**Desarrollo Avanzado de Aplicaciones I**

**Evaluación Continua 3**

Desarrollar una aplicación que simule una feria inmobiliaria. Considerar las siguientes instrucciones:

1. Definir una clase que tenga como nombre el apellido paterno del alumno. La clase debe ser pública y debe tener su correspondiente archivo “apellidopaterno.java”. La clase debe tener su método “main”.
2. Dentro del mismo archivo del numeral 1. Se debe crear el código que implemente la siguiente imagen:

alquilar()

vender()

ALQUILER = true

VENDER = true

<<Interface>>

Habitable

extends

extends

ofrecerAlquiler()

ofrecerVender()

toString()

int pisos

boolean piscina

Casa

ofrecerAlquiler()

ofrecerVender()

toString()

int piso

boolean ascensor

Departamento

alquilar()

vender()

ofrecerAlquiler()

ofrecerVender()

int numCuartos

int metros2

boolean mascotas

String estado

Vivienda

implements

1. Definir una interface “Habitable” con las siguientes consideraciones:
   1. Debe tener 2 atributos constantes de tipo boolean que indiquen si la propiedad se puede alquilar y/o vender.
   2. Debe tener 2 métodos abstractos: alquiler() y vender(), como se muestra en la figura.
2. Definir una clase “Vivienda” con las siguientes consideraciones:
3. Debe ser una clase abstracta “abstract” y debe implementar la interface “Habitable”. También debe ser una clase sellada “sealed” y permitir las subclases “Departamento” y “Casa”.
4. Debe tener los atributos indicados en la figura encapsulados.
5. Debe tener un método constructor con 3 parámetros donde se reciban los valores de los atributos “numCuartos”, “metros2” y “mascotas”. Al atributo “estado” se le asignará por defecto el valor “Disponible”.
6. Los métodos que implemente de la interface deben ser finales “final” y deben cambiar el atributo “estado” a “Alquilado” o “Vendido” según corresponda.
7. Debe tener los métodos getters y setters necesarios. El atributo “estado” debe ser de solo lectura.
8. Debe tener dos métodos abstractos: ofrecerAlquiler() y ofrecerVender() que devuelven un valor boolean.
9. Definir una clase “Departamento” con las siguientes consideraciones:
10. Debe ser una subclase de la clase “Vivienda” y debe definirse como “final”.
11. Debe tener los atributos indicados en la figura encapsulados.
12. Debe tener un método constructor con 5 parámetros donde se reciban los valores de todos los atributos excepto “estado”. Con los parámetros de la clase padre se debe llamar al constructor de la clase padre. A los atributos propios se le asignará su valor directamente con los parámetros recibidos.
13. Debe sobrescribir el método ofrecerAlquiler() y devolver como valor la constante “ALQUILER” heredada de la interface.
14. Debe sobrescribir el método ofrecerVender() y devolver como valor la negación de la constante “VENDER” heredada de la interface.
15. Debe sobrescribir el método toString() y devolver una cadena de caracteres de la siguiente forma:

[Dpto. de 90m2, Cuartos: 3, Mascotas: No, Piso: 7, Ascensor: Sí, Alquiler: Sí, Vender: No, Estado: Disponible]

\* Los valores mostrados son solo ejemplos.

Se debe utilizar un arreglo dinámico de tipo String almacenando en cada elemento una cadena de caracteres con el correspondiente atributo convertido en String, con el formato: “nombreAtributo: valorAtributo”. Para los valores booleanos convertir en “Sí” para verdadero y en “No” para falso. Por último, devolver el arreglo, utilizando de forma explicita su método “toString”.

1. Definir una clase “Casa” con las siguientes consideraciones:
2. Debe ser una subclase de la clase “Vivienda” y debe definirse como no sellada “non-sealed”.
3. Debe tener los atributos indicados en la figura encapsulados.
4. Debe tener un método constructor con 5 parámetros donde se reciban los valores de todos los atributos excepto “estado”. Con los parámetros de la clase padre se debe llamar al constructor de la clase padre. A los atributos propios se le asignará su valor directamente con los parámetros recibidos.
5. Debe sobrescribir el método ofrecerAlquiler() y devolver como valor la negación de la constante “ALQUILER” heredada de la interface.
6. Debe sobrescribir el método ofrecerVender() y devolver como valor la constante “VENDER” heredada de la interface.
7. Debe sobrescribir el método toString() y devolver una cadena de caracteres de la siguiente forma:

[Casa de 150m2, Cuartos: 3, Mascotas: Sí, Pisos: 2, Piscina: No, Alquiler: No, Vender: Sí, Estado: Vendido]

\* Los valores mostrados son solo ejemplos.

Se debe utilizar un arreglo dinámico de tipo String almacenando en cada elemento una cadena de caracteres con el correspondiente atributo convertido en String, con el formato: “nombreAtributo: valorAtributo”. Para los valores booleanos convertir en “Sí” para verdadero y en “No” para falso. Por último, devolver el arreglo, utilizando de forma explicita su método “toString”.

1. El método “main” debe tener las siguientes consideraciones:
   1. Debe crear un arreglo estático del tipo “Vivienda” con 6 elementos, 3 instancias de la clase “Departamento” y 3 instancias de la clase “Casa”.
   2. Se debe modificar algunos de los elementos del arreglo para alquilar o vender según corresponda, es decir cambiar su atributo “estado”, utilizando los métodos alquilar() para los departamentos y vender() para las casas.
   3. Se debe recorrer el arreglo para mostrar todos sus elementos.
   4. Todo el código debe estar protegido de suceder una excepción. De suceder una excepción debe mostrar el mensaje de error y el programa debe terminar correctamente.
   5. El resultado debe ser como en la siguiente imagen:













