Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

за 1 семестр

По дисциплине: «СПП»

Выполнил:

Студент 3 курса

Группы ПО-5

Лозейко П. А.

Проверил:

Крощенко А. А.

2021

Вариант 9



**Код программы:**

**Program.cs**

using System;

namespace lab5.\_1.\_9

{

class Program

{

static void Main(string[] args)

{

Tanker tanker = new Tanker("ULCC", 415, 500000, 200);

Console.WriteLine("Name: " + tanker.GetName());

Console.WriteLine("Size: " + tanker.GetSize());

Console.WriteLine("Max weight: " + tanker.GetWeightLimit() + " tons");

Console.WriteLine("Stuff: " + tanker.GetNumOfStaff() + " man");

}

}

}

**CargoShip.cs**

namespace lab5.\_1.\_9

{

public abstract class CargoShip

{

private bool isCreated = false, isSail = false;

private string Name = "";

private int Size = 1;

public CargoShip(string name, int size)

{

Name = name;

Size = size;

}

public void SetName(string name)

{

Name = name;

}

public string GetName()

{

return Name;

}

public void SetSize(int size)

{

Size = size;

}

public int GetSize()

{

return Size;

}

public bool IsCreate()

{

return isCreated;

}

public bool IsSail()

{

return isSail;

}

public void Create()

{

isCreated = true;

}

public void Destroy()

{

isCreated = false;

}

public void Sail()

{

isSail = true;

}

public void DontSail()

{

isSail = false;

}

}

}

**Ship.cs**

namespace lab5.\_1.\_9

{

public interface Ship

{

public void Create();

public void Destroy();

public void Sail();

public void DontSail();

}

}

**Tanker.cs**

namespace lab5.\_1.\_9

{

public class Tanker : CargoShip

{

int WeightLimit = 1;

int NumOfStaff = 1;

public Tanker(string name, int size) : base (name, size)

{

}

public Tanker(string name, int size, int weightLimit) : base(name, size)

{

WeightLimit = weightLimit;

}

public Tanker(string name, int size, int weightLimit, int numOfStaff) : base(name, size)

{

WeightLimit = weightLimit;

NumOfStaff = numOfStaff;

}

public void SetWeightLimit(int weightLimit)

{

WeightLimit = weightLimit;

}

public int GetWeightLimit()

{

return WeightLimit;

}

public void SetNumOfStaff(int numOfStaff)

{

NumOfStaff = numOfStaff;

}

public int GetNumOfStaff()

{

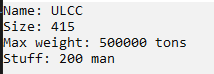
return NumOfStaff;

}

}

}

**Результат работы программы:**

****

****

**Program.cs**

using System;

using System.Collections.Generic;

namespace lab5.\_2.\_9

{

class Program

{

static void Main(string[] args)

{

List<Vehicle> list = new List<Vehicle>();

list.Add(new Car(120, 30, 3));

list.Add(new Carriage(60, 20, 2));

list.Add(new Bicycle(40, 10, 1));

Console.WriteLine("Вывод скорости и стоимости (для пассажиров и груза) транспортов:");

foreach(Vehicle lis in list)

{

lis.printTypeOfTransport();

lis.printSpeed();

lis.printPassengerTransportationCost();

lis.printCargoTransportationCost();

}

int distance = 1500;

Console.WriteLine("\nРассчет для каждого транспорта, расстояние = " + distance + "км");

foreach (Vehicle lis in list)

{

lis.calculateMovement(distance);

}

}

}

}

**Vehicle.cs**

using System;

namespace lab5.\_2.\_9

{

public abstract class Vehicle

{

private int Speed = 0, PassengerTransportationCost = 0, CargoTransportationCost = 0;

public Vehicle(int speed, int passengerTransportationCost, int сargoTransportationCost)

{

Speed = speed;

PassengerTransportationCost = passengerTransportationCost;

CargoTransportationCost = сargoTransportationCost;

}

public int GetSpeed()

{

return Speed;

}

public void SetSpeed(int speed)

{

Speed = speed;

}

public int GetPassengerTransportationCost()

{

return PassengerTransportationCost;

