RETO 1 – PROGRAMACIÓN BÁSICA

VARIANTE 2

Una nueva empresa de telefonía móvil busca hacerse campo entre las ya existentes en Colombia; busca ofrecer nuevos planes de telefonía e internet con precios más asequibles a los estratos económicos más vulnerables sin perder el factor de la calidad del servicio, además de ofrecer un servicio al cliente donde el consumidor sienta que verdaderamente le importa a la compañía.

Para la configuración de la SIMCard, se buscará hacer un prototipo inicial, que buscará simular su funcionamiento en un entorno de consola, para ello se ha decidido realizar la implementación en Java.

Usted ha sido contratado como Java Expert Developer, porque ha logrado demostrar habilidades de desarrollo en este lenguaje de programación y se le ha concedido implementar una clase llamada SIMCard.

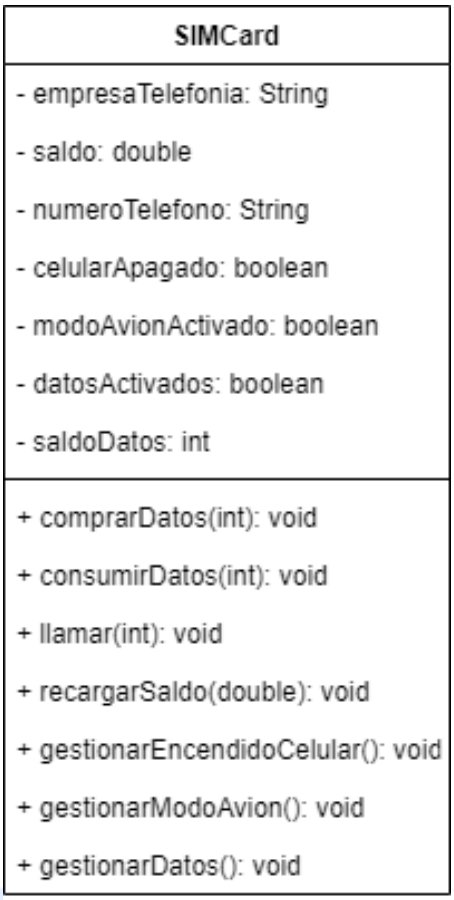
La empresa hace las siguientes observaciones sobre el funcionamiento de este prototipo:

1. Si el celular está apagado o en modo avión, no se pueden realizar llamadas, ni consumir datos.
2. Si los datos están apagados no se puede consumir datos.
3. Si el celular se pone en modo avión y los datos están encendidos, se apagan, y si se intentan activar teniendo el modo avión activado, NO se activarán.
4. Si el celular está apagado, los datos y el modo avión estarán desactivados.
5. El cobro de los datos se hará en la unidad de medida Gigabytes (GB), y los precios son los siguientes:
    a. Si el cliente compra 10 GB o menos, se cobran a 3000 pesos cada GB.
    b. Si el cliente compra entre 10 GB a 30 GB, las primeras 10GB se cobran a 3000 pesos cada GB, y las demás a 2500 pesos cada una.
    c. Si el cliente compra más de 30 GB, las primeras 20GB se cobran a 3000 pesos cada GB, y las demás a 1500 pesos cada una.

6. El cobro de los datos se hará en la unidad de medida Segundos, y los precios son los siguientes:
    a. Si la llamada dura 60 segundos o menos, se cobra cada segundo a 1 peso.
    b. Si la llamada dura más de 60 segundos, se cobran los primeros 60 segundos a 1 peso, y los demás a medio peso.

Para facilitar la implementación de la clase SIMCard, el equipo de Ingeniería de software le hace entrega del diagrama de clases del prototipo que se desea implementar, recuerde que los métodos relacionados a los getters y setters son obviados en el diagrama de clases, pero deberán ser incluidos en el código (Estos métodos deberán ser creados con el estándar camel case: Por ejemplo, si el atributo se llama `numeroTelefono`, sus métodos correspondientes a get y set serían `getNumeroTelefono` y `setNumeroTelefono`).

