87 - Clases abstractas

Listado completo de tutoriales

El lenguaje Java permite definir clases que no se pueden instanciar o definir objetos de las mismas, este tipo de clases se las llama clases abstractas.

Una clase abstracta tiene por objetivo agrupar un conjunto de propiedades, métodos y firmas de métodos (métodos sin implementar, que deberán ser implementados por sus subclases)

Por si misma la clase abstracta no tiene sentido que se creen objetos, inclusive se genera un error sintáctico si se lo intenta.

Es común que se definan muchas subclases de una clase abstracta. Mediante un ejemplo veremos la sintaxis para crear una clase abstracta y como las subclases implementan los métodos abstractos de la misma.

Problema:

Confeccionar una clase abstracta que represente una Lista simplemente encadenada que defina dos métodos abstractos llamados: insertar y extraer un entero. Además definir un método que imprima la lista.

Luego plantear dos clases llamadas Pila y Cola que hereden de la clase abstracta Lista e implementen los dos métodos abstractos.

Clase: Lista

```
public abstract class Lista {
    public class Nodo {
        int info;
        Nodo sig;
    }

    protected Nodo raiz;

    public abstract void insertar(int x);

    public abstract int extraer(int x);

    public void imprimir() {
        Nodo reco = raiz;
        while (reco != null) {
            System.out.print(reco.info + " ");
            reco = reco.sig;
        }
        System.out.println();
```

```
    }
}
```

Una clase es abstracta si se le agrega la palabra clave 'abstract' previo a 'class', con esto ya no podemos crear objetos de la misma:

```
Lista lista1=new Lista(); // error de compilación
```

Una clase abstracta puede declarar clases internas:

```
    public class Nodo {
        int info;
        Nodo sig;
    }
```

Definir atributos:

```
    protected Nodo raiz;
```

Implementar métodos:

```
    public void imprimir() {
        Nodo reco = raiz;
        while (reco != null) {
            System.out.print(reco.info + " ");
            reco = reco.sig;
        }
        System.out.println();
    }
```

Podemos declarar métodos abstractos que obligan a las subclases a implementarlos:

```
    public abstract void insertar(int x);

    public abstract int extraer(int x);
```

Para que una clase abstracta tenga sentido deben declararse subclases de la misma, en nuestro ejemplo implementaremos las subclases 'Pila' y 'Cola'.

Clase: Pila

```
public class Pila extends Lista {

    public void insertar(int x) {
```

```
        Nodo nuevo = new Nodo();
        nuevo.info = x;
        nuevo.sig = raiz;
        raiz = nuevo;
    }

    public int extraer(int x) {
        if (raiz == null)
            return Integer.MAX_VALUE;
        else {
            int valor = raiz.info;
            raiz = raiz.sig;
            return valor;
        }
    }

    public static void main(String[] args) {
        Pila pila1 = new Pila();
        pila1.insertar(20);
        pila1.insertar(5);
        pila1.insertar(600);
        pila1.imprimir();
    }

}
```

Recordemos que mediante la palabra clave 'extends' indicamos que una clase hereda de otra:

```
public class Pila extends Lista {
```

Como heredamos de la clase Lista debemos obligatoriamente implementar los métodos insertar y extraer (en el caso que no lo hagamos se genera un error sintáctico):

```
    public void insertar(int x) {
        Nodo nuevo = new Nodo();
        nuevo.info = x;
        nuevo.sig = raiz;
        raiz = nuevo;
    }

    public int extraer(int x) {
        if (raiz == null)
            return Integer.MAX_VALUE;
        else {
            int valor = raiz.info;
            raiz = raiz.sig;
            return valor;
        }
    }
```

La clase Pila es una clase 'concreta' y no 'abstracta', luego podemos crear objetos de la misma:

```
    public static void main(String[] args) {
        Pila pila1 = new Pila();
        pila1.insertar(20);
        pila1.insertar(5);
        pila1.insertar(600);
        pila1.imprimir();
    }
```

De forma similar codificamos la clase 'Cola' e implementamos los algoritmos para insertar y extraer de una cola de enteros.

Clase: Cola

```
public class Cola extends Lista {

    public void insertar(int x) {
        Nodo nuevo = new Nodo();
        nuevo.info = x;
        if (raiz == null) {
            nuevo.sig = raiz;
            raiz = nuevo;
        } else {
            Nodo reco = raiz;
            while (reco.sig != null)
                reco = reco.sig;
            reco.sig = nuevo;

        }
    }

    public int extraer(int x) {
        if (raiz == null)
            return Integer.MAX_VALUE;
        else {
            int valor = raiz.info;
            raiz = raiz.sig;
            return valor;
        }
    }

    public static void main(String[] args) {
        Cola cola1 = new Cola();
```

Una clase abstracta permite que un conjunto de clases que heredan de la misma tengan una serie de métodos homogéneos para administrarlas. En nuestro ejemplo siempre que tenemos que agregar o eliminar elementos de una pila o cola debemos

llamar obligatoriamente a los métodos 'insertar' y 'extraer' que son las firmas definidas en la clase abstracta.

Problema propuesto

1. Plantear una clase abstracta llamada 'Operacion' que defina los atributos enteros:

```
valor1
valor2
resultado
```

Define el método abstracto:

```
operar
```

Defina el método concreto imprimir que muestre el atributo resultado:

```
imprimir
```

Declarar cuatro clases que hereden de la clase Operacion llamadas: Suma, Resta, Multiplicacion y Division.

Crear un objeto de cada clase que hereda de Operacion.

Solución

Retornar