}

public void SetPassengerTransportationCost(int passengerTransportationCost)

{

PassengerTransportationCost = passengerTransportationCost;

}

public int GetCargoTransportationCost()

{

return CargoTransportationCost;

}

public void SetCargoTransportationCost(int cargoTransportationCost)

{

CargoTransportationCost = cargoTransportationCost;

}

public void printSpeed()

{

Console.WriteLine("\tСкорость = " + Speed + "км/ч");

}

public void printPassengerTransportationCost()

{

Console.WriteLine("\tСтоимость перевозки пассажира = " + PassengerTransportationCost + "$/km");

}

public void printCargoTransportationCost()

{

Console.WriteLine("\tСтоимость перевозки груза = " + CargoTransportationCost + "$/km");

}

public abstract void printTypeOfTransport();

public abstract void calculateMovement(int distance);

}

}

**Car.cs**

using System;

namespace lab5.\_2.\_9

{

public class Car : Vehicle

{

public Car(int speed, int passengerTransportationCost, int сargoTransportationCost) : base(speed, passengerTransportationCost, сargoTransportationCost)

{

}

public override void printTypeOfTransport()

{

Console.WriteLine("Машина: ");

}

public override void calculateMovement(int distance)

{

int hour = distance / GetSpeed();

Console.WriteLine("Поездка на машине займет " + hour + " час(ов) и будет стоить " + GetPassengerTransportationCost() \* hour + " рублей для пассажиров, и " + GetCargoTransportationCost() \* hour + " рублей для товара ");

}

}

}

**Bicycle.cs**

using System;

namespace lab5.\_2.\_9

{

class Bicycle : Vehicle

{

public Bicycle(int speed, int passengerTransportationCost, int сargoTransportationCost) : base(speed, passengerTransportationCost, сargoTransportationCost)

{

}

public override void printTypeOfTransport()

{

Console.WriteLine("Велосипед: ");

}

public override void calculateMovement(int distance)

{

int hour = distance / GetSpeed();

Console.WriteLine("Поездка на велосипеде займет " + hour + " час(ов) и будет стоить " + GetPassengerTransportationCost() \* hour + " рублей для пассажиров, и " + GetCargoTransportationCost() \* hour + " рублей для товара ");

}

}

}

**Carriage.cs**

using System;

namespace lab5.\_2.\_9

{

class Carriage : Vehicle

{

public Carriage(int speed, int passengerTransportationCost, int сargoTransportationCost) : base(speed, passengerTransportationCost, сargoTransportationCost)

{

}

public override void printTypeOfTransport()

{

Console.WriteLine("Повозка: ");

}

public override void calculateMovement(int distance)

{

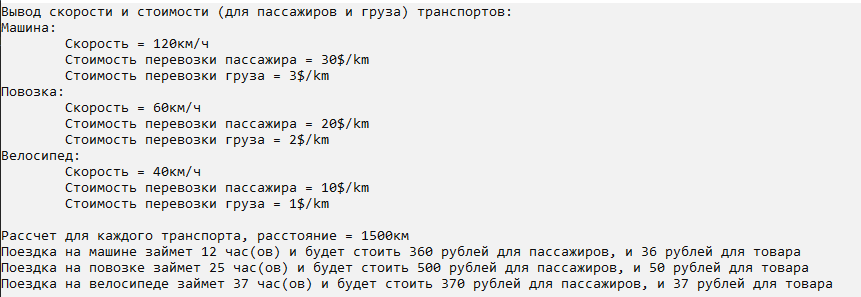
int hour = distance / GetSpeed();

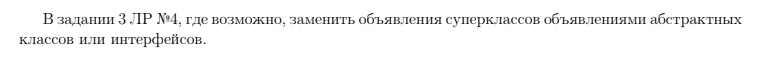
Console.WriteLine("Поездка на повозке займет " + hour + " час(ов) и будет стоить " + GetPassengerTransportationCost() \* hour + " рублей для пассажиров, и " + GetCargoTransportationCost() \* hour + " рублей для товара ");

