ILERNA Online

PAC DESARROLLO

CFGS Desarrollo de Aplicaciones

Módulo 03B:

Programación

1S 2019/2020

Profesor: Alvaro Ortega Marmol

INFORMACIÓN IMPORTANTE

Para la correcta realización de la PAC el alumno deberá consultar los contenidos recogidos en el **Tema 1, Tema 2, Tema 3 y Tema 4** del material didáctico.

Requisitos que deben cumplirse en vuestros trabajos:

• Siempre que utilicéis información de Internet para responder / resolver alguna pregunta, tenéis
que citar la fuente (la página web) de dónde habéis sacado aquella información.
• No se aceptarán respuestas sacadas de Internet utilizando la metodología de copiar y pegar. Podéis utilizar Internet para localizar información, pero el redactado de las respuestas ha de ser vuestro.
• Las respuestas a las preguntas deben estar bien argumentadas, no se admiten respuestas escuetas o monosílabas.
•La PAC debe entregarse en **formato ZIP o mediante enlace a github.**
•Este ZIP, contendrá el proyecto **realizado en Java**
•En el caso de **no** realizarse la entrega en dicho formato **el alumno se hace responsable** de posibles incompatibilidades en la visualización de su entrega y por ende afectará a su calificación.

CRITERIOS DE CORRECCIÓN

1. Todos los programas realizados en la PAC deben realizarse con IDE con que se pueda trabajar con el lenguaje Java
2. Para la realización de esta PAC es necesario que se utilicen las estructuras de control y las estructuras repetitivas siempre que sea posible.
3. Se deben poner comentarios para su mejor comprensión. Estos comentarios explicarán la funcionalidad del código. Se valorarán los comentarios en la parte de presentación.
4. No se permite el uso de ficheros

Introducción:

El enunciado de la práctica está inspirado en el siguiente problema que apareció en la revista Communications of the ACM, Vol. 52, No. 8 de agosto de 2009:

Cien personas embarcan en un avión cuyas plazas han sido vendidas en su totalidad. Desgraciadamente, la primera persona de la cola ha perdido su tarjeta de embarque al entrar en el avión y se sienta en un asiento elegido al azar. Los pasajeros siguientes se sentarán en el asiento indicado por su tarjeta si éste está libre o bien harán como el primer pasajero y elegirán un asiento libre al azar. ¿Cuál es la probabilidad de que el último pasajero encuentre el asiento correspondiente a su tarjeta de embarque libre?

Clases necesarias:

Clase Pasajero:

Representa a un único pasajero, esta tendrá 3 atributos.

1. Nombre.
2. Número del asiento asignado.
3. Variable booleana para comprobar si ha perdido la tarjeta o no.

El constructor de la clase recibirá el nombre del pasajero y su asiento asignado.

La clase incluirá los diferentes métodos Setter i Getters y la función toString().

Clase Pasajeros:

Representa al conjunto de pasajeros que se subirán al avión.

Esta clase tiene como **mínimo** 1 atributo (en caso de necesitar más añadirlos) que será una lista de los pasajeros.

El constructor de la clase recibirá el número de pasajeros totales.

Por otro lado, nos encontremos diferentes funciones:

tieneMasPasajeros()

La cual nos permitirá saber si tenemos más pasajeros en la lista.

siguientePasajero()

La cual nos devolverá el siguiente pasajero de la lista.

addPasajero(Pasajero pasajero)

La cual nos permitirá añadir pasajeros a la lista.

desordenar()

La cual nos desordenará la lista de pasajeros.

Clase Avion

Esta clase es la que mejor deberéis pensar. Tiene un constructor el cual recibe las plazas que tiene el avión y por tanto cuantos pasajeros podrá haber. Los atributos necesarios los tendréis que pensar vosotros, forma parte del análisis de clases.

La clase avión tendrá la **función obligatoria embarque** la cual recibirá la lista de pasajeros que se van a embarcar en el avión.

Ejercicio 1:

Crea la clase Pasajero con sus atributos y métodos correspondientes.

Ejercicio 2:

Crea la clase Pasajeros con sus atributos y métodos correspondientes.

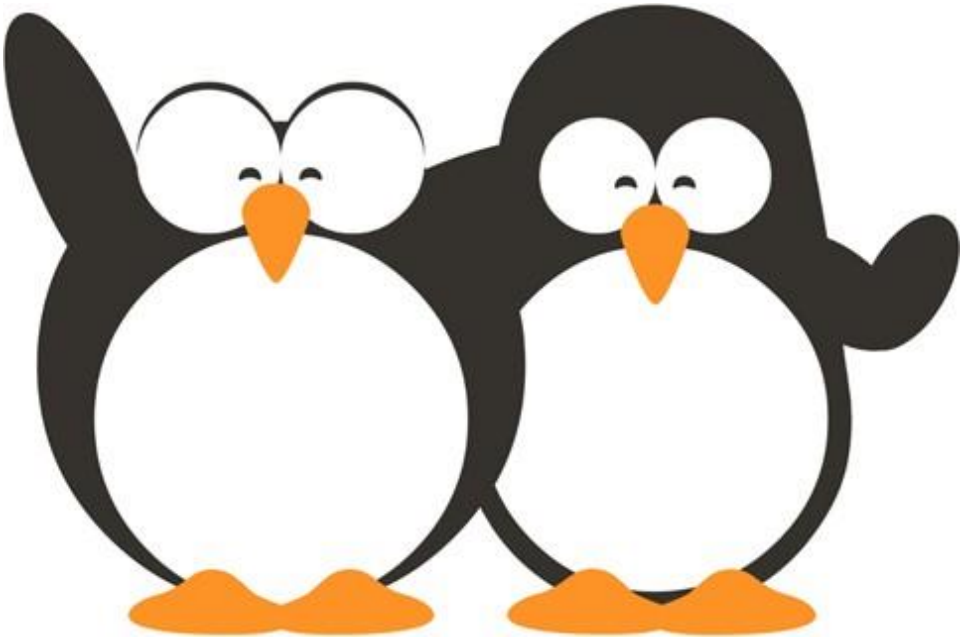
Ejercicio 3:

Crea la clase Avion, la cual tendrá toda la lógica de nuestra aplicación.
La función de la clase avión consistirá en embarcar a los pasajeros y comprobar si tienen tarjeta de embarque, en caso de no tenerla buscará un nuevo asiento aleatorio que esté libre. Por otro lado, si el pasajero tiene tarjeta de embarque y su asiento está ocupado por una persona que se ha sentado previamente también deberá buscar otro asiento libre de forma aleatoria.
Finalmente, cuando se mire el último pasajero nos mostrará por pantalla si su (el que tiene en la tarjeta de embarque) coincide con el último asiento libre.

Ejercicio 4:

Crea la clase main, en ella deberás crear los diferentes pasajeros y el avión, para hacer el ejercicio más simple el número de pasajeros siempre será igual al numero de plazas en el avión.
Cuando hayamos creado todos los pasajeros deberemos desordenar la lista y asignar al pasajero que se encuentra en la posición 0 la tarjeta de embarque perdida.
Esta lista desordenada es la que le pasaremos a la función embarque de la clase Avion.

¡Buen trabajo!



Profesor: Alvaro Ortega Marmol