Becerra Ramírez David Enrique

Operaciones básicas de las pilas

Una pila tiene las siguientes operaciones básicas:
Crear: se crea la pila vacía
Apilar: (push), que coloca un elemento en la pila
Retirar (o desapilar, pop), es la operación inversa que retira el último elemento apilado
Cima: devuelve el elemento que esta en la cima de la pila (top o peek).
Vacía: devuelve true si la pila está vacía o falso en caso contrario.

Especificación

TAD Pila [T]
{ invariante: TRUE }
Constructoras:
crearPila()
Modificadoras:
apilar()
desapilar()
Analizadoras:
cima()
esVacia()
Destructora:
destruirPila()

---

crearPila(void)
/* Crea una pila vacía */
{ post: crearPila = }

---

void apilar(T elem)
/* Coloca sobre el tope de la pila el elemento elem */
{ post: pil = e1, e2, .. elem}

---

void desapilar()
/* Elimina el elemento que se encuentra en el tope de la pila */
{ pre: pil =e1, e2, ..en, n > 0 }
{ post: pil =e1, e2, .., en-1 }

T cima()
/* Retorna el elemento que se encuentra en el tope de la pila */
{ pre: n > 0 }
{ post: cima = en }

---

boolean esVacia()
/* Informa si la pila está vacía */
{ post: esVacia = (pil =) }

---

void destruirPila()

Becerra Ramírez David Enrique

/* Destruye la pila retornando toda la memoria ocupada */
{post: pil ha sido destruida }

Implementación en JAVA

```
package capitulo2.pilas;
public class Nodo<T> {
        private T valor;
        private Nodo<T> siguiente;

        public Nodo() {
                valor = null;
                siguiente = null;

        }

        public T getValor() {
                return valor;
        }

        public void setValor(T valor) {
                this.valor = valor;
        }

        public Nodo<T> getSiguiente() {
                return siguiente;
        }

        public void setSiguiente(Nodo<T> siguiente) {
                this.siguiente = siguiente;
        }
}
```

```
package capitulo2.pilas;
public class Pila<T> {
        private Nodo<T> cabeza;
        private int tamanio;

        public Pila() {
                cabeza = null;
                tamanio = 0;
        }

        public int getTamanio() {
                return tamanio;
        }

        public boolean esVacia() {
                return (cabeza == null);
        }

        public void apilar(T valor) {
                Nodo<T> nuevo = new Nodo<T>();
                nuevo.setValor(valor);
                if (esVacia()) {
                        cabeza = nuevo;
                } else {
                        nuevo.setSiguiente(cabeza);
                        cabeza = nuevo;
```

Becerra Ramírez David Enrique

```
        }
        tamanio++;
}

public void retirar() {
        if (!esVacia()) {
                cabeza = cabeza.getSiguiente();
                tamanio--;
        }
}

public T cima() {
        if (!esVacia())
                return cabeza.getValor();
        else
                return null;

        }
}

package capitulo2.pilas;
public class ClienteMain {
        public static void main(String[] args) {
                Pila<Integer> pila2 = new Pila<Integer>();
                pila2.apilar(2);
                pila2.apilar(5);
                pila2.apilar(7);
                System.out.println(pila2.cima());
                pila2.retirar();
                System.out.println(pila2.cima());
                pila2.retirar();
                System.out.println(pila2.cima());
                pila2.retirar();
                System.out.println(pila2.cima());
                //Probar con otra pila, donde se almacenen objetos tipo Persona o Contacto o Libro, etc.
//Algo así;
                //Pila <Contacto>pila2 = new Pila<Contacto>();
                //pila2.apilar(new Contacto(2,"Juan Perez", "31245434","juanito@hotmail.com");
                //pila2.desapilar();
                //...
        }
}
```

Bibliografía

José Fager, W. Libardo Pantoja Yépez (2014), Estrcuturas de Datos, LATIn, Mexico. Pags 141-143