}

}

}

****

****

**Program.cs**

using System;

using System.Collections.Generic;

namespace lab4.\_3.\_9

{

class Program

{

static List<string> RandomStatoins(int NumStations)

{

List<string> stations = new List<string>();

if (NumStations <= 0)

{

return stations;

}

string[] stationsName = { "Брест", "Минск", "Витебск", "Гомель", "Гродно", "Могилёв", "Бобруйск", "Барановичи", "Новополоцк", "Пинск", "Борисов", "Лида", "Мозырь", "Полоцк", "Слоним", "Орша", "Молодечно", "Жлобин", "Кобрин", "Слуцк" };

Random rand = new Random();

for(int i = 0; i < NumStations; i++)

{

stations.Add(stationsName[rand.Next(0, stationsName.Length)]);

}

return stations;

}

public static DateTime RandomDateTime(DateTime min, DateTime max)

{

Random \_ran = new Random();

return DateTime.MinValue.Add(TimeSpan.FromTicks(min.Ticks + (long)(\_ran.NextDouble() \* (max.Ticks - min.Ticks))));

}

static List<DateTime> GenerateTime(int NumStations)

{

List<DateTime> dateTimes = new List<DateTime>();

DateTime maxdate = new DateTime(2021, 12, 10, 0, 0, 0);

for(int i = 0; i < NumStations; i++)

{

dateTimes.Add(RandomDateTime(DateTime.Now, maxdate));

}

dateTimes.Sort();

return dateTimes;

}

static void Main(string[] args)

{

int NumStations = 10;

TrainStation trainStation = new TrainStation();

trainStation.AddTrain(new Train(RandomStatoins(NumStations), GenerateTime(NumStations), 10, 1));

trainStation.AddTrain(new Train(RandomStatoins(NumStations), GenerateTime(NumStations), 15, 2));

trainStation.AddTrain(new Train(RandomStatoins(NumStations), GenerateTime(NumStations), 12, 3));

trainStation.AddTrain(new Train(RandomStatoins(NumStations), GenerateTime(NumStations), 11, 4));

trainStation.AddTrain(new Train(RandomStatoins(NumStations), GenerateTime(NumStations), 13, 5));

RailwayTicketOffice railwayTicketOffice = new RailwayTicketOffice();

railwayTicketOffice.SetTrainStation(trainStation);

Administrator Administrator = new Administrator(38, "Kirill Smekalov", railwayTicketOffice);

Passenger Passenger = new Passenger(19, "Stas Sokolov", 100);

railwayTicketOffice.AddAdministrator(Administrator);

railwayTicketOffice.AddPassenger(Passenger);

if (Passenger.RequestOrderAndCheck(railwayTicketOffice, "Брест", "Минск", trainStation))

{

Console.WriteLine("Train successfully found");

}

else

{

Console.WriteLine("Train not found");

}

}

}

}

**Person.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public abstract class Person

{

private int Age = 0;

private string Name;

public Person(int age, string name)

{

Age = age;

Name = name;

}

public void SetAge(int age)

{

Age = age;

}

public int GetAge()

{

return Age;

}

public void SetName(string name)

{

Name = name;

}

public string GetName()

{

return Name;

}

}

}

**Passenger.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public class Passenger : Person

{

private List<Order> Orders = new List<Order>();

private int Bill = 0;

private Check Check = null;

public Passenger(int Age, string Name) : base (Age, Name)

{

}

public Passenger(int Age, string Name, int Bill) : base(Age, Name)

{

this.Bill = Bill;

}

public Passenger(int Age, string Name, int Bill, List<Order> Orders) : base(Age, Name)

{

this.Bill = Bill;

this.Orders = Orders;

}

public Passenger(int Age, string Name, int Bill, List<Order> Orders, Check Check) : base(Age, Name)

{

this.Bill = Bill;

this.Orders = Orders;

this.Check = Check;

}

public void SetOrders(List<Order> orders)

{

Orders = orders;

}

public List<Order> GetOrders()

{

return Orders;

}

public void SetCheck(Check check)

{

Check = check;

}

public Check GetCheck()

{

return Check;

}

public void SetBill(int bill)

{

Bill = bill;

}

public int GetBill()

