Tarea online

Título de la tarea: Trabajando con clases, objetos y métodos.

Unidad: 03

Ciclo formativo y módulo: DAM/DAW, Programación.

Curso académico: 2025/2026

¿Qué contenidos o resultados de aprendizaje trabajaremos?

El principal resultado de aprendizaje que se va a trabajar con esta tarea será:

- RA2. Escribe y prueba programas sencillos, reconociendo y aplicando los fundamentos de la programación orientada a objetos.

Esto significa que se trabajará esencialmente con los siguientes contenidos:

- Creación de objetos.
- Uso de métodos y propiedades de los objetos.
- Uso de constructores
- Uso de parámetros en llamadas a métodos
- Llamadas a métodos estáticos.

1.- Descripción de la tarea.

Caso práctico

Gerd Altmann (Pixabay License)

María está entrenándose en la programación orientada a objetos. Está empezando a realizar pruebas, creando objetos, invocando métodos, consultando propiedades, etc. En fin, familiarizándose poco a poco con el lenguaje Java y algunas de las bibliotecas o librerías que lleva incorporadas el paquete básico de desarrollo.

Ahora mismo está construyendo una pequeña aplicación para poder usar algunas clases que han realizado algunos de sus compañeros (clase **`Velero`**). También está practicando con el uso de la **documentación javadoc** que se ha generado para esa clase y está observando lo útil que puede llegar a ser disponer de este tipo de documentación. ¡Le están literalmente "salvando la vida"!

¿Qué te pedimos que hagas?

Las actividades a realizar se centrarán en el uso y manipulación de objetos mediante operaciones sencillas, trabajando con métodos de la clase **`Velero`** para obtener una serie de resultados, **siendo obligatorio el uso de objetos de esa clase**. Para ello, es vital que consultes la **documentación javadoc** de esa clase. Esa información podrás encontrarla a través de la documentación javadoc específica que se te proporciona en esta tarea (para el caso de las clase **`Velero`**).

Debes usar obligatoriamente el proyecto NetBeans que se te proporciona (**`Controlador`**) en el apartado **Información de interés** de esta tarea, el cual incluye una **biblioteca específica para esta tarea** con la clase **`Velero`**, que no forma parte de la API de Java.

1.1.- Ejercicio 1. Controlador.

La clase `Velero` implementa un modelo de gestión que permite controlar la actividad de cada velero así como sus navegaciones y regatas. Es importante que le eches un vistazo a la **documentación javadoc** de esta clase antes de empezar a trabajar con ella. Así podrás ver qué tipo de información contiene, qué constructores proporciona, qué restricciones se le pueden aplicar, cuáles son los métodos disponibles, qué información puedes obtener a partir de un objeto, qué información puedes obtener a partir de la clase (métodos estáticos), etc.

mariya_m. *Velero* (Licencia Pixabay)

Recuerda que tendrás que hacer el `import` correspondiente para poder usar estas clases en tu programa.

Para practicar con la creación y utilización de objetos de este tipo, vamos a trabajar con **cuatro instancias del tipo `Velero`.**

- **Para cada velero se debe indicar su nombre, número de mástiles y número de tripulantes.** (En caso de no indicar estos parámetros, se asignarán sus atributos por defecto).

> Recuerda que para resolver esta actividad debes utilizar obligatoriamente instancias de la clase `Velero` junto con sus métodos. Y no olvides que tendrás que hacer los `import` correspondientes para poder usar esas clases en tu programa.

En el proyecto de trabajo que se proporciona para esta tarea dispones de una clase `Controlador` en la que deberás implementar lo siguiente:

