Silvio Orozco Vizquerra
Carné 18282
Fecha 02/08/2018

Laboratorio 1

Funcionalidad del Sistema y Clases:

El gobierno de EE. UU. deporta una gran cantidad de inmigrantes a Guatemala. Diariamente llegan 3 vuelos con personas deportadas a Guatemala.
Se necesita un sistema que gestione la información de estas personas:
Se debe modelar los vuelos con esta información:

- Porcentaje de cantidad de mujeres, hombres y niños deportados por el vuelo.
- Cantidad total de personas que arribaron en todos los vuelos (Static).
- Mostrar de que grupo arribaron más personas en el vuelo escogido.
- Advertencia de mensaje de preocupación media cuando la cantidad de niños que llegaron en el día es mayor a 8.

- Se deben crear 3 vuelos. (1 Clase vuelo y 3 instancias.)

  ### Clase vuelo:

  ### Objetivo de la clase:

  El objetivo de esta clase es ser quien gestiona la información de cada uno de los vuelos de inmigrantes deportados provenientes de los Estados Unidos. En él se gestionan los datos e información relevante perteneciente a estos vuelos, al igual que los métodos para ejecutar acciones como desplegar información, comparar información, reportes de estado, etc. Cada vuelo contiene información sobre cantidad personas de cada grupo (hombres int, mujeres int, niños int), numero de vuelo (string), piloto, copiloto, avión y aeropuerto.

  ### Propiedades:

  - Private int cantidadhombres; Se refiere a la cantidad numérica de personas del grupo hombres de solamente un vuelo. Sirve para realizar el reporte de cantidad y porcentajes.
  - Private int cantidadmujeres; Se refiere a la cantidad numérica de personas del grupo mujeres de solamente un vuelo. Sirve para realizar el reporte de cantidad y porcentajes.
  - Private int cantidadninos; Se refiere a la cantidad numérica de personas del grupo niños de solamente un vuelo. Sirve para realizar el reporte de cantidad y porcentajes. De igual forma, analiza la cantidad para mostrar un mensaje de advertencia.
  - Private int cantidadtotalvuelo; Se refiere a la cantidad numérica total de personas por vuelo. Sirve para realizar porcentajes sobre totales de cada grupo.
  - Public static int totalgrupos; Se refiere a la cantidad numérica total sumada de los 3 grupos de persona de cada vuelo. Es por ello que es static, pues aumenta con cada vuelo y todos los vuelos tienen acceso a dicha información sobre cantidad total.
  - Private int numerovuelo; Se refiere al número de vuelo para identificar el vuelo y compararlo a los demás.

- Private piloto pilot; Se refiere al atributo de tipo piloto que es una persona que pilotea la nave. Es una instancia de otro objeto que tiene también sus propias características.
- Private piloto copilot; Se refiere al atributo de tipo copiloto que es una persona que copilotea la nave. Es una instancia de otro objeto que tiene también sus propias características.
- Private avion nave; Se refiere al atributo avión que no es independiente del vuelo, puesto que el sistema no gestiona aviones. Es una instancia de otro objeto que tiene también sus propias características.
- Private aeropuerto puerto; Se refiere al atributo aeropuerto de llegada que no es independiente del vuelo, puesto que el sistema no gestiona aeropuertos. Es una instancia de otro objeto que tiene también sus propias características.

Métodos:

- Public vuelo(); Este es el método constructor que tendrá como parámetros los atributos iniciales del vuelo. Esto para el momento de crear un nuevo vuelo que tenga toda la información necesaria para utilizarse como una instancia de esta clase. La info recibida serán todos los atributos y asignara los atributos a la instancia del vuelo, sin retornar algún valor.
- Public string mostrarcantidades(); Este método muestra las cantidades de cada grupo del vuelo. Además, muestra un mensaje de advertencia si la cantidad de niños en el vuelo es mayor a 8. Sin parámetros y retorna el mensaje.
- Public string mostrarporcentajes(); Este método muestra los porcentajes de cada grupo del vuelo. Es decir convierte las cantidades en porcentajes y luego los muestra. Sin parámetros y retorna el mensaje.
- Public string mostrardatosvuelo(); Este método muestra los datos del vuelo, tales como el número del vuelo, el piloto, el copiloto, el avion utilizado y el aeropuerto. Sin parámetros y retorna el mensaje.
- Public string Totaldeldia(); Este método no recibe parámetros y solo retorna el total de pasajeros de todos los vuelos en conjunto.
- Public string piloto getCopiloto(); Este método retorna el copiloto de un vuelo.
- Public void SetCopiloto(); Este método es accedido por un piloto, el cual cambia el copiloto de un vuelo por otro copiloto, intercambiando a los mismos. Pedira como parametro otro copiloto para intercambiarlo.
- Public string mostrarreporte(); Este método retorna un mensaje sobre su estado para enviarlo a la torre de control, sin ningún parámetro.
- Public string utilizaravion(String funcionalidad); Este método sirve para acceder al avión del vuelo y ejecutar alguna de sus funciones según el parámetro de funcionalidad, retornando un mensaje con la ejecución de la misma..
  Los métodos descritos anteriormente sirven para mostrar resumenes de datos del vuelo.
- Public int gethombres(); Este método retorna la cantidad de hombres para ser comparada con la de otro vuelo.
- Public int getmujeres(); Este método retorna la cantidad de mujeres para ser comparada con la de otro vuelo.
- Public int getninos(); Este método retorna la cantidad de niños para ser comparada con la de otro vuelo.

