

## Proyecto UT3 (para hacer en casa y entregar en GitHub)

### Objetivos

Saber:

- definir y utilizar constantes y atributos (variables de instancia) dentro de una clase
- definir constructores
- definir y utilizar parámetros en los constructores y métodos
- definir métodos accesorios y mutadores
- usar las sentencias de asignación y escritura
- construir una sentencia *if*
- construir una sentencia *switch*
- expresar el algoritmo correspondiente a un método
- usar operadores aritméticos y relacionales

### Antes de empezar

- Este ejercicio es para realizar de forma **individual** en casa.
- El proyecto de partida está en <https://github.com/progdaw1/ENTRE-UT3-Parking>. Deberás hacer un *fork* a tu cuenta y clonarlo en tu PC desde BlueJ tal y como se explicó en clase
- Una vez completado desde BlueJ haz un *push* del último *commit* a GitHub
- No olvides **entregar vía Moodle el texto de la actividad “Terminado proyecto UT3 Parking”** y pulsar *Enviar para calificar*
- Se valorará en la corrección que el programa esté probado (compila y ejecuta bien) y que esté claramente escrito y organizado (se respetan las reglas de estilo del lenguaje Java, nombres descriptivos, código no duplicado, ...)

- La fecha tope de entrega es el **Miércoles 14 Octubre** hasta las **23,59h**.

- Se anulará automáticamente la corrección del ejercicio y se **evaluará con un 0** si:
  - ➔ se detecta que ha sido copiado o dejado copiar a algún compañero/a
  - ➔ no se siguen las normas de entrega del ejercicio
    - × no se ha hecho un *fork* / no se sube vía *commit*
    - × hay algún *commit* posterior a la fecha tope de entrega
    - × no se ha enviado el texto de la actividad vía Moodle

### Especificaciones

En este proyecto vamos a modelar una clase que describe el funcionamiento de un parking de una ciudad europea. El parking dispone de dos tarifas de aparcamiento: una tarifa regular (que incluye una tarifa plana para entradas / salidas tempranas) y una tarifa comercial para usuarios que trabajan cerca del parking, aparcan un nº elevado de horas y se benefician de esta tarifa más económica. Las tarifas se describen en detalle después.

El parking además realiza ciertos cálculos con el fin de obtener estadísticas de uso del mismo.

Haz el *fork* del proyecto **ENTRE-UT3-Parking** desde <https://github.com/progdaw1> a tu cuenta GitHub y desde BlueJ clona el proyecto a tu PC.

Abre el proyecto BlueJ. **Tienes que completar únicamente la clase Parking**. La clase DemoParking no tienes que modificarla, te servirá para probar la otra.

Define dentro de la clase `Parking` las siguientes constantes y atributos (deduce los tipos de datos adecuados):

