Universidad Tecnológica Nacional.
Facultad Regional Haedo.
Tecnicatura Superior en Programación.

Alumno/a: Programación III - 1º Parcial

Un cliente, requiere un sistema modelado e implementado observando los principios de la POO para registrar y administrar las ventas de autopartes de un comercio de venta de repuestos del automotor. Los datos pertinentes a dicho objeto *Autoparte* son:

- *GANANCIA*: Número real (punto flotante de precisión simple) – Valor único compartido para todas las instancias del objeto. Representa el porcentaje de incremento sobre el costo que se desea establecer como margen de ganancia.Ejemplo:*25,75%.*
- *Código de Pieza*: Número entero del intervalo [0 a 999.999.999]. Ejemplo:*733.615.993*.
- *Descripción:* Cadena de caracteres descriptiva de la Autoparte. Ejemplo:*"Correa de Distribución".*
- *Costo*: Número real (punto flotante de precisión simple) que representa el valor de reposición del artículo. Ejemplo:*$189,50*.

Se pide:

A) Código fuente del archivo de clase `CAutoparte.cs` conteniendo:

1) Adecuada declaración de las variables miembro.
2) Un método set*Descripción (detalle*), que recibiendo por valor como argumento, un valor *detalle,* cadena de caracteres, establezca adecuadamente valor para la variable miembro *Descripción.*
3) Un método *setGANANCIA(porcentaje),* que recibiendo por valor como argumento, un valor *porcentaje,* número real (punto flotante de precisión simple), establezca adecuadamente valor para la variable miembro *GANANCIA.*
4) Un método constructor *CAutoparte (código)*, que recibiendo por valor como argumento, un dato *código,* número entero del intervalo [0 a 999.999.999], establezca valor inicial para la variable miembro *CodPieza.*
5) Un método *setCosto(monto),* que recibiendo por valor como argumento, un valor *monto,* número real (punto flotante de precisión simple), establezca valor para la variable miembro *Costo.*
6) Un método *darPrecio(),* que sin argumento alguno, devuelva como número real (punto flotante de precisión simple), el *Precio* a abonar por la pieza, producto de aplicarle el porcentaje de incremento de *GANANCIA* al valor del *Costo* de la autoparte(Si *GANANCIA* es 100%, se abonará un monto igual al doble del *Costo*).
7) Un método *darDatos(),*que sin argumento alguno, devuelva una cadena de caracteres que concatene los valores presentes en las variables miembro *CodPieza, Descripción, Costo* más el valor calculado de *Precio* a abonar por la Autoparte(emplear método anterior).
8) Un método de instancia *costoMayorQue(otraAutoparte),* que recibiendo por valor como argumento, una referencia a una instancia de *CAutoparte,* devuelva como dato lógico, el valor *True*, cuando la instancia invocante presente un *Costo* mayor que la recibida como argumento y, el valor *False*, en cualquier otro caso.
9) Un método sobrecargado *darPrecio(cuotas)*, que recibiendo por valor como argumento la cantidad de cuotas de pago en que se abonará (número entero corto positivo), devuelva como número real (punto flotante de precisión simple), el valor total a abonar, conforme adicionar una penalización del 10% por cada cuota, aplicada sobre el monto original (interés simple – Si *cuotas* es igual a1, el recargo es 0%).

B) Código fuente del archivo de clase `CEjecutora.cs` conteniendo el método *Main( )*de una aplicación de consola*,* que permita:

1) Solicitar al usuario y establecer el valor de *GANANCIA* único y compartido para todas las Autopartes.
2) Solicitar iterativamente y registrar, los datos de un conjunto de extensión desconocida de Autopartes; finalizando dicho proceso de carga con el ingreso de un número de *Código de Pieza* de valor igual a 0 (cero), para cuyo caso no deberán solicitarse ni registrarse los datos restantes *Descripción* y *Costo.*
3) Finalizado el proceso de carga, informar *Código de Pieza–Descripción – Costo – Precio* de la autoparte más cara (primero de las ingresadas en caso de existir más de una), y el monto a abonar en cada uno de los pagos en un plan de 12 (doce) cuotas.
4) En caso de no haberse ingresado Autoparte válida alguna, en lugar de lo indicado en el ítem anterior, informar la leyenda *"No se ingresaron Autopartes.".*

Criterios:

Se evaluará sólo lo presentado por escrito, en tinta y sin tachaduras ni correcciones de ningún tipo.

- Adicionalmente, se calificará:
  - La eficacia en la funcionalidad de la aplicación.
  - Las técnicas, metodologías y estrategias de programación aplicadas.
  - La prolijidad general y la claridad conceptual en el desarrollo.
  - La eficiencia en el empleo y la gestión de los recursos de memoria.
  - La eficiencia en el empleo de estructuras de control y gestión del flujo de ejecución.
  - La correcta promoción de tipos, que debe ser estricta y explícita.
  - La observancia de los principios de la P.O.O. estudiados.
  - El cumplimiento de los tiempos y pautas de examinación.
  - Importante: No se requiere validación de los datos de entrada.