TAREA 7

Nombre: Emmanuel Peto Gutiérrez

1.- Complete las siguientes oraciones con las palabras que se encuentran en el recuadro:

Concretas, abstract, Polimorfismo, Abstracta

a) Si una clase contiene al menos un método abstracto, es una clase abstracta.
b) Las clases a partir de las cuales pueden instanciarse objetos se llaman clases concretas.
c) El polimorfismo implica el uso de una variable de referencia del tipo de la superclase para invocar métodos en objetos de superclase y subclase, lo cual nos permite “Programar en general”.
d) Los métodos que no son métodos de interfaz y que no proporcionan implementaciones deben declararse utilizando la palabra clave abstract.

2.- Conteste verdadero o Falso a cada una de las siguientes proposiciones; en caso de ser falso, explique por qué.

a) Todos los métodos en una clase abstract deben declararse como métodos abstract.
Falso. Solo los métodos que no tienen cuerpo deben tener el *abstract*.

b) No está permitido invocar a un método que solo pertenece a una subclase, a través de una variable de referencia de la subclase.
Verdadero

c) Si una superclase declara a un método como abstract, toda subclase debe implementar a ese método.
Falso. Si la subclase es abstracta no es necesario que lo implemente.

3.- Dadas las siguientes declaraciones:

```
class A {}
abstract class B {}
class C extends B {}
class D extends A {}
class E extends C {}
class F extends E {}
```

1. A no es una clase abstracta.
2. C hereda de la clase A.
3. No se pueden crear instancias de la clase B.
4. F hereda de la clase B.
5. F hereda de la clase A.
6. D hereda de B.

¿Cuáles afirmaciones son ciertas?

a) 1, 3 y 4
b) 1,2,3, y 5
c) 1 y 5
d) Todas

4.- ¿Identifica el error en el siguiente código?

```
abstract class AbstractClass
{
    private abstract int abstractMethod();
}
```

El error está en que un método abstracto no puede ser privado (en este caso, el método abstractMethod). Esto es porque, si fuera privado, solamente la clase AbstractClass lo podría ver y los hijos de AbstractClass no podrían implementar este método (porque no lo podrían ver).

5.- ¿Qué clase es instanciable? ¿Clase A o Clase B?

```
abstract class A
{

}

class B extends A
{

}
```

Solo la clase B. La clase A no es instanciable porque es abstracta.

6.- ¿El fragmento de código siguiente muestra un error de compilación? ¿Puede sugerir las correcciones?

```
abstract class A
{
    abstract int add(int a, int b);
}

class B extends A
{

}
```

El error de compilación surge porque la clase B debe implementar el método abstracto add(int, int), ya que está declarado en la clase abstracta A, B hereda de A y B es una clase concreta que actualmente tiene un cuerpo vacío.

Una posible solución sería implementar el método add en la clase B. El siguiente código se pondría en la clase B:

```
@Override
int add(int a, int b){
    return a+b;
}
```

7.- ¿Es correcto declarar métodos protegidos en una interfaz?
No es correcto.

8.- ¿Podemos escribir constructores explícitos en una clase abstracta?
Sí se puede.

9.- ¿Cuál será la salida del siguiente programa?

```
abstract class A
{
    abstract void firstMethod();

    void secondMethod()
    {
        System.out.println("SECOND");

        firstMethod();
    }
}

abstract class B extends A
{
    @Override
    void firstMethod()
    {
        System.out.println("FIRST");

        thirdMethod();
    }

    abstract void thirdMethod();
}
```

```
class C extends B
{
    @Override
    void thirdMethod()
    {
        System.out.println("THIRD");
    }
}

public class MainClass
{
    public static void main(String[] args)
    {
        C c = new C();

        c.firstMethod();

        c.secondMethod();

        c.thirdMethod();
    }
}
```

La salida es la siguiente:

FIRST
THIRD
SECOND
FIRST
THIRD
THIRD