Prueba de Resiliencia

Programación Orientada a Objetos – 25 de mayo 2024

Nombre		**RUT**	
Paralelo	( ) APaolini ( ) MMoraga ( ) ERoss		

Antecedentes generales:

Puntaje total de la prueba Puntos para nota aprobatoria (4.0)	1 puntos 0.6 puntos	Puntaje Obtenido	
Duración de la prueba	3 horas	Nota final	
Resultados de Aprendizaje a evaluar	1. Aplicar técnicas de ingeniería de software en la creación de software legible, mantenible y testeable. 2. Aplicar técnicas de programación orientada al objeto en la resolución de problemas. 3. Crear tipos de datos abstractos con bajo acoplamiento entre la implementación y su comportamiento que permitan la resolución problemas. 4. Analizar las relaciones causa efecto de los procesos en estudio. 6. Seleccionar los procesos, técnicas y herramientas adecuados de acuerdo a los requerimientos.		
Fecha de entrega de resultados	8 junio 2024		

Instrucciones:

1. Esta evaluación tiene 6 páginas (incluyendo la portada). Compruebe que dispone de todas las páginas.
2. Lea la prueba completamente **DOS** veces antes de hacer cualquier pregunta
3. Durante la prueba no se puede utilizar: teléfono móvil, apuntes. Está prohibido intentar conectarse a internet de cualquier manera (excepto a Campus Virtual, y solo para subir su solución). Si es sorprendido obtendrá la calificación mínima. Tampoco puede utilizar dispositivos de almacenamiento externos o cualquier otro dispositivo relojes inteligentes, ábacos, etc.
4. Una prueba respondida correctamente en un 60%, de acuerdo con las ponderaciones asignadas, corresponde a una nota 4,0.
5. La prueba es individual, cualquier sospecha de copia será calificada con la nota mínima y el caso será remitido al comité de ética.
6. En su espacio personal no debe haber nada más que hojas de papel en blanco, lápiz, goma.
7. El resto de sus implementos debe guardarlos dentro de su mochila/bolso y ésta debe posicionarse al frente debajo de la pizarra. Si leyó hasta este punto, felicidades, para saber que lo hizo dibuje una esfera al final de esta página.
8. Los estudiantes quienes se les compruebe falta de honestidad académica o cualquier otro acto contrario a las normas de permanencia universitaria o al espíritu universitario, serán sancionados, según sea la gravedad de la falta, con medidas desde la amonestación verbal hasta la suspensión o pérdida de la condición de estudiante, los estudiantes expulsados no podrán volver a ingresar a ninguna carrera, programa o curso de la institución. El estudiante que incurriere en falta de honestidad, durante la realización de un proceso evaluativo, será calificado con la nota mínima 1,0.

Acepto las condiciones firmando: _____

Problema 1. Ruteo (0.3 puntos)

Rutee el siguiente código. Solo debe mostrar la salida por pantalla.

```java
public class App {

    public static void main(String[] args) {
        Completo c = new Completo("Completo", "Casera");
        LaBarra barra = new LaBarra();
        barra.pushCompleto(c);
        barra.venderCompleto(2);
        Completo c2 = new Completo("Completo", "Casera");
        Completo c3 = new Completo("Vegano", "Ajo");
        Completo c4 = new Completo("Italiano", "Casera");
        Completo c5 = new Completo("Italiano", "Ajo");
        barra.pushCompleto(c2);
        barra.pushCompleto(c4);
        barra.venderCompleto(0);
        barra.pushCompleto(c3);
        Completo c7 = new Completo("Italiano", "Casera");
        Completo c8 = new Completo("Vegano", "Ajo");
        barra.venderCompleto(2);
        Completo c9 = new Completo("Italiano", "Ajo");
        Completo c10 = new Completo("Completo", "Ajo");
        barra.pushCompleto(c7);
        barra.pushCompleto(c9);
        barra.venderCompleto(3);
        barra.pushCompleto(c10);
        barra.pushCompleto(c8);
        barra.venderCompleto(4);
        barra.venderCompleto(4);
        barra.mostrarVendidos();
        System.out.println(barra.getTotal());
    }
}
```

```java
public class Completo {
    private String tipo;
    private int precio;
    private String mayo;

    public Completo(String tipo, String mayo) {
        System.out.print("Pre");
        this.tipo = tipo;
        this.mayo = mayo;
        calcularPrecio();
    }

    private void calcularPrecio() {
        switch (tipo) {
        case "Italiano":
            System.out.println(" I");
            if (mayo.equals("Casera"))
                precio = 2500;
            else
                precio = 2300;
            break;
        case "Completo":
            System.out.println(" C");
            if (mayo.equals("Ajo"))
                precio = 2800;
            else
                precio = 2600;
            break;
        default:
            System.out.println(" O");
            precio = 2700;
            break;
        }
    }

    public int getPrecio() {
        return precio;
    }

    public String toString() {
        return tipo + " " + precio;
    }
}
```

```java
public class LaBarra {
    private Completo[] preparados = new Completo[5];
    private Completo[] vendidos = new Completo[8];
    private int total;

    public void pushCompleto(Completo completo) {
        if (preparados[0] == null) {
            preparados[0] = completo;
        } else {
            for (int i = preparados.length - 1; i >= 1; i--) {
                preparados[i] = preparados[i - 1];
            }
            preparados[0] = completo;
        }
        mostrarPreparados();
    }

    public void venderCompleto(int index) {
        if (preparados[index] != null) {
            for (int i = 0; i < vendidos.length; i++) {

                if (vendidos[i] == null) {
                    vendidos[i] = preparados[index];
                    preparados[index] = null;
                    total = getTotal() + vendidos[i].getPrecio();
                    break;
                }

            }
        } else
            System.out.println("No existe");
        mostrarPreparados();
    }

    private void mostrarPreparados() {
        System.out.print("[");
        for (int i = 0; i < preparados.length - 1; i++) {
            System.out.print(preparados[i] + ",");
        }
        System.out.println(preparados[preparados.length - 1] + "]");
    }

    public void mostrarVendidos() {
        System.out.print("[");
        for (int i = 0; i < vendidos.length - 1; i++) {
            System.out.print(vendidos[i] + ",");
        }
        System.out.println(vendidos[vendidos.length - 1] + "]");
    }

    public int getTotal() {
        return total;
    }
}
```

