universidad cenfotec
La U de la informática

Fundamentos de Programación

Tarea 5.
Listas, funciones y procedimientos.

Objetivos

universidad cenfotec
La U de la informática

- Utilizar listas y la abstracción procedimental en la solución de problemas, modelando rutinas, usando diagramas de flujo y el lenguaje de programación Java.

- Utilizar estructuras iterativas, condicionales y secuenciales estudiadas anteriormente.

- Utilizar las expresiones lógicas vistas en el curso.

Enunciado

Problema:

Se quiere hacer un programa en Java para jugar "*Battleship*" o Batalla Naval, simplificado. Este juego consiste en que el usuario pueda jugar contra la computadora y gana quien logre derribar todos los barcos del equipo contrario o quien tenga mejor puntaje.

Para implementar el juego tanto el usuario como la computadora cuentan con una lista de enteros, de 20 posiciones cada uno, llamados **`barcosJugador`** y **`barcosComputadora`**, que contendrán la información de los barcos. Cada barco se simboliza con un número del 1 al 5, así estas listas en cada una de sus posiciones contienen la siguiente información: 0 si no hay barco, o un número entre 1 y 5 para indicar que en esa posición se encuentra uno de los cinco barcos, por ejemplo:

`barcosJugador`:

1	0	0	0	0	2	0	0	3	0	0	4	0	0	0	0	5	0	0	0

`barcosComputador`:

0	0	0	2	0	5	0	0	0	0	0	0	0	1	0	0	3	0	0	4

Enunciado

universidad cenfotec
La U de la informática

Problema (continuación):

¿Cómo colocar barcos en las listas?

Los barcos del jugador se colocan en forma manual, el usuario indica en qué posiciones quiere colocar sus 5 barcos dentro de la lista correspondiente. La computadora coloca sus 5 barcos de manera aleatoria (para esto se usa la rutina estándar **`random`**) dentro de la lista correspondiente.

¿Cómo derribar los barcos del contrincante?

Un vez colocados los barcos, se inicia el juego con 10 oportunidades que tienen tanto el jugador como la computadora para derribar los barcos de su contrario o acumular puntos. En cada intento primero inicia el jugador, el cual indica en qué posición cree que hay un barco, el programa debe indicar si derribó un barco o si fue un intento fallido. Luego será el turno de la computadora que de manera aleatoria (para esto se usa la rutina estándar **`random`**) indica en qué posición cree que hay un barco, el programa debe indicar si derribó un barco o si fue un intento fallido.

¿Cómo se gana el juego?

Si antes de las 10 oportunidades, el jugador o la computadora derriban los 5 barcos de su contrincante, entonces gana. Si ninguno lo hace, entonces gana quien haya encontrado o derribado más barcos. Pero si tienen igual cantidad de barcos derribados, el ganador es quien tenga la mayor suma de los números de los barcos derribados. Si la suma es igual entonces se declara empate.

Enunciado

universidad cenfotec
La U de la informática

Problema (continuación):

Para desarrollar el programa se debe cumplir con los siguientes requerimientos:

1. **Rutina `inicializarBarcosComputadora` (15%):** se debe crear una rutina que coloque en forma aleatoria los 5 barcos (números del 1 al 5 en cinco diferentes índices de una lista de enteros). En las posiciones donde no hay barcos debe asignarse el valor 0.
2. **Rutina `inicializarBarcosJugador` (15%):** se debe crear una rutina que permita al jugador digitar cinco índices para colocar en ellos los 5 barcos (números del 1 al 5 en cinco diferentes índices de una lista de enteros). En las posiciones donde no hay barcos debe asignarse el valor 0.
3. **Rutina `imprimirBarcos` (10%):** se debe crear una rutina que imprima el contenido de una lista de enteros.
4. **Rutina `hacerDisparo` (10%):** se debe crear una rutina que recibe una lista de enteros y una posición dentro de esa lista, y retorna el número del barco derribado o -1 si no se derribó ningún barco. Si se derribó un barco, la rutina debe asignar 0 en la posición donde estaba el barco.

