejas`, `Zorros` o `Ratones`.

Cada uno de ellos tiene, al menos, una *cantidad de energía*, un *identificador*, un *sexo*, *especie* y una *ubicación* designada como un vector (x, y).

Además, cada uno puede realizar alguno de las siguientes cuatro acciones: `mover`, `comer`, `reproducir`, `morir`. Para el método mover, se debe especificar una posición del `Pasto` y que esta esté vacía (sin animal). Los métodos de `comer` reciben como argumento un objeto dependiendo de quien lo invoque, puede ser `Pasto`, `Ratones` u `Ovejas`. El mensaje de reproducir necesita que contiguo a un animal, encuentre otro, de la misma especie pero de diferente sexo y con una energía mayor al 66 % de su máximo para ambos y debe haber una casilla adyacente a ambos disponible (sin animal).

Los `Animales` tiene diferentes velocidades; al día, un `Lobo` puede moverse 3 posiciones, un `Oveja` y un `Zorro` pueden moverse 2 posiciones y un `Ratón` puede moverse 1 posición.

Cada día que pase, todos los animales deben moverse, `comer`, reproducirse y morir, en caso **de ser posible** y deben hacer cada acción **exactamente una vez al día** y **en ese orden**. Por

ejemplo si un lobo se encuentra en una casilla rodeado por lobas, este sólo se reproducirá una vez y dicha loba no se reproducirá más por ese día.

Los `Animales` y el `Pasto` interactúan bajo las siguientes reglas (considere como adyacentes a un `Pasto` la vecindad de Moore):

1. Los `Lobos` tiene una energía máxima de 100.
2. Las `Ovejas`ene una energía máxima de 75.
3. Los `Zorros` tiene una energía máxima de 50.
4. Los `Ratones` tiene una energía máxima de 25.
5. Al final de cada día, los Animales pierden 5 % de su energía total.
6. Al cabo de 3 días, cada `Pasto` que tenga una cantidad positiva de energía ganará 5 puntos de energía.
7. Un `Pasto` no puede acumular más de 50 puntos de energía.
8. Un `Pasto` con energía 0 no puede perder más energía.
9. Una `Oveja` come energía de un `Pasto`, consume 10 puntos del `Pasto` y recupera la misma cantidad.
10. Un `Ratón` come energía de un `Pasto`, consume 5 puntos del `Pasto` y recupera la misma cantidad.
11. Un `Zorro` come energía de un `Ratón`, lo consume por completo y recupera 2 puntos de energía.
12. Un `Lobo` come energía de cualquier otro animal, lo consume por completo y recupera 10 puntos por cada `Oveja`, 5 por cada `Zorro` y 2 por cada `Ratón`.
13. Un `Lobo` macho que tenga en un `Pasto` adyacente a otro `Lobo` macho causa que alguno de los dos muera.
14. Un `Lobo` que tenga en un `Pasto` adyacente a cualquier otro Animal menos un `Lobo` causa que el `Lobo` consuma al Animal.
15. Un `Zorro` que tenga en un `Pasto` adyacente a un `Ratón` causa que el `Zorro` consuma al `Ratón`.
16. Un Animal que tenga en un `Pasto` adyacente a otro Animal de la misma especie pero de sexo opuesto y contando con otro `Pasto` adyacente a ambos desocupado, se reproducirá y genera un nuevo Animal de la misma especie de cualquier sexo.

El programa que genere recibirá por argumento la cantidad de días por los que se ejecutará la simulación y la ruta de un archivo de configuración que proporcionará la configuración inicial del juego.

Este archivo tiene la siguiente configuración:

```
3                   # M
3                   # N
0   0   25  LH      # celda (0,0); 25e; lobo hembra
0   1   75  CM      # celda (0,1); 75e; conejo macho
0   2   0           # celda (0,2); 00e; sin animales
1   0   100         # ...
1   1   50  LM      # ...
1   2   25  CH      # ...
2   0   75  OM      # ...
2   1   0   OM      # ...
2   2   0   OH      # ...
```

El anterior archivo representaría el siguiente mapa:

	0		1		2	
0	A:	LH	A:	CM	A:	
	en:	25	en:	75	en:	0
1	A:		A:	LM	A:	CH
	en:	100	en:	50	en:	25
2	A:	OM	A:	OM	A:	OH
	en:	75	en:	0	en:	0

Mapa de ejemplo.

Al inicio y al final de la simulación es necesario que imprima en pantalla un reporte de los animales disponibles en el mapa. Para esto construya una **función** que reporte, el identificador, la especie, el sexo y las coordenadas del Animal.

En cada día de ejecución de la simulación imprima el estado del juego de una forma clara en pantalla.

Utilice un Makefile para facilitar la construcción y prueba de su programa.

Recuerde utilizar todas las capacidades vistas de C++ (a menos que sea necesario otra cosa ((C)).

2. Consideraciones

- Haga grupos de hasta 3 personas.
- Genere un reporte en LaTeX que incluya su código, su abordaje para la solución y sus conclusiones.
- Suba su código y documentación (doxygen, README, INSTALL) al git respectivo de su grupo y el directorio del laboratorio.
- Cada estudiante debe subir el reporte a Schoology.
- Recuerde que por cada día tardío de entrega se le rebajaran puntos de acuerdo con la formula: 4^d, donde $d > 1$ es la cantidad de días tardíos.