**Выполнил:** Александр Абзаев, группа ИСТБп-18-1

# Лабораторное задание №3

### Наследование

### Постановка задачи

**Задача №3.** Для раздачи объектов космоса (удалённость от земли)

* Планеты (радиус, наличие атмосферы, сила притяжения)
* Звезды (плотность, цвет, температура)
* Кометы (период прохождения через солнечную систему, название)

### Описание методов

**Класс Spasee**

Метод GetInfo используется для описания космических объектов, возвращает строку, содержащую “Удаленность от земли (км)”.

**Класс Planet**

Метод GetInfo используется для описания планеты, возвращает строку, содержащую вид космического объекта "Планета" и значение характеристик: Удаленность от земли (Используется стандартный метод GetInfo из Spasee), Радиус планеты (км), Сила притяжения (g) и Наличие атмосферы.

Метод Generate возвращает объект, содержащий параметры: Удаленность от земли, Радиус планеты, Сила притяжения и Наличие атмосферы, у которых заданы случайные значение.

**Класс Star**

Метод GetInfo используется для описания звезды, возвращает строку, содержащую вид космического объекта "Звезда" и значение характеристик: Удаленность от земли (Используется стандартный метод GetInfo из Spasee), Плотность (мг/м^3), Цвет звезды и Температура (K).

Метод Generate возвращает объект, содержащий параметры: Удаленность от земли, Плотность, Цвет звезды и Температура, у которых заданы случайные значение.

**Класс Comet**

Метод GetInfo используется для описания кометы, возвращает строку, содержащую вид космического объекта "Комета" и значение характеристик: Удаленность от земли (Используется стандартный метод GetInfo из Spasee), Период прохождения через солнечную систему (лет) и Название кометы.

Метод Generate возвращает объект, содержащий параметры: Удаленность от земли, Период прохождения через солнечную систему и Название кометы, у которых заданы случайные значение.