Declarar cuatro variables referencia a objetos instancia de la clase `Velero`.
Los **cuatro veleros** serán:
`velero1` creado con el constructor por defecto.
`velero2` creado con los parámetros nombre *Poseidón*, número de mástiles *3* y número de tripulantes *6.*
`velero3` creado con los parámetros nombre *Atlántico*, número de mástiles *3* y número de tripulantes *5*.
`veleroIncorrecto`: intentaremos instanciar un velero incorrecto dos veces, capturando los diferentes errores:
El nombre se establecerá como nulo *(null).*
El número de mástiles se establecerá a un valor incorrecto: *número negativo -5*.
Llevar a cabo una secuencia de instrucciones haciendo uso de los distintos métodos de la clase `Velero`, de manera que muestre en pantalla la información utilizando la herramienta de E/S `System.out.printf`:
`velero1` inicia su navegación con velocidad *10*, rumbo *ceñida,* patrón *Carlos Ruiz* y tripulación *0.*
`velero2` inicia su navegación con velocidad *15*, rumbo *empopada,* patrón *Laura Gómez* y tripulación *4.*
`velero3` inicia su navegación con velocidad *12*, rumbo *empopada,* patrón *Pedro Torres* y tripulación *2.*
Comprobar si `velero1` está navegando.
Mostrar la siguiente información de `velero2`:
Nombre.
Número de mástiles.
Mostrar la siguiente información de `velero3`:

Patrón.
Velocidad de navegación.

Modificar el rumbo de **`velero1`** a *empopada* y mostrarlo.
Parar la navegación de **`velero1`** después de *120* minutos.
Mostrar el tiempo total de navegación de **`velero1`**.
Iniciar una regata entre **`velero2`** y **`velero3`**.
Parar la navegación de **`velero2`** después de *90* minutos.
Parar la navegación de **`velero3`** después de *150* minutos.
Mostrar toda la información de **`velero1`**, **`velero2`** y **`velero3`** haciendo uso del método **`toString()`**.

Consulta final de valores globales de la clase `Velero`. Una vez finalizado el paso anterior, pasaremos a mostrar los valores globales haciendo uso de los métodos estáticos de la clase:

Número total de veleros creados.
Número total de veleros navegando en este momento.
Tiempo total de navegación de todos los veleros (en horas).

Resultado del ejercicio

Dado que no hay que introducir datos de entrada y todos los valores son fijos, la ejecución de tu programa siempre deberá proporcionar el mismo resultado. Aquí tienes una muestra de cómo podría quedar:

```
-------------------------
--- Creando Veleros -----
-------------------------

Creando veleros con datos correctos...
**Veleros creados correctamente**

Creando veleros con datos incorrectos/nulos...
Error al crear el velero: El nombre del velero no puede ser nulo.
Error al crear el velero: El número de mástiles debe estar entre 1 y 4.

--------------------------------
-- Secuencia de instrucciones --
--------------------------------

El velero1 ha iniciado su navegación.
El velero2 ha iniciado su navegación.
El velero3 ha iniciado su navegación.

¿El velero1 está navegando?: Sí

El nombre del velero2 es: Poseidón
El número de mástiles del velero2 es: 3

El patrón del velero3 es: Pedro Torres
La velocidad del velero3 es: 12 nudos

El nuevo rumbo del velero1 es: empopada
```

```
El velero1 ha parado su navegación.
Tiempo total de navegación del velero1: 120 minutos

Iniciando regata...
El barco Poseidón ha llegado antes a la línea de llegada.

El velero2 ha parado su navegación.
El velero3 ha parado su navegación.

------------------------------
--- Información de veleros ---
------------------------------

Velero1: {Nombre del barco: Velero 1, Número de mástiles: 1, Tripulación: 0, Nav
Velero2: {Nombre del barco: Poseidón, Número de mástiles: 3, Tripulación: 0, Nav
Velero3: {Nombre del barco: Atlántico, Número de mástiles: 3, Tripulación: 0, Na

-------------------------
--- Métodos estáticos ---
-------------------------

Número total de veleros: 3
Número de veleros navegando: 0
Tiempo total de navegación de todos los veleros: 6,00 horas
```

IMPORTANTE: uso de `System.out.printf`

Para la realización de **este ejercicio**, siempre que tengas que mostrar algo por pantalla, deberás hacerlo usando la herramienta de E/S `System.out.printf` en lugar de `System.out.println` o `System.out.print`. Es decir, tendrás que usar la salida con formato utilizando los indicadores de formato `%s, %d, %f,` etc.

