Práctica 3

Ejercicio 1.

Complete el código de la clase Stack en el paquete practica3, de manera que implemente una pila de String:

```
public class Stack {
   private java.util.ArrayList items;
   public Stack() { . .  }
   public void push(Object item) { . . }
   public Object pop() { . . }
   public boolean isEmpty() { . . }

}
```

a) Implemente un método main() para probar la pila. Agregue Strings a la pila y recórrala para imprimir sus valores. ¿Cuántas veces puede recorrerla?

   Se puede recorrer una sola vez ya que al conseguir los valores estamos eliminándolos en el método pop().

b) Agregue una clase anidada llamada StackIterator que provea un objeto de tipo Iterator para recorrer la pila.

c) Agregue en la clase Stack un método para que retorne una instancia de StackIterator. ¿Cuántas veces puede recorrer la pila ahora?

   Se puede recorrer la cantidad de veces que se desee, ya que ahora el Iterator no elimina los elementos.

d) ¿Es posible crear objetos StackIterator desde una clase diferente a la clase Stack con el operador new?, ¿cómo lo hace?

Si es posible, se debe hacer la clase anidada publica y se crea como *clase.new claseAnidada*().

e) ¿Cómo haría para evitar crear instancias de una clase anidada desde una clase que no sea la que la definió?

Haciendo a la clase privada anidada.

Ejercicio 2.

Analice el código que figura debajo.

```
class InnerStatic {
    static double PI = 3.1416;
    static class Circulo {
        static double radio = 2;
        static double getArea() {
            double a= PI* Math.pow(radio,2);
            System.out.println("El area es: "+a);
            return a;
        }
        static double getLongitudCircunsferencia(){
            double l= 2*PI*radio;
            System.out.println("La longitud es: "+l);
            return l;
        }
    }
    . . .
}
```

a) Modifique el código de la clase interna estática para que el valor inicial del radio sea ingresado por el usuario en el momento de la ejecución.

b) Defina una clase llamada InnerTest en el paquete practica3 con un método main() que imprime en la pantalla el área y la longitud de la circunferencia. Ejecútela varias veces ingresando distintos radios.

Introduce el valor del radio: 2
El área es: 12.5664

La longitud es: 12.5664

Introduce el valor del radio: 2

El área es: 12.5664

La longitud es: 12.5664

Introduce el valor del radio: 4

El área es: 50.2656

La longitud es: 25.1328

Introduce el valor del radio: 4

El área es: 50.2656

La longitud es: 25.1328

Introduce el valor del radio: 5

El área es: 78.53999999999999

La longitud es: 31.416

c) Remplazar PI* Math.pow(radio,2) por PI* pow(radio,2), siendo pow() el método de la clase java.lang.Math.

import static java.lang.Math.pow;

Introduce el valor del radio: 2

El área es: 12.5664

La longitud es: 12.5664

Introduce el valor del radio: 2

El área es: 12.5664

La longitud es: 12.5664

Introduce el valor del radio: 4

El área es: 50.2656

La longitud es: 25.1328

Introduce el valor del radio: 4

El área es: 50.2656

La longitud es: 25.1328

Introduce el valor del radio: 5

El área es: 78.53999999999999

La longitud es: 31.416

Ejercicio 3.

Implemente una clase llamada StringConverterSet como subclase de AbstractSet, la cual permita realizar todas las operaciones contempladas para los Set, con la salvedad que el método iterator() retorne un Iterator que al recorrelo devuelva cada uno de los elementos como Strings.

Para su solución, defina un Adapter llamado IteratorStringAdapter como una clase anidada de StringConverterSet para cumplir lo solicitado.

Ejercicio 4.

Indicar si son verdaderas o falsas las siguientes afirmaciones sobre las clases anónimas y en cada caso justifique su respuesta:

- Se pueden instanciar más allá del punto en donde fueron declaradas.