### Код программы

|  |
| --- |
| Space.cs  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace WindowsFormsApp12  {  public class Spacee  {  public static Random rnd = new Random();  public int RemotFromTheEarth = 0; // удаленность от земли  public virtual String GetInfo()  {  var str = String.Format("\nУдаленность от земли (км): {0}", this.RemotFromTheEarth);  return str;  }  }  // планеты  public class Planet : Spacee  {  public int Radius = 0; // радиус планеты  public int ForceOfGravity = 0; // сила притяжения  public bool Atmosphere = false; //наличие атмосферы  // переопределили метод  public override String GetInfo()  {  string planAtmosphere = "";  switch (this.Atmosphere)  {  case false:  planAtmosphere = "атмосфера отсутствует";  break;  case true:  planAtmosphere = "атмосфера присутствует";  break;  }  var str = "Планета";  str += base.GetInfo();  str += String.Format("\nРадиус планеты (км): {0}", this.Radius);  str += String.Format("\nСила притяжения (g): {0}", this.ForceOfGravity);  str += String.Format("\nНаличие атмосферы: {0}", planAtmosphere);  return str;  }  // добавили статический метод генерации случайных дпнных планеты  public static Planet Generate()  {  return new Planet  {  RemotFromTheEarth = rnd.Next() % 1000000,  Radius = rnd.Next() % 30000,  ForceOfGravity = 1 + rnd.Next() % 8,  Atmosphere = rnd.Next() % 2 == 0  };  }  }  // цвета звезд  public enum StarsColor { blue, yellow, red };  // звезды  public class Star : Spacee  {  public int Density = 0; // плотность  public StarsColor type = StarsColor.blue; // цвет звезды  public int Temperature = 0; // температура (K)  // переопределили метод  public override String GetInfo()  {  string colorStars = "";  switch (this.type)  {  case StarsColor.blue:  colorStars = "синий";  break;  case StarsColor.yellow:  colorStars = "желтый";  break;  case StarsColor.red:  colorStars = "красный";  break;  }  var str = "Звезда";  str += base.GetInfo();  str += String.Format("\nПлотность (мг/м^3): {0}", this.Density);  str += String.Format("\nЦвет звезды: {0}", colorStars);  str += String.Format("\nТемпература (K): {0}", this.Temperature);  return str;  }  // добавили статический метод генерации случайных дпнных звезды  public static Star Generate()  {  return new Star  {  RemotFromTheEarth = rnd.Next() % 1000000,  Density = 1 + rnd.Next() % 20,  type = (StarsColor)rnd.Next(3),  Temperature = 2000 + rnd.Next() % 60000  };  }  }  // названия комет  public enum CometName { Tempely, Hallea, Donati };  // комета  public class Comet : Spacee  {  public int PassageThroughTheSolarSystem = 0; // период прохождения через солнечную систему  public CometName type = CometName.Donati; // название кометы  // переопределили метод  public override String GetInfo()  {  string nameComet = "";  switch (this.type)  {  case CometName.Tempely:  nameComet = "Темпели";  break;  case CometName.Hallea:  nameComet = "Галлея";  break;  case CometName.Donati:  nameComet = "Донати";  break;  }  var str = "Комета";  str += base.GetInfo();  str += String.Format("\nПериод прохождения через солнечную систему (лет): {0}", this.PassageThroughTheSolarSystem);  str += String.Format("\nНазвание кометы: {0}", nameComet);  return str;  }  // добавили статический метод генерации случайных дпнных комет  public static Comet Generate()  {  return new Comet  {  RemotFromTheEarth = rnd.Next() % 1000000,  PassageThroughTheSolarSystem = 3 + rnd.Next() % 300,  type = (CometName)rnd.Next(3)  };  }  }  }  Form1.cs  using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace WindowsFormsApp12  {  public partial class Form1 : Form  {  // поле под список (планет, звезд, комет)  List<Spacee> spaceList = new List<Spacee>();  public Form1()  {  InitializeComponent();  ShowInfo();  }  private void btnRefill\_Click(object sender, EventArgs e)  {  this.spaceList.Clear();  var rnd = new Random();  for (var i = 0; i < 10; ++i)  {  switch (rnd.Next() % 3)  {  case 0: // если 0, то планета  this.spaceList.Add(Planet.Generate());  break;  case 1: // если 1 то звезда  this.spaceList.Add(Star.Generate());  break;  case 2: // если 2 то комета  this.spaceList.Add(Comet.Generate());  break;  }  }  wtInfo();  ShowInfo();  }  // функция выводит информацию о количестве косм объектов на форму  private void ShowInfo()  {  // заведем счетчики под каждый тип  int planetCount = 0;  int starCount = 0;  int cometCount = 0;  // пройдемся по всему списку  foreach (var spase in this.spaceList)  {  if (spase is Planet)  {  planetCount += 1;  }  else if (spase is Star)  {  starCount += 1;  }  else if (spase is Comet)  {  cometCount += 1;  }  }  // выводим на форму  txtInfo.Text = "Планет\tЗвезд\tКомет";  txtInfo.Text += "\n";  txtInfo.Text += String.Format("{0}\t{1}\t{2}", planetCount, starCount, cometCount);  }  private void wtInfo()  {  if (this.spaceList.Count == 0)  {  labInfo.Text = "ПУСТО!!!!!";  return;  }  var space = this.spaceList[0]; // берем объект  if (space is Planet)  {  labInfo.Text = "Объект в очереди: Планета";  }  else if (space is Star)  {  labInfo.Text = "Объект в очереди: Звезда";  }  else if (space is Comet)  {  labInfo.Text = "Объект в очереди: Комета";  }  else if (this.spaceList.Count == 0)  {  labInfo.Text = "ПУСТО!!!!!";  return;  }  }  private void btnGet\_Click(object sender, EventArgs e)  {  // если список пуст, то напишем что пусто и выйдем из функции  if (this.spaceList.Count == 0)  {  txtOut.Text = "ПУСТО!!!!!";  return;  }  var space = this.spaceList[0]; // берем первый объект  this.spaceList.RemoveAt(0); // удаляем  // вывод информации о объекте на поле  txtOut.Text = space.GetInfo();  wtInfo();  ShowInfo();  }  }  } |

### UML