Los getters de las variables booleanas son especiales, por ejemplo, si el atributo que contiene un dato de tipo booleano se llama `celularApagado`, su "getter" correspondiente es `isCelularApagado()`



Además del diagrama, el equipo de Ingeniería entrega esta documentación para comprender mejor los elementos del diagrama:

Clase SIMCard

Atributos

NOMBRE	TIPO DATO	CONCEPTO	INICIALIZACIÓN
`empresaTelefonia`	`String`	Nombre de la empresa de telefonia	Debe estar inicializado en `"HI"`
`saldo`	`double`	Cantidad de dinero a gastar en datos o llamadas	Debe estar inicializado en 0
`numeroTelefono`	`String`	Contiene el número de celular con el que puede llamar y recibir llamadas	En el método constructor
`celularApagado`	`boolean`	`true` si el celular se encuentra apagado y `false` en caso de que esté encendido	Debe estar inicializado en `true`
`modoAvionActivado`	`boolean`	`true` si el celular está en modo avión y `false` en caso de que no lo esté.	Debe estar inicializado en `false`
`datosActivados`	`boolean`	`true` si el celular tiene los datos activados y `false` en caso de que no lo estén.	Debe estar inicializado en `false`
`saldoDatos`	`int`	Contiene la cantidad de GB compradas	Debe estar inicializado en 0

Métodos

NOMBRE	TIPO RETORNO	PARÁMETROS	CONCEPTO
`comprarDatos`	`void`	`int c`: Cantidad de GB a comprar	Resta el costo de las `c` GB a `saldo`, y suma `c` a `saldoDatos`. (Si el costo supera el saldo, NO modificar ningún atributo.
`consumirDatos`	`void`	`int c`: Cantidad de GB a consumir	Resta `c` a `saldoDatos` *Leer la nota aclaratoria*
`llamar`	`void`	`int s`: Cantidad de segundos gastados en la llamada.	Resta el costo de los `s` segundos a `saldo`. *Leer la nota aclaratoria*
`recargarSaldo`	`void`	`double s`: Cantidad de saldo a recargar para ser usado en llamadas o compra de datos.	Suma `s` a `saldo`
`gestionarEncendidoCelular`	`void`	No recibe parámetros	Enciende el celular si está apagado al invocar este método y viceversa. Si se apaga el celular, los datos móviles y el modo avión deberán apagarse. Si se enciende el celular, los datos móviles y el modo avión permanecerán apagados, hasta que explícitamente se enciendan.

`gestionarModoAvion`	`void`	No recibe parámetros	Enciende el modo avión si está apagado al invocar este método y viceversa. Si se apaga el modo avión, los datos móviles permanecerán apagados. Si se enciende el modo avión, los datos se apagarán. *Leer la nota aclaratoria*
`gestionarDatos`	`void`	No recibe parámetros	Enciende los datos móviles si está apagado al invocar este método y viceversa. *Leer la nota aclaratoria*

Notas aclaratorias:

- Los datos móviles no se pueden encender si el celular se encuentra apagado o con el modo avión activado.
- El modo avión no se puede encender si el celular se encuentra apagado.
- No se puede llamar ni consumir datos si el celular está apagado o si el modo avión está encendido (Esta incapacidad de llamar, o consumir datos se ve reflejada en NO modificando los atributos, dejándolos intactos, no deberá hacer ningún retorno sobre si pudo llamar o no, si pudo gastar datos o no).
- Tampoco se puede comprar datos si el saldo es menor al costo de los datos a comprar.
- Suponga que el cliente **NO** consumirá más datos o más segundos de llamada de lo que tiene en el saldo.

PRECISIONES

1. No hay métodos estáticos.
2. El método constructor debe asignar **SOLAMENTE** el atributo `numeroTelefono`.

3. Los atributos `saldo` y `saldoDatos` deben ser inicializados en `0`, el atributo `empresaTelefonia` debe ser inicializado con `"HI"`, y los atributos `modoAvionActivado`, `datosActivados` deben ser inicializados en `false` y el atributo `celularApagado` debe ser inicializado en true.
4. Deben existir `getters` y `setters` de todos los atributos de cada clase, estos deben ser escritos en la forma estándar camel case.