Problema 2. La Gran Guerra (0.7 puntos)

En la gran China, una guerra está afectando diversas regiones. Cuando estas regiones entran en conflicto, el gobierno comienza a confiscar los suministros y recursos de la tierra, dejando a la población en condiciones inhabitables. Cada terreno tiene una ubicación específica en el mapa del gobierno.

Cada terreno se dedica a la producción de un tipo específico de recurso (carne, maíz) y almacena estos recursos en una bodega. Los terrenos están habitados por familias que viven en casas, que pueden ser de normales o precarias.

Para poder rellenar el campo de terrenos se debe usar el archivo `Terrenos.txt`, en dónde la estructura es la siguiente:

Archivo

`Terreno 1,0,1,Carne,10000,4000,2`
`Casa 1,Precaria`
`Casa 2,Normal`
`Terreno 2,0,-1,Maíz,3000,12000,2`
`Casa 1,Normal`
`Casa 2,Precaria`

Explicación
La primera línea indica el **Nombre del terreno**, **su coordenada x**, **coordenada y**, **tipo de cultivo del terreno**, **Cantidad de Carne en bodega** [entero], **Cantidad de Maíz en la bodega** [entero], **Cantidad de casas en el terreno**[entero] Las siguientes líneas dependen de la cantidad de casas en el terreno y contienen **Nombre de la casa** [string], **tipo de casa** [string]

Las familias están compuestas por personas, cada una con su nombre, apellido y etnia. Cada persona tiene una cantidad específica de oro. Si vive en una casa normal, se les deja el 30% del oro para la persona y el ejército se quedará con el 70% en caso de que sea escasa se dejará el 50%. Además, se confiscarán los recursos de la bodega del terreno si hay una casa normal y más de tres miembros una de las casas. En tal caso, los recursos de la bodega se les va a quitar la mitad.

El programa debe permitir mostrar el mapa, que es una matriz de terrenos, y moverse desde el punto inicial (0,0) en las direcciones norte, sur, este, oeste. Al moverse, se evaluarán todas las condiciones para saquear los recursos según lo descrito. cada vez que se haga un movimiento se debe poder mostrar todos los recursos obtenidos, además de que cada movimiento reducirá en un 2% los recursos que van obteniendo por el motivo de que necesitan estos mismos recursos para poder sobrevivir.

Para poder ingresar las familias de cada casa en cada terreno se debe usar el siguiente txt, en dónde la estructura es la siguiente:

Archivo

`Terreno 1, Casa 1,Zhang,Wei,Han,1000`
`Terreno 1, Casa 1,Zhang,Li,Han,500`
`Terreno 1, Casa 1,Zhang,Fen,Han,750`
`Terreno 1, Casa 1,Li,Ping,Hui,600`
`Terreno 1, Casa 1,Li,Ming,Hui,800`
`Terreno 1, Casa 2,Wang,Hao,Zhuang,300`
`Terreno 1, Casa 2,Wang,Xia,Zhuang,400`
`Terreno 2, Casa 1,Chen,Long,Miao,450`
`Terreno 2, Casa 1,Chen,Hua,Miao,700`

Explicación
Cada línea indica el **Nombre del terreno**, **la casa asociada**, **el nombre de la persona, apellido, etnia, el oro de la persona**

Requisitos Funcionales

1. **Mostrar Mapa:** El programa debe mostrar el mapa de terrenos.
2. **Moverse:** Permitir al usuario moverse desde el punto inicial (0,0) en las direcciones norte, sur, este y oeste, al moverse en una dirección se va a mostrar los recursos obtenidos en cada movimiento (revisar las condiciones para saquear los recursos según lo descrito).
3. **Reducir Recursos por Movimiento:** Reducir los recursos obtenidos en un 2% después de cada movimiento, ya que el ejército necesita gastar recursos para movilizarse.
4. **Meta de Recursos:** Verificar si se cumple la meta de recursos y, en tal caso, imprimir un mensaje de que están listos para la guerra. Si cumplen la meta de recursos al saquear el terreno en el que se encuentran en conjunto con el acumulado, la cual es 1000 de oro, 1000 de carne y 1000 de maíz, imprimir un mensaje que ya están listos para la guerra.

Nota: Se puede realizar un movimiento para volver al terreno previo en el que estuvo, en ese caso los recursos cambiarían con respecto a la primera vez que se visitó.

Debe entregar:

- Diagrama del dominio (10%)
- Diagrama de clases (20%)
- Código Java (70%)

Consideraciones:

- Debe usar orientación al objeto.
- No se deben utilizar ciclos dentro de ciclos. Use funciones para hacerse la vida más fácil.
- El código fuente debe comprimirlo en un solo archivo .zip y subirlo a Campus Virtual.
- Archivos .txt de ejemplo se encuentran en CampusVirtual.
- Los diagramas de clases debe escribirlos en papel y entregarlos junto a la prueba.
- Hojas sin nombre no se revisarán.
- Debe usar referencias
- La matriz no va a tener más de 7 filas y 7 columnas