Enunciado

Problema (continuación):

Para desarrollar el programa se debe cumplir con los siguientes requerimientos:

5. **Rutina `imprimirGanador` (15%)**: se debe crear una rutina que recibe el número de barcos derribados del jugador, el número de barcos derribados de la computadora, la suma de los barcos del jugador y la suma de los barcos de la computadora, e imprima quién es el ganador o si hay un empate, de acuerdo con las reglas mencionadas.
6. **Rutina `imprimirMenu` (5%)**: se debe crear una rutina que imprima un menú con las siguientes opciones: 1. **Inicializar el Juego** (posicionar los barcos del jugador y de la computadora), 2. **Imprimir dónde están los barcos** (del jugador y de la computadora), 3. **Jugar** (el jugador contra la computadora) , 4. **Salir del juego**.
7. **Programa principal (30%)**: Se debe hacer un programa que imprima el menú de opciones y ejecute cada una de las opciones que el usuario escoja, mientras que la opción sea diferente de salir. En la opción 1 se deben usar las rutinas para inicializar los barcos de la computadora y del jugador. En la opción 2 se debe usar la rutina para imprimir los barcos de la computadora y del jugador. En la opción 3 se debe simular el juego entre el jugador y la computadora hasta que se llegue a 10 intentos o hasta que uno de los dos gane el juego. En la opción 3 se debe usar la rutina de hacer disparo y al final del juego la de imprimir ganador. Suponga que siempre la opción 1 siempre se ejecuta antes que cualquiera de las demás opciones.

Rúbrica

universidad cenfotec
La U de la informática

Problema: 100%

Se calificarán únicamente los siguientes rubros con su respectivo porcentaje:

1. **Rutina inicializarBarcosComputadora (15%):**
    - Rutina en Java (que compile y se ejecute correctamente): 15%
2. **Rutina inicializarBarcosJugador (15%):**
    - Rutina en Java (que compile y se ejecute correctamente): 15%
3. **Rutina imprimirBarcos (10%):**
    - Rutina en Java (que compile y se ejecute correctamente): 10%
4. **Rutina hacerDisparo (10%):**
    - Rutina en Java (que compile y se ejecute correctamente): 10%
5. **Rutina imprimirMenu (5%):**
    - Rutina en Java (que compile y se ejecute correctamente): 5%

Rúbrica

(Continuación)

6. **Rutina imprimirGanador (15%):**
    - Tabla de entradas y salidas: 2%
    - Diagrama de flujo final: 3%
    - Rutina en Java (que compile y se ejecute correctamente): 10%
7. **Programa principal: 30%**
    - Tabla de entradas y salidas: 3%
    - Diagrama de flujo final: 7%
    - 1 caso de prueba completo en la documentación: 5%
    - Programa en Java (que compile y se ejecute correctamente): 15%

Entrega

universidad cenfotec
La U de la informática

Usted debe entregar lo siguiente:

- **Un solo archivo PDF** con la documentación correspondiente al problema. Si lo cree necesario, puede crear los diagramas usando Draw.io y adjuntarlos en la entrega. Puede incluir el código Java en la documentación o entregarlo aparte.
- **Un archivo .java,** del problema.
- Cada archivo **debe** tener el siguiente formato:
  - NombreEstudiante_Tarea5.pdf
  - NombreEstudiante_Tarea5.java
  - ❖ (*)NombreEstudiante_Tarea5_diagrama.drawio
  - Ejemplo:

    AlanTuring_Tarea5.pdf

    AlanTuring_Tarea5.jav

    ❖ (*)AlanTuring_Tarea5_diagrama.drawio
- **Usted debe hacer la entrega en Moodle en la semana correspondiente, antes de la fecha de entrega a las 23:59.**
- **El archivo de Draw.io marcado con (*) es opcional, su uso es solamente si usted lo considera necesario.**

Anexo: `random.randint()`

universidad cenfotec
La U de la informática

`random.randint()` es una función que retorna un número positivo de manera aleatoria, de tipo *entero*, entre un rango establecido.

Esta rutina retorna un número real positivo de tipo *double*, pero únicamente entre 0.0 y 1.0, si se quiere un número aleatorio de mayor valor, debe multiplicarse por 10, 100, 1000 o hacer la operación aritmética que se requiera para conseguir el valor dentro de los rangos que se necesiten. Vea los siguientes ejemplos:

```
import random

numero = random.randint(0, 100)

# Imprime un número entero aleatorio entre 0 y 100
print("El número aleatorio es: " + str(numero))
```


universidad
cenfotec_
La U de la informática