El único caso en el que podrás usar `print` o `println` será cuando vayas a escribir por pantalla un literal de cadena (un texto entre comillas), sin mostrar el contenido de ninguna variable. En estos casos el uso de `printf` no ofrece ninguna ventaja adicional.

Recomendación

En el caso de crear los objetos de la clase `Velero` con datos no válidos o nulos hay que tener en cuenta que en estas invocaciones al constructor saltará una excepción de tipo `IllegalArgumentException` o `NullPointerException` respectivamente. En tales casos habrá que recogerla mediante una estructura de tipo `try-catch` y mostrar el mensaje de error por pantalla usando la herramienta de E/S `System.out.printf`.

2.- Información de interés.

Recursos necesarios y recomendaciones

Debes usar obligatoriamente el proyecto base NetBeans que te proporcionamos aquí. Ese proyecto incluye una **biblioteca específica para esta tarea (`libtarea3`)** que contiene la clase **`Velero`**. Esa clase no forma parte de la API de Java.

Mohamed Hassan (Licencia Pixabay)

Es vital que utilices este proyecto porque si no lo haces no podrás implementar correctamente tus programas al no disponer de esa clase ni de la configuración necesaria de tus bibliotecas.

Además, este proyecto contiene un programa (clase **`Controlador`**) correspondiente al ejercicio que debes implementar. La clase controlador tendrá su propio **`main`** con el esqueleto del programa y algunas indicaciones de las pautas que debes seguir y así orientarte un poco el trabajo que debes realizar. ¡Aprovéchalos!

Descarga y utiliza este proyecto. ¡No crees tu propio proyecto en NetBeans! Si lo haces, tu tarea será devuelta pues no tendrá las características que necesitamos que tenga.

Descargar proyecto base NetBeans para la tarea 3 (zip - 129 KB) .

Una vez descargado el proyecto, renómbralo con el estilo de nombre que solemos darle al paquete de entrega de la tarea:

Apellido1_Apellido2_Nombre_PROG03_Tarea

Y ya podrás ponerte a trabajar sobre él. ¡Ánimo!

Aunque en el proyecto base que se te proporciona ya contiene la **documentación javadoc** de la clase **`Velero`**. Te las proporcionamos también aparte por si quieres disponer de ellas para consultarlas independientemente del IDE NetBeans:

Descargar javadoc de la clase Velero para la tarea 3 (zip - 91 KB)

Una vez descargado y descomprimido el paquete debes abrir con un navegador web el archivo **index.html** para empezar a navegar por el javadoc de esta API y consultar la documentación de la clase.

Indicaciones de entrega

Una vez completados todos los ejercicios de la tarea, el envío se realizará a través de la plataforma. Comprime la carpeta del proyecto NetBeans en un fichero .zip y nómbralo siguiendo las siguientes pautas:

Apellido1_Apellido2_Nombre_PROG03_Tarea

Recuerda que **tu proyecto NetBeans también debe llamarse así** (tanto la carpeta donde se encuentra el proyecto como el nombre del proyecto en Netbeans tienes que renombrarlos respecto al nombre original del proyecto base que vas a utilizar). **Si no lo haces, todos los proyectos se llamarán igual y la corrección de tu tarea podría confundirse con la de otra persona**.

Por tanto, **si no entregas tu tarea siguiendo estas reglas, se te devolverá para que lo hagas correctamente** .

3.- Evaluación de la tarea.

Criterios de evaluación implicados

Del **RA2** (*Escribe y prueba programas sencillos, reconociendo y aplicando los fundamentos de la programación orientada a objetos*):

- a) Se han identificado los fundamentos de la programación orientada a objetos.
- b) Se han instanciado objetos a partir de clases predefinidas.
- c) Se han utilizado constructores.
- d) Se han utilizado métodos y propiedades de los objetos.
- e) Se han escrito llamadas a métodos estáticos.
- f) Se han utilizado parámetros en la llamada a métodos.
- g) Se han incorporado y utilizado librerías de objetos.
- h) Se han escrito programas simples.
- i) Se ha utilizado el entorno integrado de desarrollo en la creación y compilación de programas simples.