  Falso. Solo pueden instanciarse en el lugar donde fueron declaradas, son simultáneamente declaradas e instanciadas en el punto en que se van a usar.

- Unos de los usos más comunes de este tipo de clases es la creación de objetos función y procesos on the fly.

  Verdadero. Estas clases se crean para poder crear instancias rápidas de clases como funciones callback o manejadores de eventos, cuando no es necesario definir una clase completa.

- Se puede utilizar el instanceof siempre y cuando la interfaz de la que deriva la clase anónima sea de tipo marker.

  Falso. Se puede usar independientemente de que la interfaz sea de tipo marker o no. La referencia que devuelve el método que retorna esta clase anónima es automáticamente upcasteada a una referencia de la clase padre, por lo que es

posible determinar mediante instanceof si esa referencia es una instancia de dicha clase padre.

- No se puede implementar múltiples interfaces o extender clases e implementar interfaces al mismo tiempo.

  Verdadero. Como se crean de la siguiente manera *return new Clase(){...}* solamente pueden implementar o extender a una sola clase (*Clase* en este caso). Es por ello que las clases anónimas son limitadas dado que sólo pueden extender una clase o implementar una interface, no pueden hacer ambas cosas a la vez ni tampoco pueden implementar más de una interface.

Ejercicio 5.

Modifique el código de la clase Stack, para que ahora la clase anidada StackIterator, se convierta en una clase anónima.

a) ¿En qué situación es conveniente definir a una clase cómo anónima?

   Conviene definir una clase anónima en los casos donde no es conveniente crear toda una clase completa ya que su uso esta estrechamente relacionado con una tarea especifica que se quiere realizar y no se necesita de la clase en ninguna otra parte del programa, por lo que se realiza una implementación rápida, sencilla y compacta.

b) Si tendría que inicializar valores de la clase anónima (cuando se crea una instancia de la misma), ¿cómo lo haría?

   Si se tendría que inicializar valores de clase anónima cuando se instancia la misma lo haría con un bloque de inicialización.

Ejercicio 6.

Defina una clase llamada Estudiante que contenga las siguientes variables de instancia: apellido, nombre, edad, legajo y materiasAprobadas. Se necesita poder ordenar un arreglo con estos objetos por los siguientes criterios:

- Por cantidad de materias aprobadas en forma ascendente.
- Por edad en forma descendente.
- Por legajo en forma ascendente.
- Por nombre y apellido en forma descendente.

Implemente un método main() que imprima los resultado de las distintas ordenaciones utilizando clases anónimas y el método Arrays.sort().

Ejercicio 7.

Declaración e implementación de Tipos Enumerativos

a) Implemente un tipo enumerativo llamado Notas que define los valores de las notas musicales y con su correspondiente cifrado americano (almacenado en un String).

b) Implemente un tipo enumerativo llamado FrecuenciasDeLA que represente las siguientes frecuencias estándares de afinación:
    - 440 Hz: Organización Internacional de Estandarización ISO 16.
    - 444 Hz: Afinación de cámara.
    - 446 Hz: Renacimiento.
    - 480 Hz: Órganos alemanes que tocaba Bach.

c) Sobrecargue los métodos hacerSonar() y afinar() de la interface InstrumentoMusical del ejercicio 1b) de la práctica 2 de manera que el nuevo hacerSonar(Notas n, int duracion) reciba como parámetro una nota musical y una duración, y el nuevo método afinar(FrecuenciaDeLA f) reciba como parámetro una frecuencia de LA.

d) Defina una clase llamada Piano que implemente la interface InstrumentoMusical y una clase TestPiano que permita probar los métodos implementados.

e) Implemente el patrón de diseño Singleton mediante un tipo Enumerativo el cual represente a Fito Páez. Fito cuenta con un instrumento musical (piano) y en algún momento se le puede pedir que toque una canción (especificando un arreglo de notas musicales con sus tiempos).