- dos constantes que indican el tipo de tarifa
  - `REGULAR` con el valor asociado 'R'
  - `COMERCIAL` con el valor asociado 'C'
- tres constantes
  - `PRECIO_BASE_REGULAR` con el valor asociado 2.0 (indica el precio tarifa base regular)
  - `PRECIO_MEDIA_REGULAR_HASTA11` con el valor asociado 3.0 (indica el precio a pagar por cada media hora completa hasta las 11:00)
  - `PRECIO_MEDIA_REGULAR_DESPUES11` con el valor asociado 5.0 (indica el precio a pagar por cada media hora completa después de las 11:00)
- cinco constantes
  - `HORA_INICIO_ENTRADA_TEMPRANA` con el valor  $6 * 60$  (indica la hora de comienzo en minutos a partir de la cual se considera entrada temprana al parking – 6:00h.)
  - `HORA_FIN_ENTRADA_TEMPRANA` con el valor  $8 * 60 + 30$  (indica la hora en minutos a partir de la cual se considera fin entrada temprana al parking – 8:30h.)
  - `HORA_INICIO_SALIDA_TEMPRANA` con el valor asociado  $15 * 60$  (indica la hora en minutos a partir de la cual se considera inicio salida temprana del parking – 15:00h.)
  - `HORA_FIN_SALIDA_TEMPRANA` con el valor asociado  $18 * 60$  (indica la hora en minutos a partir de la cual se considera fin salida temprana del parking – 18:00h.)
  - `PRECIO_TARIFA_PLANA_REGULAR` con el valor asociado 15.0 (indica el precio tarifa plana que se cobra cuando se entra entre las 6:00h. y 8:30h. y se sale entre las 15:00h. y las 18:00h.)
- dos constantes
  - `PRECIO_PRIMERAS3_COMERCIAL` con el valor asociado 5.00 (es el precio que se cobra por las tres primeras horas en tarifa comercial)
  - `PRECIO_MEDIA_COMERCIAL` con el valor asociado 3.00 (precio de cada media hora completa en tarifa comercial)
- los siguientes atributos o variables de instancia
  - *nombre* - guarda el nombre del parking
  - *cliente* - guarda el nº de cliente que hace uso del parking (lo genera el programa)
  - *importeTotal* - guarda el importe total facturado por el parking entre todos los clientes
  - *regular* - almacena la cantidad de clientes con tarifa regular (sea temprana o no) que usaron el parking
  - *comercial* - almacena la cantidad de clientes con tarifa comercial que usaron el parking
  - *clientesLunes* - guarda el nº de clientes que usaron el parking el lunes
  - *clientesSabado* - guarda el nº de clientes que usaron el parking el sábado
  - *clientesDomingo* - guarda el nº de clientes que usaron el parking el domingo
  - *clienteMaximoComercial* - nº de cliente en tarifa comercial que ha pagado un importe mayor
  - *importeMaximoComercial* - importe máximo pagado por un cliente en tarifa comercial

**No debes incluir más atributos.** Solo los indicados. Respeta los nombres que se te dan

Completa los siguientes métodos:

- el **constructor**, recibe un parámetro, el nombre del parking. Inicializa el resto de atributos a 0
- incluye un accesor y un mutador para el nombre del parking
- el método **void facturarCliente(char tipoTarifa, int entrada, int salida, int dia)** – recibe cuatro parámetros que **supondremos correctos**:
  - *tipoTarifa* - un carácter 'R' o 'C'
  - *entrada* - hora de entrada al parking – un nº entero de hasta 4 dígitos que representa la hora de entrada al parking. Los dos más significativos representan la hora y los dos menos significativos los minutos (hora de 24h.)  
**Por ej.** 1235 significa que ha entrado a las 12:35h.  
930 significa que ha entrado a las 9:30h.  
1315 significa que ha entrado a las 13:15h.
  - *salida* – hora de salida del parking - un nº entero de hasta 4 dígitos que representa la hora de salida del parking.  
**Por ej.** 1607 significa que ha salido a las 16:07h.  
722 significa que ha salido a las 7:22h.  
1300 significa que ha salido a las 13:00h.
  - *dia* – nº de día de la semana (un valor entre 1 y 7)

A partir de estos parámetros el método debe calcular el importe que debe pagar el cliente y mostrarlo en pantalla y **actualizará adecuadamente** el resto de atributos del parking para poder mostrar posteriormente (en otro método) las estadísticas.

Para calcular el importe que ha de pagar el cliente se tendrá en cuenta:

- a) tarifa **regular** – salvo que se trate de una entrada/salida temprana esta tarifa cuesta
- 2€ tarifa base más
  - 3€ por cada media hora completa antes de las 11:00h. más
  - 5€ por cada media hora completa después de las 11:00h.

Se considera *entrada/salida temprana* si el usuario entra al parking entre las 6:00h. y las 8:30 y sale entre 15:00h. y las 18:00h. En este caso se le aplica la tarifa plana de 15€.

- b) tarifa **comercial** - las primeras tres horas valen 5€. Después de tres horas se paga cada media hora completa adicional a 3€

Por simplicidad consideraremos que un cliente entra y sale en un mismo día.

Utiliza una sentencia **switch** para analizar el tipo de tarifa.

Usa las constantes que has declarado.

El código resultará más sencillo si se **convierten todas las horas a minutos**.

Presenta los resultados tal como se indica (*ver figura 1*).