- Public static string comparar(vuelo vuelo1, vuelo vuelo2, int opcion); Este método tiene como función comparar dos vuelos, entre alguno de los grupos según la opción elegida. Retornara un mensaje que indique la comparación.
- Public static string comparar(vuelo vuelo1, vuelo vuelo2, vuelo vuelo3, int opcion); Este método tiene como función comparar tres vuelos, entre alguno de los grupos según la opción elegida. Retornara un mensaje que indique la comparación.
- Public static string comparar(int cantidad1, int cantidad 2, int opcion); Este metodo permitira comparar las cantidades de algún grupo de 2 vuelos según la opción. Retornara un mensaje que indique la comparación.
- Public static string comparar(int cantidad1, int cantidad2, int cantidad3,int opcion); Este metodo permitira comparar las cantidades de algún grupo de 3 vuelos según la opción. Retornara un mensaje que indique la comparación.
  Los métodos mencionados anteriormente son para la comparación de vuelos y algunos de ellos son overloading y static para ser utilizados por la clase como medio indicador de comparación.

- Se deben crear 3 aviones (1 Clase Avión y 1 instancia por vuelo.) Composición
  Clase avion:
  Objetivo de la clase:
  El objetivo de esta clase es ser uno de los atributos de nuestra clase vuelo. Esto para que el vuelo tenga asignado un avion para usarse en los recorridos.

  Propiedades:

- Private string fabricante; Nombre del fabricante.
- Private string modelo; El modelo del avión.
- Private double alcance; El alcance del avión.
- private String dueño; Quién es el dueño del avion.
-

  Métodos:
  - Public avion(); Este es el método constructor que tendrá como parámetros los atributos iniciales del avion, que solo funcionará como una instancia del vuelo. Esto para el momento de crear un nuevo avion que tenga toda la info necesaria. Los parámetros serán fabricante, modelo, alcance y dueño, asignando los valores al nuevo avion. No tendrá ningún valor de retorno.
  - Public string volar(); Este método permitirá que el avión pueda volar desplegando un mensaje que ha volado.
  - Public string desplegarinfo(); Este método permitirá desplegar la información sobre el avión de algún vuelo.

- Se deben crear 3 pilotos y 3 copilotos (1 Clase Piloto y 1 Pareja para cada vuelo.) Agregación
  Cada uno de estos debe tener información sobre nombre, edad y horas de vuelo.
  Clase piloto:
  Objetivo de la clase:

El objetivo de esta clase es ser uno de los atributos de nuestra clase vuelo, pudiendo crear pilotos y copilotos para los vuelos. Este es independiente del sistema de vuelos pues puede ejecutar diversas funciones y no depende del mismo.

Propiedades:

- Private string nombre; Nombre del aviador.
- Private int edad; La edad del piloto.
- Private int horasdevuelo; Las horas de vuelo del aviador.
- Private String tipo; Si este es piloto o copiloto del avion.

Métodos:

- Public piloto(); Este es el método constructor que tendrá como parámetros los atributos iniciales del piloto, que solo funcionará como una instancia del vuelo. Esto para el momento de crear un nuevo avion que tenga toda la info necesaria. Los parámetros serán fabricante, modelo, alcance y dueño, asignando los valores al nuevo avion. No tendrá ningún valor de retorno.
- Public string mensajeBienvenida(); Este método permitirá que el piloto brinde un mensaje de bienvenida en el vuelo.
- Public string intercambiarcopilotos(vuelo vuelo1, vuelo vuelo2); Este método permitirá por medio de parámetros, intercambiar copilotos de dos vuelos en específico. El parámetro será un numero de opción que tendrá info sobre los dos vuelos que se desea intercambiar. Retorna un mensaje que el cambio fue hecho.
- Public string desplegarinfo(); Este método permitirá desplegar la información sobre el piloto de algún vuelo. Retorna un mensaje con la info.
- Public string entrenarpiloto(); Este método permite crear un nuevo piloto por medio del entrenamiento de un piloto posterior. Pedira los parámetros de nombre, edad y un valor de horas de vuelo=0 por defecto. Retorna un mensaje con la info que se ha creado un nuevo piloto.

Se debe crear 1 aeropuerto (1 Clase Aeropuerto y el mismo aeropuerto para todos los vuelos.)

Clase aeropuerto:

Objetivo de la clase:

El objetivo de esta clase es ser uno de los atributos de nuestra clase vuelo, pudiendo crear el aeropuerto para cada vuelo.

Propiedades:

- Private string nombre; Nombre del aeropuerto.
- Private string ubicacion; La ubicación del aeropuerto.

Métodos:

- Public aeropuerto(); Este es el método constructor que tendrá como parámetros los atributos iniciales del aeropuerto tales como nombre y ubicación.

- Public string mensajeBienvenida(); Este método permitirá que mostrara un mensaje de bienvenida al aeropuerto. Retorna el mensaje.
- Public string desplegarinfo(); Este método permitirá desplegar la información sobre el aeropuerto de un vuelo en especifico. Retorna el mensaje.

Clase driver:
Objetivo de la clase:

- Interactuar con el usuario para ejecutar las funcionalidades de la clase vuelo y sus derivadas. Ser el driver del programa.

No propiedades.
Metodo:
public static void main(String args[]): Dónde se ejecuta el programa, se hacen verificaciones y programación defensiva.

El requisito funcional del sistema es que debe de pedir todos los datos del usuario.
Se debe poder ingresar 3 vuelos, sus respectivos pilotos, aviones y aeropuertos.
Luego deberá poder mostrar los datos listados anteriormente, al igual que ejecutar cada uno de los métodos de las clases mencionadas.
Esto requiere ejecutar las funcionalidades del vuelo, del piloto y del avión.
Las funcionalidades son descritas en los métodos de cada clase.