Solucion Taller 4
Cristhian David Amaya 25481092

1. Responda las siguientes preguntas de Colecciones:
a. ¿Cuál es la principal diferencia entre Set y Map?

las diferencias entres set y Map radican en que Set no permite entradas duplicadas, el almacenamiento es desordenado, esto quiere decir que no se tiene total certeza de que posición hace referencia al elemento aunque su implementación es más eficiente, mientras que Map requiere de una clave para acceder a un elemento, es decir que para guardar un elemento en Map se requieren dos parámetros (Key,E) en donde Key es el identificador del elemento guardado y por obvias razones tiene que ser única, igualmente es desorganizada esto quiere decir que el orden de los elementos no se mantiene en orden de inserción pero tampoco garantiza que la posición de un elemento se mantenga constante en el tiempo.

b. ¿Qué pasa cuando se agrega un valor de tipo primitivo a una colección?

Una condición fundamental para agregar un elemento a una colección es que este elemento sea un objeto, es decir que no se pueden agregar tipos primitivos, pero existe la posibilidad de usar una clase envolvente, una operación que se conoce como Box, aunque directamente en el código se pueden agregar tipos primitivos, el compilador realiza la operación de Box automáticamente, Autoboxing, convirtiendo el tipo primitivo a un objeto para agregarlo a la colección e invirtiendo el proceso al intentar setear una variable primitiva con el elemento de la colección.

c. En clase se estudió, ArrayList, HashMap, TreeMap; busque otra colección y realice una tabla comparativa entre las 4 colecciones, en donde se muestre las características especiales de cada una, cuando usar una con respecto a la otra, etc.

ArrayList	**HashMap**	**TreeMap**	**ArrayDeque**
Permite elementos repetidos	Las claves de cada elemento deben ser únicas	Solo los valores pueden ser nulos	No se permiten elementos iguales
Implementación de list	implementación de map	implementación de VavigableMap, SortedMap y map	Collection, Deque, Queue
Mantiene el orden de inserción	No mantiene ningún orden	Se ordena de acuerdo al orden natural de pedidos	Orden LIFO o FIFO
se obtiene un elemento por la posición	se obtiene un elemento por la clave	se obtiene un elemento por la clave	Se obtiene el último elemento de la lista o el primero

2. Responda las siguientes preguntas (Tema de parcial)

a. ¿Por qué se puede decir que un método abstracto es un contrato?

Al usar un método abstracto en una una clase, para poder instanciar la misma se requiere la implementación de todos y cada uno de sus métodos, por lo que la definición de método abstracto implica que ese método debe ser implementado con posterioridad.

b. ¿Cuál es el propósito principal de la clase Abstracta?

Una clase abstracta tiene como propósito proporcionar un esquema de herencia mediante una superclase que permite evitar la repetición de código unificando datos, es decir que si por ejemplo se tiene varias derivaciones da la misma naturaleza estas se pueden referenciar mediante la misma clase evitando duplicar el código para cada una de estas siendo accesibles mediante una variable de tipo ClaseAbstacta, un ejemplo sencillo e ilustrativo sería una clase abstracta llamada Perro de la cual se derivan todas las razas, entonces se evitaría duplicar código al implementar por ejemplo un método llamado caminar o comer ya que para todas las razas es igual pero se diferencian en algunos de sus atributos o métodos como por ejemplo en el color, altura, la conducta .. etc que se definirian en las clases correspondientes para cada raza.

c. ¿Cómo se puede tener ventaja del polimorfismo teniendo un método abstracto?

De hecho es, considero yo, el ejemplo más sencillo de polimorfismo ya que la clase hija definirá el método según le sea más adecuado, por lo tanto, siguiendo con el hilo anterior, teniendo como ejemplo el método conducta, para algunas razas esta es diferente y por lo tanto haciendo un llamado al mismo método el resultado puede ser diferente dependiendo del tipo de raza que se esté considerando y de su llamada peligrosidad.

d. Falso o verdadero, Si es falso Justifique.

i. Todos los métodos en una clase abstracta deben ser declarados como métodos abstractos.

FALSO: una clase se considera abstracta cuando mínimo uno de sus métodos es abstracto, aunque si se define la clase mediante la palabra reservada abstract todos sus métodos deben ser abstractos.

ii. Si una superclase declara un método abstracto, una subclase debe implementar ese método.

FALSO: la subclase o clase hija tiene la opción de definir ese método como abstracto convirtiéndose a su vez en una clase abstracta.

iii. Un objeto de una clase que implementa una interfaz puede ser pensado como un objeto de ese tipo de interfaz.

VERDADERO

e. ¿Cuándo se usa una clase abstracta y cuando una interfaz?

