TRABAJO PRÁCTICO N° 2

Forma de entrega: Comprima cada carpeta de proyecto con el nombre "TP2_EjercicioN°" y luego comprima todos en "Nombre_Apellido–TP2.rar" subalo a su carpeta en la correspondiente carpeta de la materia en GooleDrive.
Ayuda: En caso de no comprender alguna consigna o tener dudas, puede solicitarse asistencia enviando un email a consultas@profmatiasgarcia.com.ar con el asunto "Programación 3 TM Nombre y Apellido TP2 CONSULTA".
Modalidad: Individual

Ejercicio 1

Cree una clase llamada Empleado, que incluya tres variables de instancia: el nombre (String), un apellido (String) y un salario mensual (double). Su clase debe tener un constructor que inicialice las tres variables de instancia. Proporcione los métodos get y set para cada variable de instancia. Si el salario mensual no es positivo, no establezca su valor. Escriba una aplicación de prueba llamada PruebaEmpleado, que demuestre las capacidades de la clase Empleado. Cree dos objetos Empleado y muestre el salario anual de cada objeto. Después, proporcione a cada Empleado un aumento del 10% y muestre el salario anual de cada Empleado otra vez.

Ejercicio 2

Realizar una aplicación sencilla para simular una cuenta bancaria (caja de ahorro). Una cuenta bancaria vista como un objeto tiene, por una parte, atributos que definen su estado, como Tipo de interés y Saldo, y por otra, operaciones que definen su comportamiento, como Establecer tipo de interés, Ingresar dinero, Retirar dinero, Saldo actual o Abonar intereses. Asegúrese que el monto a retirar no exceda el saldo de Cuenta. Si lo hace, el saldo debe permanecer sin cambio y el método debe imprimir un mensaje que indique " El monto a retirar excede el saldo de la cuenta." Realice la clase PruebaCuenta para probar los diferentes métodos.

Ejercicio 3

Cree una clase llamada Factura, que una ferretería podría utilizar para representar una factura para un artículo vendido en la tienda. Una Factura debe incluir un número de factura, un número de articulo, la descripción del articulo, la cantidad de artículos de ese tipo que se van a comprar y el precio por artículo. Su clase debe tener un constructor que inicialice las variables de instancia. Proporcione los métodos get y set para cada variable de instancia. Además, proporcione un método llamado obtenerMontoFactura, que calcule el monto de la factura (es decir, que multiplique la cantidad por el precio por artículo) y después lo devuelva como un valor double. Si la cantidad no es positiva, debe establecerse en 0. Si el precio por artículo no es positivo, debe establecerse en 0.0. Escriba una aplicación de prueba llamada PruebaFactura, que demuestre las capacidades de la clase Factura.

Ejercicio 4

Cree una clase llamada Rectángulo con los atributos longitud y anchura, cada uno con un valor predeterminado de 1. Debe tener métodos para calcular el perímetro y el área del rectángulo. Debe tener métodos get y set para longitud y anchura. Los métodos set deben verificar que longitud y anchura sean números de punto flotante mayores de 0.0, y menores de 20.0. Escriba un programa para probar la clase Rectángulo y cada uno de sus métodos.

Ejercicio 5

Cree una clase llamada Fecha, que incluya tres variables de instancia: un mes (int), un día (int) y un año (int). Su clase debe tener un constructor que inicialice las tres variables de instancia, y debe verificar que los valores que se proporcionan son correctos. Proporcione los métodos get y set para cada variable de instancia. Proporcione un método mostrarFecha , para Imprimir la fecha en varios formatos, como

05/21/2010

Junio 14, 2012

20 Agosto 11

Usar constructores sobrecargados para crear objetos Fecha inicializados con fechas de los formatos solicitados para imprimir. En el primer caso, el constructor debe recibir tres valores enteros. En el segundo, debe recibir un objeto String y dos valores enteros. En el tercero debe recibir un valor entero, string y otro entero. Escriba una aplicación de prueba llamada PruebaFecha, que demuestre las capacidades de la clase Fecha.

Ejercicio 6

El sector de ventas online de notebooks Lenovo paga a sus vendedores mediante comisiones. Los vendedores reciben $2000 por semana, más el 6% de sus ventas brutas durante esa semana. Por ejemplo, un vendedor que vende $50000 de mercancía en una semana, recibe $2000 más el 6% de $50000, o un total de $5000. Usted acaba de recibir una lista de los artículos vendidos por cada vendedor. Los valores de estos artículos son los siguientes:

Articulo	Valor
1	$ 7239.99
2	$ 9129.75
3	$ 6899.95
4	$ 13150.89

Desarrolle una aplicación en JAVA que reciba como entrada los artículos vendidos por un vendedor durante cada día de la última semana, y que calcule y muestre los ingresos de ese vendedor. No hay límite en cuanto al número de artículos que un representante puede vender.

Ejercicio 7

Una compañía tiene cuatro vendedores que venden cinco productos distintos. Al finalizar la jornada laboral, cada vendedor pasa una nota por cada tipo de producto vendido. Cada nota contiene lo siguiente:

a) El número del vendedor

b) El número del producto

c) El valor total de ese producto vendido en ese día

Así, cada vendedor pasa entre 0 y 5 notas de venta por día. Suponga que está disponible la información sobre todas las notas del mes pasado. Escriba una aplicación que lea toda esta información para las ventas del último mes y que resuma las ventas totales por vendedor, y por producto. Todos los totales deben guardarse en el vector bidimensional ventas.

Después de procesar toda la información del mes pasado, muestre los resultados en formato tabular, en donde cada columna represente a un vendedor específico y cada fila simboliza un producto. Saque el total de cada fila para obtener las ventas totales de cada producto durante el último mes. Calcule el total de cada columna para sacar las ventas totales de cada vendedor durante el último mes. Su impresión tabular debe incluir estos totales cruzados a la derecha de las filas totalizadas, y en la parte inferior de las columnas totalizadas.