{

return Bill;

}

public void AddOrder(Order order)

{

Orders.Add(order);

}

public void RemoveOrder(Order order)

{

Orders.Remove(order);

}

public bool AddCheck(Train train)

{

if(Bill - train.GetPrice() >= 0)

{

Bill -= train.GetPrice();

return true;

}

return false;

}

public bool RequestOrderAndCheck(RailwayTicketOffice railwayTicketOffice, string StartStation, string EndStation, TrainStation trainStation)

{

var temp = railwayTicketOffice.TrainSearch(StartStation, EndStation);

if (temp.Item1 != -1)

{

Order order = railwayTicketOffice.ProcessOrder(this, temp.Item2);

if (order == null)

{

return false;

}

AddOrder(order);

return AddCheck(temp.Item2);

}

return false;

}

}

}

**Administrator.cs**

using System;

using System.Collections.Generic;

namespace lab4.\_3.\_9

{

public class Administrator : Person

{

private RailwayTicketOffice railwayTicketOffice = new RailwayTicketOffice();

public Administrator(int age, string name, RailwayTicketOffice railwayTicketOffice) : base(age, name)

{

this.railwayTicketOffice = railwayTicketOffice;

}

public void SetRailwayTicketOffice(RailwayTicketOffice railwayTicketOffice)

{

this.railwayTicketOffice = railwayTicketOffice;

}

public RailwayTicketOffice GetRailwayTicketOffice()

{

return railwayTicketOffice;

}

public void AddOrder(Order order)

{

railwayTicketOffice.AddOrder(order);

}

}

}

**Train.cs**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

namespace lab4.\_3.\_9

{

public class Train

{

private List<string> Stations = new List<string>();

private List<DateTime> TrainTime = new List<DateTime>();

private int Price = 1;

private int ID = 0;

public Train(List<string> stations, List<DateTime> trainTime, int price, int id)

{

Stations = stations;

TrainTime = trainTime;

Price = price;

ID = id;

}

public Train(List<string> stations, List<DateTime> trainTime)

{

Stations = stations;

TrainTime = trainTime;

}

public void SetStations(List<string> stations)

{

Stations = stations;

}

public List<string> GetStations()

{

return Stations;

}

public void SetTrainTime(List<DateTime> trainTime)

{

TrainTime = trainTime;

}

public List<DateTime> GetTrainTime()

{

return TrainTime;

}

public void SetPrice(int price)

{

Price = price;

}

public int GetPrice()

{

return Price;

}

public void SetID(int id)

{

ID = id;

}

public int GetID()

{

return ID;

}

public bool \_Equals(object Other)

{

if (GetStations().Equals(((Train)Other).GetStations()) && GetTrainTime().Equals(((Train)Other).GetTrainTime()) && GetPrice().Equals(((Train)Other).GetPrice()) && GetID().Equals(((Train)Other).GetID()))

{

return true;

}

return false;

}

}

}

**TrainStation.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public class TrainStation

{

private List<Train> Trains = new List<Train>();

public TrainStation()

{

}

public TrainStation(List<Train> train)

{

Trains = train;

}

public void SetTrains(List<Train> train)

{

Trains = train;

}

public List<Train> GetTrains()

{

return Trains;

}

public void AddTrain(Train train)

{

Trains.Add(train);

}

public void RemoveTrain(Train train)

{

Trains.Remove(train);

}

public Tuple<int, Train> TrainSearch(string StartStation, string EndStation)

{

List<string> temp;

for (int i = 0; i < Trains.Count; i++)

{

temp = Trains[i].GetStations();

if (temp.IndexOf(StartStation) < temp.IndexOf(EndStation) && temp.IndexOf(StartStation) != -1)

{

return Tuple.Create(i, Trains[i]);

}

}

Train train = null;

return Tuple.Create(-1, train);

}

public Train GiveTrain(Train train)

{

int index = -1;

for (int i = 0; i < Trains.Count; i++)

{

if (Trains[i].GetStations() == train.GetStations() && Trains[i].GetTrainTime() == train.GetTrainTime() && Trains[i].GetPrice() == train.GetPrice() && Trains[i].GetID() == train.GetID())