Peggy_Marco (Pixabay License)

¿Cómo valoramos y puntuamos tu tarea?

Cada uno de los CE cubiertos por esta tarea será evaluado a través de los criterios de corrección y puntuación indicados detalladamente en la rúbrica de la tarea que puedes consultar a continuación:

ÍTEMS	PUNTUACIONES			
(RA2.a, RA2.g, RA2.h, RA2.i) Se ha importado la librería con la que se trabaja en la tarea.	No se ha importado ni utilizado la librería. **0 puntos**	Se ha importado y utilizado la librería correctamente. **0.5 puntos**		
(RA2.a, RA2.b, RA2.c, RA2.h, RA2.i) Se han declarado e inicializado objetos usando parámetros correctos.	No se han instanciado los objetos solicitados. **0 puntos**	Se han instanciado los objetos usando parámetros correctos con muchos errores. **0.5 puntos**	Se han instanciado los objetos usando parámetros correctos con algún error. **1 punto**	Se han instanciado todos los objetos usando parámetros correctos de forma correcta. **1.5 puntos**
(RA2.a, RA2.b, RA2.c, RA2.h, RA2.i) Se han inicializado objetos usando parámetros incorrectos o nulos, capturando los errores.	No se han instanciado los objetos usando parámetros incorrectos ni nulos. **0 puntos**	Se han instanciado los objetos solo con parámetros incorrectos o nulos. **0.5 puntos**	Se han instanciado los objetos con parámetros incorrectos y nulos. **1 punto**	
(RA2.a, RA2.d, RA2.f, RA2.h, RA2.i) Se han utilizado correctamente los métodos de acción de los objetos (iniciar y parar la navegación).	No se han utilizado los métodos de acción de los objetos. **0 puntos**	Se han utilizado los métodos de acción de los objetos con muchos errores. **0.5 puntos**	Se han utilizado los métodos de acción de los objetos con algún error. **1 punto**	Se han utilizado los métodos de acción de los objetos de forma correcta. **1.5 puntos**
(RA2.a, RA2.d, RA2.f, RA2.h, RA2.i) Se han utilizado correctamente los métodos de acción de los objetos (cambiar rumbo).	No se han utilizado los métodos de acción de los objetos. **0 puntos**	Se han utilizado los métodos de acción de los objetos con muchos errores. **0.35 puntos**	Se han utilizado los métodos de acción de los objetos con algún error. **0.75 puntos**	Se han utilizado los métodos de acción de los objetos de forma correcta. **1.25 puntos**
(RA2.a, RA2.d, RA2.f, RA2.h, RA2.i) Se han utilizado correctamente los métodos de acción de los objetos (iniciar regata).	No se han utilizado los métodos de acción de los objetos. **0 puntos**	Se han utilizado los métodos de acción de los objetos con algún error. **0.35 puntos**	Se han utilizado los métodos de acción de los objetos de forma correcta. **0.75 puntos**	
(RA2.a, RA2.d, RA2.f, RA2.h, RA2.i) Se han utilizado correctamente los métodos para obtener algún atributo de los objetos (getters).	No se han utilizado los métodos para obtener algún atributo de los objetos. **0 puntos**	Se han utilizado los métodos para obtener algún atributo de los objetos con muchos errores. **0.5 puntos**	Se han utilizado los métodos para obtener algún atributo de los objetos con algún error. **1 punto**	Se han utilizado los métodos para obtener algún atributo de los objetos de forma correcta. **1.5 puntos**