Se podría decir que el uso de una interfaz es de tipo más global al contexto de la aplicación como por ejemplo haciendo alusión a la interfaz Serializable que permite a un objeto que instancia a la clase declarada con esta interfaz guardarse en un archivo o una cadena binaria, a diferencia de las clases abstractas cuyo uso generalmente se enmarca en el contexto de la aplicación para hacer uso de la herencia y evitar la duplicación de código.

3. Características Estáticas

a. (Parcial) Existen clases que nos proporcionan formas convenientes de realizar conductas de uso frecuente sin tener que crear una instancia de un objeto, para realizar algún tipo de comportamiento (Ej, Math, Calendar, Integer), ¿Cuáles son los requisitos para que una clase sea una “Utility class” en términos de métodos o atributos?

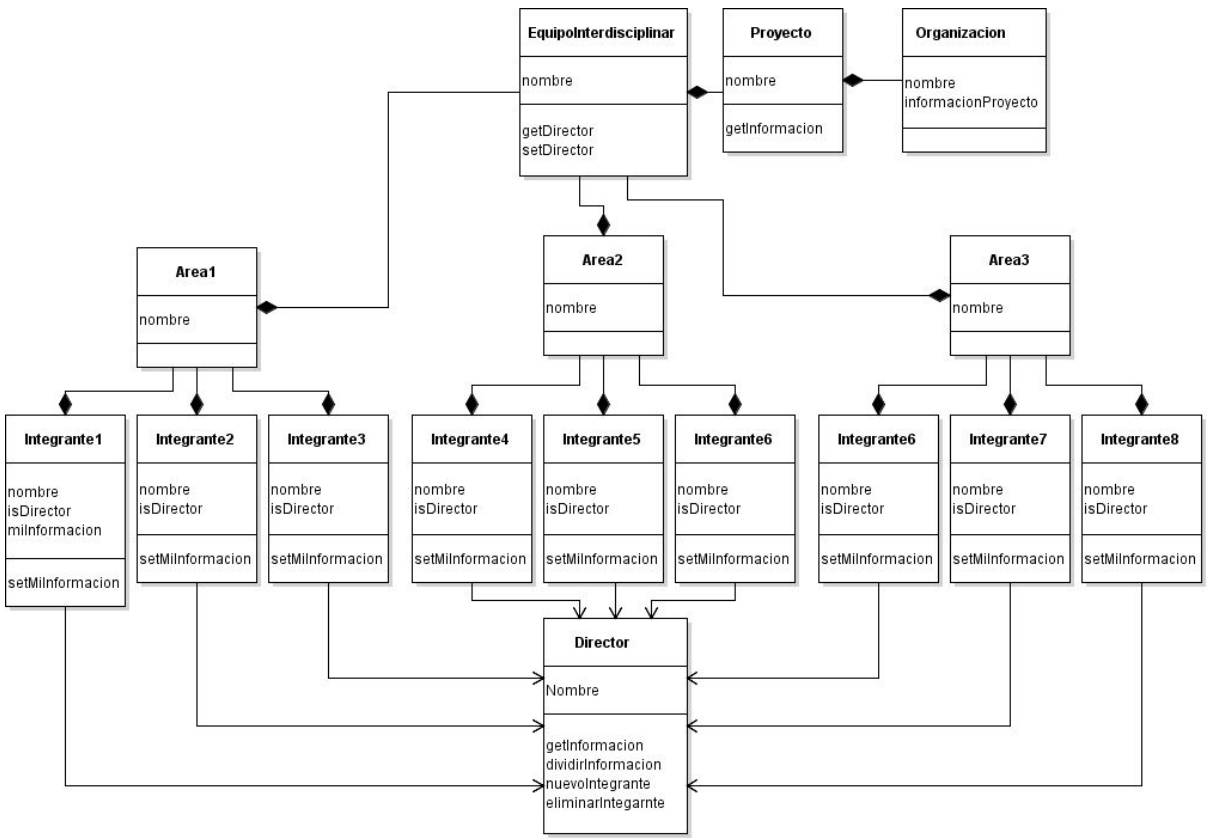
Una Utility class no requiere ser instanciada para poder usar algunos de sus métodos, por lo tanto se requiere que una clase posea métodos estáticos que de ser necesario hagan referencia a sus atributos estáticos que deben ser previamente inicializados, garantizando así que estos atributos tengan un valor inicial, y que los métodos declarados como estáticos estén disponibles mediante la llamada a la misma clase y mediante el operador punto poder usar estos métodos estáticos, de esta forma no se requiere instanciar la clase ni usar un objeto para realizar las diferentes acciones.

4. Polimorfismo y Clases Abstractas

c. ¿De qué tipo debe ser arrayList? ¿Por qué?

El ArrayList deberá ser inicializado usando el tipo Figuras, clase de la cual heredad diferentes atributos y métodos las clases que se instancian, pudiendose considerar del tipo Figuras.

6. Diseño UML



Los Puntos faltantes son prácticos y se encuentran adjuntos.