TRABAJO PRACTICO N° 2

INTRODUCCION AL DISEÑO ORIENTADO A OBJETOS

Nota: para cada clase hacer un programa que pruebe cada uno de los métodos que provee.

Clases y Objetos

1. Crear la clase ***Complejo*** para realizar aritmética de números complejos.
a) Utilizar variables de punto flotante para representar los datos privados de la clase.
b) Proporcionar un constructor que permita inicializar un objeto de esta clase cuando se declara.
c) Incluir métodos públicos para cada una de las siguientes operaciones con complejos: suma, resta, producto, cociente, división escalar.
d) Implementar el método ***toString*** para imprimir un número complejo.

2. Crear la clase ***Racional*** para realizar aritmética de números racionales.
a) Utilizar variables enteras para representar los datos privados de la clase.
b) Proporcionar un método constructor que permita inicializar un objeto de esta clase cuando se declara.
c) Incluir métodos públicos para cada una de las siguientes operaciones con racionales: suma, resta, producto, división y potencia.
d) Implementar el método ***toString*** para imprimir un número racional.
e) Agregar métodos para simplificar un racional.
f) Modificar los métodos de c) para que den racionales simplificados.
g) Incluir el atributo *static* ***cuenta*** que se incrementa cada vez que se crea un objeto de esta clase.

3. Crear la clase ***Tiempo*** para trabajar con tiempos. La clase debe registrar horas, minutos y segundos. Verificar la consistencia de los datos. Agregar los siguientes métodos:
a) ***incrementarSegundo*** - incrementa el tiempo en un segundo.
b) ***incrementarMinuto*** - incrementa el tiempo en un minuto.
c) ***incrementarHora*** - incrementa el tiempo en una hora.

4. Crear la clase ***Fecha*** para trabajar con fechas. La clase debe registrar año, mes y día. Verificar la consistencia de los datos. Agregar los siguientes métodos:
a) ***diaSiguiente*** - incrementa la fecha en un día.
b) ***diaAnterior*** - decrementa la fecha en un día.

5. Agregar métodos a la clase ***Fecha*** para proporcionar las siguientes operaciones: verificar si una fecha es mayor, menor o igual a otra fecha dada, calcular la cantidad de días entre dos fechas, calcular una nueva fecha a partir

TRABAJO PRACTICO N° 2

INTRODUCCION AL DISEÑO ORIENTADO A OBJETOS

de una fecha y una cantidad de días que se suman (o restan según su signo) a la misma, devolver el día de la semana de una fecha dada.

6. Crear una clase llamada ***ConjuntoEnteros***. Cada objeto ***ConjuntoEntero*** puede almacenar enteros en el rango de 0 a 100. El conjunto se representa mediante un arreglo de valores *boolean.* El elemento del arreglo *a[j]* es *false* si el entero *j* no se encuentra dentro del conjunto. El constructor sin argumentos inicializa el arreglo con el "conjunto vacío" (el arreglo con todos sus valores en *false*)

Proporcionar los siguientes métodos:
a) ***insertarElemento*** - inserta un nuevo entero *k* en el conjunto (*a[k]* en *true*)
b) ***eliminarElemento*** - elimina el entero *k* del conjunto (*a[k]* en *false*)
c) ***aStringConjunto*** - devuelve una cadena que muestra los elementos que están en el conjunto separados por espacios.
d) ***esIgualA*** - determina si dos conjuntos son iguales
e) ***union*** - crea un tercer conjunto con la unión de dos conjuntos dados
f) ***interseccion*** - crea un tercer conjunto con la intersección de dos conjuntos dados.
g) ***diferencia*** - crea un tercer conjunto con todos los elementos que están en el primero conjunto y no están en el segundo.

Herencia

7. Crear la clase ***Empleado*** que tiene como atributos nombre y número de documento, además crear las subclases: ***EmpleadoAsalariado*** que tiene el valor del salario mensual, ***EmpleadoPorHora*** el valor de la hora y el número de horas trabajadas, ***EmpleadoPorComision*** el porcentaje de comisiones y las ventas brutas, ***EmpleadoBaseMasComision*** (subclase de la anterior) que contiene el salario base. Cada clase debe contener los constructores y los métodos *gets* y *sets* apropiados. Escribir un programa que cree instancias de cada clase y muestre toda la información asociada para cada objeto incluso la información heredada.

8. Crear la clase ***CuentaBancaria*** la misma debe registrar el número de cuenta, titular y saldo. Crear la clase ***CajaAhorro*** que extiende de ***CuentaBancaria*** en la misma se puede hacer depósitos y extracciones siempre que el saldo sea mayor o igual a cero. Crear la clase ***CuentaCorriente*** que también extiende de ***CuentaBancaria***, aquí también los clientes puede hacer depósitos y extracciones, en este caso puede tener un saldo negativo hasta determinado monto fijado para cada cliente. De cada cliente se registra su nombre, cuit, dirección y email. Escribir un programa que cree instancias de objetos para cada clase y realizar depósitos y extracciones.

TRABAJO PRACTICO N° 2

INTRODUCCION AL DISEÑO ORIENTADO A OBJETOS

9. Crear la clase ***Figura*** y las subclases ***FiguraBidimensional***, ***FiguraTridimensional***, ***Circulo***, ***Rectangulo***, ***Esfera*** y ***PrimaRectangular*** en la jerarquía que corresponda. Utilizar objetos de la clase ***Punto*** para indicar sus coordenadas. Para todas las figuras agregar métodos para calcular su perímetro, área y volumen. Escribir un programa que cree instancias de objetos para cada clase y llame a los métodos implementados.

Polimorfismo

10. Cargar una lista de empleados utilizando las clases creadas en el ejercicio 7 y emitir un listado de todos los empleados y sus salarios. Adicionar un 10% de incremento en el listado para los ***EmpleadoBaseMasComision.***

11. Crear la clase ***Factura*** que tiene el nombre del proveedor, número de factura, fecha de compra y una lista de ítems que tiene la descripción, el precio unitario y la cantidad comprada. Crear la interface ***PorPagar*** que tiene el método ***double obtenerPago().*** Implementar la interface en la clase ***Factura*** y en la clase ***Empleado*** del ejercicio anterior. Cargar una lista de empleados y facturas y calcular los importes a pagar utilizando el método de la interface implementada.

12. Modificar las clases que correspondan del ejercicio 8 para que cada cliente contenga referencias a las cuentas bancarias que posee. Agregar un método en la clase ***Cliente*** que retorne el saldo total de todas sus cuentas.

13. Crear una clase ***Banco*** que contenga una lista de clientes y una lista de cuentas bancarias. Crear métodos dentro de la clase ***Banco*** para calcular la suma total de los saldos de cada cliente y la suma total de los saldos negativos de los clientes que tienen cuenta corriente.