(RA2.a, RA2.d, RA2.f, RA2.h, RA2.i) Se han utilizado correctamente los métodos para mostrar toda la información de los objetos (toString).	No se han utilizado los métodos para mostrar toda la información de los objetos. **0 puntos**	Se han utilizado los métodos para mostrar toda la información de los objetos con algún error. **0.25 puntos**	Se han utilizado los métodos para mostrar toda la información de los objetos de forma correcta. **0.5 puntos**	
(RA2.a, RA2.e, RA2.h, RA2.i) Se han utilizado correctamente llamadas a métodos estáticos.	No se han utilizado llamadas a métodos estáticos. **0 puntos**	Se han utilizado llamadas a métodos estáticos con muchos errores. **0.35 puntos**	Se han utilizado llamadas a métodos estáticos con algún error. **0.75 puntos**	Se han utilizado llamadas a métodos estáticos de forma correcta. **1.25 puntos**
(RA2.a, RA2.b, RA2.c, RA2.d, RA2.e, RA2.f, RA2.g, RA2.h, RA2.i) Se ha comentado y documentado el código del programa.	No se han realizado comentarios. **0 puntos**	Se han realizado algunos comentarios, sin explicar procesos importantes. **0.125 puntos**	Se han realizado comentarios de manera adecuada para comprender fácilmente todo el código implementado. **0.25 puntos**	

Tanto en la tarea en general como en la actividad que tenga que desarrollar en particular, se tendrán en cuenta los siguientes puntos de control o elementos evaluables a la hora de corregir tu tarea:

- Se ha creado un único proyecto con todos los ejercicios en él utilizando **el entorno NetBeans 16 y Java Development Kit 17 (JDK)**.
- Los programas **compilan**.
- Se sigue la **estructura** propuesta de **declaración de variables**, **entrada de datos**, **procesamiento** y **salida de resultados**, tal y como se indica en la plantilla y el enunciado.
- Se declaran las **variables necesarias** con el **tipo adecuado** y con **nombres apropiados**.
- No se incluyen **elementos superfluos** ni **innecesarios**, ni se llevan a cabo **operaciones sin sentido**.
- Se declaran **identificadores** que con **nombres significativos o descriptivos** que representen de alguna manera la información que están almacenando para que **el código quede lo más claro, legible y autodocumentado posible**.
- Se llevan a cabo los **cálculos y comprobaciones necesarias** para llevar a cabo las tareas que se pide en cada actividad. Para ello habrá que construir **expresiones apropiadas** y hacer uso de los **operadores correctos**.
- El código de los programas está correctamente **indentado**.
- El código está apropiadamente **comentado**. Se observa una **corrección ortográfica y gramatical**, así como la **coherencia en las expresiones lingüísticas** en los comentarios incluidos.
- Los **identificadores** cumplen con el **convenio** sobre **"*asignación de nombres a identificadores*"** establecido para el lenguaje Java para constantes, variables, métodos, clases, etc.
- En los programas **se muestra la información esperada y correcta por pantalla en un formato apropiado**. Se observa una **corrección ortográfica y gramatical,** así como la coherencia en las expresiones lingüísticas, tanto en los textos de los mensajes que aparezcan en pantalla para pedir información de entrada al usuario como a la hora de mostrar resultados de salida. Se evitan mensajes de entrada de datos y/o salida de resultados inapropiados, descontextualizados, insuficientes o incorrectos.

Aparte de todos estas cuestiones de carácter "técnico", también se valorará en cada uno de los apartados:

- que la respuesta sea personal y no copiada de Internet o de otras fuentes;
- que sea coherente con el resto de respuestas y con los conceptos de la unidad;
- que muestre que se comprenden adecuadamente los conceptos tratados.

IMPORTANTE: Motivos de devolución de la tarea o de calificación mínima

La tarea obtendrá automáticamente una calificación de **cero** si:

- No se entrega **un único proyecto** que incluya los dos ejercicios que se piden en la tarea utilizando el IDE NetBeans.
- Alguno de los programas presenta **errores de compilación** impidiendo su ejecución y prueba.
- **No se respeta el proyecto base** que se entrega al alumno para realizar la tarea.
- Se utilizan **contenidos** en alguno de los ejercicios del proyecto **que no se han estudiado** en la unidad actual o anteriores a la misma.
- Se presenta un proyecto total o parcialmente copiado de otro alumno.

Recordatorio: Si una tarea obtiene calificación cero por alguno de los motivos previamente mencionados, será devuelta al alumno. Este tendrá derecho a una segunda entrega únicamente en los siguientes casos:

- Que se trate de la primera vez que presenta la tarea.
- Que la fecha de entrega inicial haya sido al menos una semana antes de la fecha límite establecida para dicha tarea.