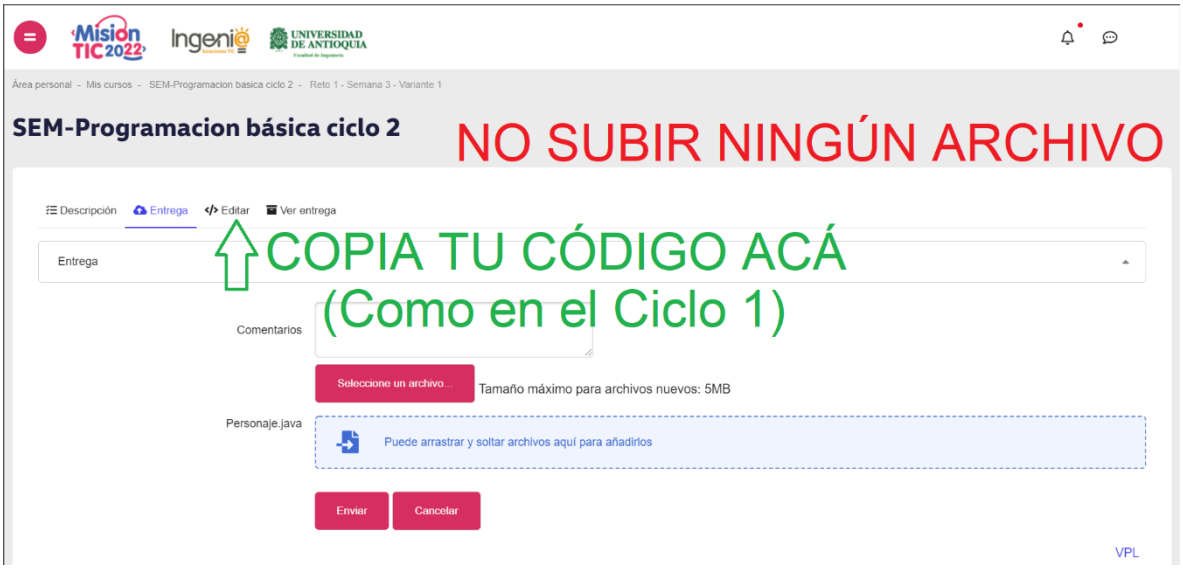
TAREAS

- En el archivo preconstruido en la plataforma Moodle, implementar la clase especificada en el diagrama de clases, teniendo en cuenta las precisiones dadas por el equipo de Ingeniería de software.
- Los nombres de los métodos y atributos **DEBEN** ser nombrados tal y como aparecen en el diagrama de clases.
- Usted **NO** debe solicitar datos por teclado, ni programar un método `main`, tampoco use `Java Source Package`, usted está solamente encargado de la construcción de la clase.