Ejercicio 8

Cree una clase llamada CuentaDeAhorros. Use una variable static llamada tasalnteresAnual para almacenar la tasa de interés anual para todos los clientes. Cada objeto de la clase debe contener una variable de instancia llamada saldoAhorros, que indique la cantidad que el ahorrador tiene actualmente en depósito. Proporcione el método calcularlnteresMensual para calcular el interés mensual, multiplicando el saldoAhorros por la tasalnteresAnual dividida entre 12; este interés debe sumarse al saldoAhorros. Proporcione un método static llamado modificarTasalnteres para establecer la tasalnteresAnual en un nuevo valor. Escriba un programa para probar la clase CuentaDeAhorros. Cree dos instancias de objetos CuentaDeAhorros, ahorradorl y ahorrador2, con saldos de $20000.00 y $30000.00. Establezca la tasalnteresAnual en 4%, después calcule el interés mensual para cada uno de los 12 meses e imprima los nuevos saldos para ambos ahorradores. Luego establezca la tasalnteresAnual en 5%, calcule el interés del siguiente mes e imprima los nuevos saldos para ambos ahorradores.

Ejercicio 9

Cree una clase llamada Complejo para realizar operaciones aritméticas con números complejos. Estos números tienen la forma

parte Real + parte imaginaria * i

Escriba un programa para probar su clase. Use variables de punto flotante para representar los datos private de la clase.

Proporcione un constructor que permita que un objeto de esta clase se inicialice al declararse. Proporcione un constructor sin argumentos con valores predeterminados, en caso de que no se proporcionen inicializadores. Ofrezca métodos public que realicen las siguientes operaciones:

a) Sumar dos números Complejos: las partes reales se suman entre sí y las partes imaginarias también.

b) Restar dos números Complejos: la parte real del operando derecho se resta de la parte real del operando izquierdo, y la parte imaginaria del operando derecho se resta de la parte imaginaria del operando izquierdo.

c) Imprimir números Complejos en la forma (parte Real, parte imaginaria) .

Ejercicio 10 (Opcional)

Cree una clase llamada Racional para realizar operaciones aritméticas con fracciones.

Escriba un programa para probar su clase. Use variables enteras para representar las variables de instancia de la clase: el numerador y el denominador. Proporcione un constructor que permita inicializarse a un objeto de esta clase al ser declarado. El constructor debe almacenar la fracción en forma reducida. La fracción 2/4 es equivalente a 1/2 y debe guardarse en el objeto como 1 en el numerador y 2 en el denominador. Proporcione un constructor sin argumentos con valores predeterminados, en caso de que no se proporcionen inicializadores. Proporcione métodos public que

realicen cada una de las siguientes operaciones:

a) Sumar dos números Racional: el resultado de la suma debe almacenarse en forma reducida. Implemente esto como un método static.

b) Restar dos números Racional: el resultado de la resta debe almacenarse en forma reducida. Implemente esto como un método static.

c) Multiplicar dos números Racional: el resultado de la multiplicación debe almacenarse en forma reducida. Implemente esto como un método static .

d) Dividir dos números Racional: el resultado de la división debe almacenarse en forma reducida. Implemente esto como un método static.

e) Devolver una representación String de un número Racional en la forma a/b, en donde a es el numerador y b es el denominador.

f) Devolver una representación String de un número Racional en formato de punto flotante. (Considere proporcionar capacidades de formato, que permitan al usuario de la clase especificar el número de dígitos de precisión a la derecha del punto decimal).

Ejercicio 11 (Opcional)

Un juego de azar popular es el juego de dados conocido como “Craps”, el cual se juega en casinos y callejones por todo el mundo. Las reglas del juego son simples:

Un jugador tira dos dados. Cada uno tiene seis caras, las cuales contienen uno, dos, tres cuatro, cinco y seis puntos negros, respectivamente. Una vez que los dados dejan de moverse, se calcula la suma de los puntos negros en las dos caras superiores. Si la suma es 7 u 11 en el primer tiro, el jugador gana. Si la suma es 2, 3 o 12 en el primer tiro (llamado “Craps”), el jugador pierde (es decir, la “casa” gana). Si la suma es 4, 5, 6, 8, 9 o 10 en el primer tiro, esta suma se convierte en el “punto” del jugador. Para ganar, el jugador debe seguir tirando los dados hasta que salga otra vez “su punto” (es decir, que tire ese mismo valor de punto). El jugador pierde si tira un 7 antes de llegar a su punto.

(Modificación del juego de Craps) Modifique el programa Craps para permitir apuestas. Iniciar la variable saldoBanco con $1,000. Pida al jugador que introduzca una apuesta. Compruebe que esa apuesta sea menor o igual que saldoBanco y, si no lo es, haga que el usuario vuelva a introducir la apuesta hasta que se ingrese un valor válido.

Después de esto, comience un juego de Craps. Si el jugador gana, agregue la apuesta al saldoBanco e imprima el nuevo saldoBanco. Si pierde, reste la apuesta al saldoBanco, imprima el nuevo saldoBanco, compruebe si saldoBanco se ha vuelto cero y, de ser así, imprima el mensaje "Lo siento. Se quedo sin fondos! ” A medida que el juego progrese, imprima varios mensajes para crear algo de “charla”, como "Oh, se esta yendo a la quiebra, verdad?”,o “Oh, vamos, arriesguese!”, o “La hizo en grande. Ahora es tiempo de cambiar sus fichas por efectivo!”. Implemente la “charla” como un método separado que seleccione en forma aleatoria la cadena a mostrar.