TP N°7: Tipos Genéricos

La siguiente guía cubre los contenidos vistos en la clase teórica **6**

Ejercicio 1

Implementar la clase `Pair<A,B>` que almacena dos objetos: uno de tipo `A` y otro de tipo `B`. El siguiente programa de prueba:

```
package ar.edu.itba.poo.tp7.pair;

public class PairTester {

   public static void main(String[] args) {
       Pair<String, Double> stringDoublePair = new Pair<>("hola", 0.1);
       System.out.println(stringDoublePair);
       Pair<Integer, Integer> integerIntegerPair = new Pair<>(1, 2);
       System.out.println(integerIntegerPair);
       System.out.println(stringDoublePair.equals(new Pair<>("hola", 0.1)));
   }

}
```

produce la siguiente salida:

```
[hola, 0.1]
[1, 2]
true
```

¿El siguiente fragmento de código funciona? ¿Cuál es el tipo genérico inferido?

```
Pair otherPair = new Pair(1f, "mundo");
System.out.println(otherPair);
```

Ejercicio 2

Se desea agregar a la implementación anterior la posibilidad de setear el contenido de las variables de instancia. Aprovechando la sobrecarga de métodos se propone la siguiente implementación:

```
package ar.edu.itba.poo.tp7.pair;

public class Pair<A, B> {

    private A a;
    private B b;
```

```
...
```

```
   public void set(A a) {
```

```
        this.a = a;
    }

    public void set(B b) {
        this.b = b;
    }

}
```

Se obtiene el error que el metodo set colisiona con el otro set. lo unico que hay que hacer es agregarle signature

¿Qué error se obtiene en compilación? ¿Cómo se puede solucionar?

~~*Ejercicio 3*~~

Modificar la implementación del ***Ejercicio 9*** del ***TP Nº6*** para que tanto la implementación como la interfaz `LinearList` sea genérica. Realizar el diagrama de clases correspondiente.

¿Qué ventaja tiene esta alternativa frente a la implementación anterior en donde se utiliza `Object`, si en definitiva en ambos casos se tiene una implementación genérica que permite almacenar cualquier tipo de elementos?

~~*Ejercicio 4*~~

Se cuenta con la clase `PrependLinearListImpl<T>` que, a diferencia de la implementación anterior, agrega los elementos al principio de la lista.

¿Por qué la siguiente implementación falla en compilación? Corregirlo.

```
package ar.edu.itba.poo.tp7.list;

public class PrependLinearListImpl<T> implements LinearList<T> {

    private Node<T> first;

    @Override
    public void add(T obj) {
        first = new Node<>(obj);
    }
```

```
...
```

```
    private class Node<T> {

        private T head;
        private Node<T> tail;

        Node(T head) {
            this.head = head;
            this.tail = first;
            first = this;
        }

    }

}
```

Ejercicio 5

Implementar en Java el `Stack` del ***Ejercicio 2*** del ***TP Nº2***. Diseñar una interfaz `Stack<E>` y una implementación con arrays `ArrayStack<E>`. Realizar el diagrama de clases correspondiente.

¿Por qué falla en compilación el siguiente fragmento de código? Corregirlo.

```
package ar.edu.itba.poo.tp7.stack;

public class ArrayStack<E> implements Stack<E> {

   private E[] elements;
   private static final int INITIAL_DIM = 10;

   ...

   public ArrayStack() {
       elements = new E[INITIAL_DIM];
   }

   ...

}
```

El siguiente programa de prueba:

```
package ar.edu.itba.poo.tp7.stack;

public class StackTester {

   public static void main(String[] args) {
       Stack<Integer> stack = new ArrayStack<>();
       stack.push(2);
       stack.push(3);
       System.out.println(stack);
       System.out.println(stack.peek());
       System.out.println(stack.pop());
       System.out.println(stack.isEmpty());
       System.out.println(stack.pop());
       System.out.println(stack.isEmpty());
       System.out.println(stack.pop());
   }

}
```

imprime la siguiente salida:

```
[3,2]
3
3
false
2
true
Exception in thread "main" java.util.EmptyStackException
      at ar.edu.itba.poo.tp7.stack.ArrayStack.pop(ArrayStack.java:34)
      at ar.edu.itba.poo.tp7.stack.StackTester.main(StackTester.java:15)
```

Ejercicio 6

Implementar en Java el `AccessStack` del ***Ejercicio 3*** del ***TP Nº2***. Además de la nueva implementación `AccessStack<E>` ¿corresponde crear una nueva interfaz? Realizar el diagrama de clases correspondiente.

Ejercicio 7

Consultar la documentación del método de clase `sort` de la clase `Arrays`. ¿Cuál es la salida del siguiente programa de prueba? Justificar.

```
package ar.edu.itba.poo.tp7.ej7;

import java.util.Arrays;

public class PairTester {

   public static void main(String[] args) {
       Integer intArray[] = new Integer[]{7, 3, 1, 5, 9};
       Arrays.sort(intArray);
       arrayPrinter(intArray);
   }

   public static <T> void arrayPrinter(T[] array) {
       StringBuilder stringBuilder = new StringBuilder("[");
       for(int i = 0; i < array.length; i++) {
           stringBuilder.append(array[i]).append(",");
       }
       stringBuilder.setCharAt(stringBuilder.length() - 1,  ']');
       System.out.println(stringBuilder.toString());
   }

}
```

[1,3,5,7,9]
con el metodo sort los ordena en orden ascendente

Ejercicio 8

Indicar la salida del siguiente fragmento de código adicionado al programa de prueba del ejercicio anterior.

```
Pair<String, String> stringPair1 = new Pair<>("hola", "mundo");
Pair<String, String> stringPair2 = new Pair<>("hola", "adiós");
Pair<String, String> stringPair3 = new Pair<>("buen", "día");
Pair pairArray[] = new Pair[]{stringPair1, stringPair2, stringPair3};
Arrays.sort(pairArray);
arrayPrinter(pairArray);
```

Implementar todo lo necesario para que las instancias de la clase `Pair` puedan ordenarse en un array. A modo de ejemplo, para el fragmento anterior, se espera obtener la siguiente salida:

```
[[buen, día],[hola, adiós],[hola, mundo]]
```

donde el orden natural de los pares es ascendente por el primer campo y luego por el segundo.

Ejercicio 9

Indicar la salida del siguiente fragmento de código:

```
Integer intArray[] = new Integer[]{7, 3, 1, 5, 9};
Arrays.sort(intArray, new Comparator<Integer>() {
    @Override
    public int compare(Integer o1, Integer o2) {
        return o2.compareTo(o1);
    }
});
arrayPrinter(intArray);
```

orden descendente

con una expresion lambda

o con
Comparator.reverseOrder()
Arrays.toString(intArray)

¿Se puede simplificar el código del programa de prueba?

Ejercicio 10

Modificar el fragmento de código del ***Ejercicio 8*** para que el arreglo de instancias de `Pair` tengan el orden inverso al orden natural. A modo de ejemplo, con el nuevo orden, se espera obtener la siguiente salida:

```
[[hola, mundo],[hola, adiós],[buen, día]]
```