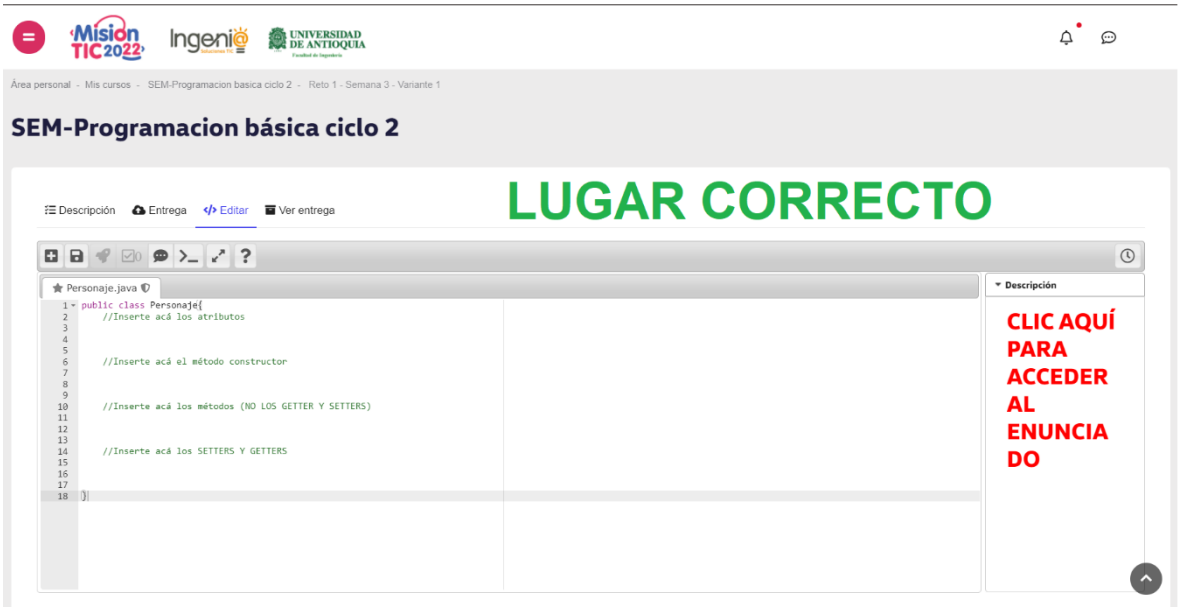
NOTA ACLARATORIA

Usted podrá desarrollar la clase requerida en un IDE como NetBeans, y al final copiar y pegar el código en la herramienta VPL, pero **NO** deberá subir archivos, es decir:

Modo incorrecto:



Modo correcto:



EJEMPLO

El calificador automático hará las veces de cliente, y será quien evalúe la experiencia de usuario que tuvo al usar el prototipo:

1. El calificador compra una SIMCard, por lo que se crea un objeto asociado a esa SIMCard

```
SIMCard cliente = new SIMCard("3015328969");
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	0
numeroTelefono	"3015328969"
celularApagado	true
modoAvionActivado	false
datosActivados	false
saldoDatos	0

2. El calificador recarga `50000` pesos

```
cliente.recargarSaldo(50000);
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	50000
numeroTelefono	"3015328969"
celularApagado	true
modoAvionActivado	false
datosActivados	false
saldoDatos	0

3. El calificador compra 12 GB de datos

```
cliente.comprarDatos(12);
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	15000
numeroTelefono	"3015328969"
celularApagado	true
modoAvionActivado	false
datosActivados	false
saldoDatos	12

4. El calificador intenta usar 3 GB de plan de datos

```
cliente.consumirDatos(3);
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	15000
numeroTelefono	"3015328969"
celularApagado	true
modoAvionActivado	false
datosActivados	false
saldoDatos	12

Todo queda igual, porque no se puede consumir datos con el teléfono apagado.

5. El calificador enciende el celular y llama a un amigo (Se demora 125 segundos).

```
cliente.gestionarEncendidoCelular();
cliente.llamar(120);
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	14910
numeroTelefono	"3015328969"
celularApagado	false
modoAvionActivado	false
datosActivados	false
saldoDatos	12

6. El calificador enciende el modo avión y enciende los datos

```
cliente.gestionarModoAvion();
cliente.gestionarDatos();
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	14910
numeroTelefono	"3015328969"
celularApagado	false
modoAvionActivado	true
datosActivados	false
saldoDatos	12

Note que los datos móviles NO se encendieron.

7. El calificador apaga el modo avión y enciende los datos móviles

```
cliente.gestionarModoAvion();
cliente.gestionarDatos();
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	14910
numeroTelefono	"3015328969"
celularApagado	false
modoAvionActivado	false
datosActivados	true
saldoDatos	12

8. El calificador consume 3 GB de datos

```
cliente.consumirDatos(3);
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	14910
numeroTelefono	"3015328969"
celularApagado	false
modoAvionActivado	false
datosActivados	true
saldoDatos	9

9. El calificador apaga el celular

```
cliente.gestionarEncendidoCelular();
```

NOMBRE	CONTENIDO
empresaTelefonia	"HI"
saldo	14910
numeroTelefono	"3015328969"
celularApagado	true
modoAvionActivado	false
datosActivados	false
saldoDatos	9

Note que apaga los datos móviles.