TALLER 2 – Lenguaje Java

Tema: Aplicación práctica del lenguaje Java en la resolución de un problema

Objetivo: Aplicar el lenguaje de programación Java mediante la búsqueda creativa grupal de una solución algorítmica al problema planteado, utilizando el concepto de abstracción y encapsulamiento de la POO y la modelación con diagramación UML de clases.

Fecha de envío del proyecto: 20 de noviembre de 2022.
Fecha de defensa del taller: 23 de noviembre de 2022.

Modalidad: El taller se llevará a cabo en modalidad GRUPAL durante 10 días, el trabajo realizado deberá ser enviado el día domingo 20 de noviembre y defendido el miércoles 23 de noviembre. Durante el período de desarrollo se organizarán encuentros destinados a la consulta y asistencia para completar la tarea.

Condiciones de presentación.

- Este taller debe realizarse en forma grupal.
- Los diagramas de flujo y UML que se requieran en los enunciados deben estar completamente desarrollados, incluyendo las descripciones necesarias para su mejor seguimiento (identificación, variables utilizadas y sus funciones, etc.).
- El código del proyecto solicitado debe estar correctamente rotulado para su identificación, incluyendo comentarios de seguimiento y deberá ser enviado mediante la plataforma CUV.FCEYT hasta el día indicado como límite de presentación.
- La resolución completa de este taller, incluyendo diagramas de flujo y UML, deberá pasar a integrar la carpeta de práctica y autoevaluación del alumno.

Criterios de evaluación y aprobación. Este taller recibirá una calificación de aprobado o desaprobado. Para aprobar la presentación debe cumplir como mínimo con los siguientes ítems:

- El taller debe estar desarrollado completamente.
- La codificación en lenguaje Java debe realizarse siguiendo las recomendaciones de la cátedra.
- El desarrollo de los diagramas de flujo y UML debe realizarse cumpliendo las indicaciones relativas a la diagramación estructurada y modular y sintaxis UML, respectivamente.
- La presentación del código de los enunciados solicitados deberá realizarse en tiempo y forma mediante la plataforma CUV.FCEYT: http://cuv.unse.edu.ar.

TALLER 2 – Lenguaje Java

ENUNCIADO

Un conductor maneja de un pueblo origen a un pueblo destino, pasando por varios pueblos.

Una vez que ha completado su recorrido y ha llegado al pueblo destino, el conductor debe regresar a casa por el mismo camino.

Desarrollar una aplicación que utilizando una estructura de datos pila estática realice las siguientes tareas:

Diario del viajante
1- Agregar lugar visitado
2- Mostrar recorrido realizado
3- Mostrar último lugar visitado
0-Salir
Ingrese su opción:

- Generar la estructura de datos pila que almacene: código de lugar, denominación del lugar, distancia (km), hora.
- Mostrar el camino recorrido tanto de ida como de vuelta.

Para llevar a cabo estas tareas se requiere desarrollar un programa con las siguientes opciones:

1. Opción Agregar lugar visitado:

Dar ingreso y almacenar la información de cada lugar indicando código, denominación, distancia desde el lugar de origen, horario de llegada.

2. Opción Mostrar recorrido realizado:

Se deberá mostrar los lugares visitados comenzando por el último y terminando por el lugar de origen indicando en forma encolumnada todos los datos. El almacenamiento deberá conservar su estado original.

Ej.:

Código	**Denominación**	**Distancia**	**Hora**
xxxx	xxxxxxxx	xxxxx	xxxxx
xxxx	xxxxxxxx	xxxxx	xxxxx
……			

3. Mostrar último lugar visitado:

Deberán mostrarse todos los datos del último lugar visitado. Posteriormente a realizada esta tarea el almacenamiento deberá conservar su estado original.

- Desarrolle la diagramación de flujo y la codificación en Lenguaje Java para cada una de las actividades.
- Los métodos correspondientes a cada opción pertenecerán a la clase principal, en la que se encuentra el método main.
- La resolución de todas las actividades deberá estar desarrollada en la carpeta de práctica de la asignatura y el programas en Java deberá ser enviado mediante la plataforma CUV hasta las fechas y horarios indicados.
- Para resolver el enunciado deberán utilizarse la clase Arreglo y la clase Pila, que implementa estáticamente la estructura de datos pila.
- Utilizar los códigos incluidos en el anexo.

ANEXO

```
public class Pila {
    private Arreglo a;
    private int tope;

    Pila(int dimension){
        a = new Arreglo(dimension);
        tope = -1;
    }

    public boolean apilar(int valor){
        if(!pilaLlena()){
            tope = tope + 1;
            a.setEle(tope, valor);
            return true;
            }
        else
            return false;
    }

    public int desapilar(){
        int valor = 0;
        if(!pilaVacia()){
            valor = a.getEle(tope);
            tope = tope - 1;
            }
        return valor;
    }

    public boolean pilaVacia(){
        return (tope == -1);
    }

    public boolean pilaLlena(){
        return (tope == (a.getDimension()-1));
    }

    public int espacio(){
        return (tope+1);
}
```

```
    //…continuación de la clase Pila

    //definicion del metodo imprimir()
    //muestra todos los elementos de la pila

    public void imprimir() {
        if (!pilaVacia()) {
            System.out.println("Listado de todos los
elementos de la pila.");
            System.out.println("Nota: la pila quedará
vacia!!!");
            while (!pilaVacia()) {
                System.out.print(desapilar() + "-");
            }
            System.out.print("\n");
        } else {
            System.out.println("ERROR: La pila esta
vacia");
        }
    }

} // fin de clase Pila
```

NOTA: en este ejemplo la clase Arreglo está definida en base al tipo int. Considere la re definición de la clase utilizando la clase Lugar.

```
public class Arreglo {
    private int[] a;

    Arreglo(){
        a = new int[5];
    }

    Arreglo(int n){
        a = new int[n];
    }

    public void cargar(){
        Scanner leer = new Scanner(System.in);
        for(int i = 0; i < a.length; i++){
            this.setEle(i, leer.nextInt());
        }
    }

    public void mostrar(){
        for(int i = 0; i < a.length; i++){
            System.out.println("Elemento ["+i+"] = " +
this.getEle(i));
        }
    }
    public void setEle(int pos, int valor){
        a[pos] = valor;
    }

    public int getEle(int pos){
        return a[pos];
    }

    public int getDimension(){
        return a.length;
    }
}
```