**Evita repetir código**

Si llamas al método `facturarCliente('R', 830, 1607, 2)` deberás ver en pantalla:

```
*****
Cliente nº: 1
Hora entrada: 8:30
Hora salida: 16:07
Tarifa a aplicar: REGULAR y TEMPRANA
Importe a pagar: 15.0€
*****
```

(figura 1)

Si ahora llamas al método `facturarCliente('C', 1235, 1752, 1)` deberás ver en pantalla:

```
*****
Cliente nº: 2
Hora entrada: 12:35
Hora salida: 17:52
Tarifa a aplicar: COMERCIAL
Importe a pagar: 17.0€
*****
```

Si ahora llamas al método `facturarCliente('R', 415, 722, 1)` deberás ver en pantalla:

```
*****
Cliente nº: 3
Hora entrada: 4:15
Hora salida: 7:22
Tarifa a aplicar: REGULAR
Importe a pagar: 20.0€
*****
```

- el método `printEstadísticas()` - muestra en pantalla las estadísticas sobre el parking. Si después de ejecutar el método `facturarCliente()` las 3 veces que se indican anteriormente llamas a `printEstadísticas()` el resultado en pantalla debe ser:

```
*****
Importe total entre todos los clientes: 52.0€
Nº clientes tarifa regular: 2
Nº clientes tarifa comercial: 1
Cliente tarifa COMERCIAL con factura máxima fue el nº 2
y pagó 17.0€
*****
```

- el método `String diaMayorNumeroClientes()` - devuelve el nombre del día en el que más usuarios han utilizado el parking - "SÁBADO" "DOMINGO" o "LUNES"

Si después de ejecutar el método `facturarCliente()` las 3 veces que se indican anteriormente llamas a `diaMayorNumeroClientes()` el resultado debe ser "LUNES"

## Posible ejecución

Para probar la aplicación completa:

- a) crea un objeto de la clase `DemoParking`
- a) llama al método `iniciar()`

Tendrás que obtener los resultados de la figura:

<p>Parking: Sky Park</p> <p>*****</p> <p>Cliente nº: 1  Hora entrada: 8:30  Hora salida: 16:07  Tarifa a aplicar: REGULAR y TEMPRANA  Importe a pagar: 15.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 2  Hora entrada: 12:35  Hora salida: 17:52  Tarifa a aplicar: COMERCIAL  Importe a pagar: 17.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 3  Hora entrada: 4:15  Hora salida: 7:22  Tarifa a aplicar: REGULAR  Importe a pagar: 20.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 4  Hora entrada: 9:35  Hora salida: 12:40  Tarifa a aplicar: REGULAR  Importe a pagar: 23.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 5  Hora entrada: 13:15  Hora salida: 16:50  Tarifa a aplicar: REGULAR  Importe a pagar: 37.0€</p> <p>*****</p>	<p>*****</p> <p>Cliente nº: 6  Hora entrada: 9:30  Hora salida: 13:00  Tarifa a aplicar: COMERCIAL  Importe a pagar: 8.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 7  Hora entrada: 11:15  Hora salida: 15:12  Tarifa a aplicar: COMERCIAL  Importe a pagar: 8.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 8  Hora entrada: 7:25  Hora salida: 22:35  Tarifa a aplicar: COMERCIAL  Importe a pagar: 77.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 9  Hora entrada: 7:30  Hora salida: 16:40  Tarifa a aplicar: REGULAR y TEMPRANA  Importe a pagar: 15.0€</p> <p>*****</p> <p>*****</p> <p>Cliente nº: 10  Hora entrada: 9:30  Hora salida: 13:00  Tarifa a aplicar: REGULAR  Importe a pagar: 31.0€</p> <p>*****</p> <p>*****</p> <p>Importe total entre todos los clientes: 251.0€  Nº clientes tarifa regular: 6  Nº clientes tarifa comercial: 4  Cliente tarifa COMERCIAL con factura máxima fue el nº 8  y pagó 77.0€</p> <p>*****</p> <p>El día que más clientes usaron el parking fue LUNES</p> <p>Pulse &lt;Intro&gt; para continuar</p>
--	--

Rúbrica evaluación	
ctes / atributos	8,80
constructor	2,00
accesor / mutador	6,00
facturarCliente	54,00
printEstadisticas	8,20
diaMayor NumeroClientes	13,00
buen estilo programación	8,00
	100,00
Penalización (no compila)	-0,5 (sobre 10)