{

index = i;

}

}

if (index == -1)

{

return null;

}

Train result = Trains[index];

RemoveTrain(train);

return result;

}

}

}

**RailwayTicketOffice.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public class RailwayTicketOffice

{

private TrainStation TrainStations = new TrainStation();

private List<Train> Trains = new List<Train>();

private List<Administrator> Administrators = new List<Administrator>();

private List<Passenger> Passengers = new List<Passenger>();

private List<Order> Orders = new List<Order>();

public RailwayTicketOffice()

{

}

public RailwayTicketOffice(TrainStation trainStations)

{

TrainStations = trainStations;

}

public RailwayTicketOffice(TrainStation trainStations, List<Train> trains)

{

TrainStations = trainStations;

Trains = trains;

}

public RailwayTicketOffice(TrainStation trainStations, List<Train> trains, List<Administrator> administrators)

{

TrainStations = trainStations;

Trains = trains;

Administrators = administrators;

}

public RailwayTicketOffice(TrainStation trainStations, List<Train> trains, List<Administrator> administrators, List<Passenger> passengers)

{

TrainStations = trainStations;

Trains = trains;

Administrators = administrators;

Passengers = passengers;

}

public RailwayTicketOffice(TrainStation trainStations, List<Train> trains, List<Administrator> administrators, List<Passenger> passengers, List<Order> orders)

{

TrainStations = trainStations;

Trains = trains;

Administrators = administrators;

Passengers = passengers;

Orders = orders;

}

public void SetTrainStation(TrainStation trainStations)

{

TrainStations = trainStations;

}

public TrainStation GetCatalog()

{

return TrainStations;

}

public void SetTrain(List<Train> train)

{

Trains = train;

}

public List<Train> GetTrain()

{

return Trains;

}

public void SetAdministrator(List<Administrator> administrator)

{

Administrators = administrator;

}

public List<Administrator> GetAdministrator()

{

return Administrators;

}

public void SetPassengers(List<Passenger> passenger)

{

Passengers = passenger;

}

public List<Passenger> GetPassenger()

{

return Passengers;

}

public void SetOrder(List<Order> order)

{

Orders = order;

}

public List<Order> GetOrder()

{

return Orders;

}

public void AddTrain(Train train)

{

TrainStations.AddTrain(train);

}

public void RemoveTrain(Train train)

{

TrainStations.RemoveTrain(train);

}

public void AddAdministrator(Administrator administrator)

{

Administrators.Add(administrator);

}

public void RemoveAdministrator(Administrator administrator)

{

Administrators.Remove(administrator);

}

public void AddPassenger(Passenger passenger)

{

Passengers.Add(passenger);

}

public void RemovePassenger(Passenger passenger)

{

Passengers.Remove(passenger);

}

public void AddOrder(Order order)

{

Orders.Add(order);

}

public void RemoveOrder(Order order)

{

Orders.Remove(order);

}

public Tuple<int, Train> TrainSearch(string StartStation, string EndStation)

{

return TrainStations.TrainSearch(StartStation, EndStation);

}

public Order ProcessOrder(Passenger passenger, Train train)

{

Train orderedTrain = TrainStations.GiveTrain(train);

if (orderedTrain == null)

{

return null;

}

Order order = new Order(orderedTrain, passenger);

bool processed = false;

while (!processed)

{

foreach (Administrator administrator in Administrators)

{

administrator.AddOrder(order);

processed = true;

break;

}

}

Orders.Add(order);

return order;

}

}

}

**Order.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public class Order

{

private Train Train = null;

private Passenger Passenger = null;

public Order(Train train, Passenger passenger)

{

Train = train;

Passenger = passenger;

}

public void SetTrain(Train train)

{

Train = train;

}

public Train GetTrain()

{

return Train;

}

public void SetPassenger(Passenger passenger)

{

Passenger = passenger;

}

public Passenger GetPassenger()

{

return Passenger;

}

}

}

**Check.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab4.\_3.\_9

{

public class Check

{

Check()

{

}

}

}

**Результат работы программы:**

****