Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт/Факультет – Информационные системы и технологии   
 наименование

**Название работы** – Наследование

Отчет по лабораторной работе № 4

по дисциплине Технология программирования   
 наименование учебной дисциплины

Выполнилcтудент ИСТб-20-3 Д.А.Пинегин

номер группы подпись И. О. Фамилия

дата

Принял Доцент З.А. Бахвалова

должность подпись И.О. Фамилия

дата

Иркутск – 2021

# Лабораторная работа №4

### Постановка задачи

1. Реализовать новый объект, который будет исчезать при пересечении с игроком и появляться на новом месте
2. Реализовать вывод очков. Увеличивать количество очков при пересечении с объектом, добавленным в предыдущем пункте. Дополнительно добавить на поле несколько зеленых кругов.
3. Добавить счетчик к зеленому объекту. Если игрок не успел добраться до объекта, то переместить его. Сам счетчик хранить в зеленом кружке, событие конца отсчета должен генерировать зеленый кружок

**Словесное описание алгоритма**

1. Подключаем библиотеку Windows Forms.
2. Создаём форму и присваиваем кнопкам их функции.
3. Создаем свой класс, который будет преобразовывать данные.
4. После нажатия на кнопку перезаполнить, форма берёт рандомные значения и класс их преобразовывает в данные и выводит данные на панель пользователю.
5. После нажатия на кнопку взять элементы идут по цепочки удаляясь последовательно из очереди**.**

### Таблица спецификаций

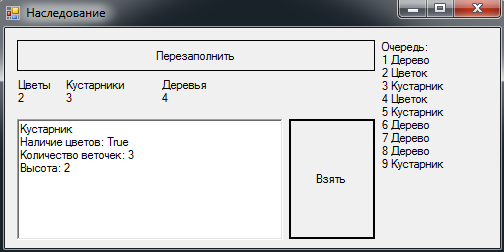
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | | Назначение | Тип | Модификатор доступа |
| **Методы класса Form1** | | | | |
| button1\_Click(object sender, EventArgs e) | | Создает экземпляры класса Plants | void | public |
| Queue() | | Вычисляет и выводит очередь объектов на вывод | void | public |
| ShowInfo() | | Показывает количество цветов, кустарников и деревьев | void | public |
| button2\_Click(object sender, EventArgs e) | | Передает “Покупателю” первый в очереди объект и удаляет его из списка | void | public |
| rnd | | Рандомное число | static Random | public |
| plantsList | | Список | List<Plants> | public |
| **Методы и поля класса Plants** | | | | |
| rnd | Рандомное число | | static Random | public |
| height | Высота растения | | Public int | public |
| GetInfo() | Получает значения высоты конкретного растения | | public virtual String | public |
| Перечисляемые типы файла Plants.cs | | | | |
| Color | Цвет цветка | | public enum | public |
| Type | Тип цветка | | public enum | public |
| TypeTree | Тип дерева | | public enum | public |
| **Методы и поля класса Flower** | | | | |
| number | Количество лепестков | | public int | public |
| color | Цвет | | public Color | public |
| GetInfo() | Передает значения всех полей класса Flower в строку | | public override String | public |
| Generate() | Генерирует значения всех полей класса Flower | | public static Flower | public |
| **Методы и поля класса Shrub** | | | | |
| isFlower | Указание на наличие цветов | | public bool | public |
| number | Количество веточек | | public int | public |
| GetInfo() | Передает значения всех полей класса Flower в строку | | public override String | public |
| Generate() | Генерирует значения всех полей класса Flower | | public static Shrub | public |
| **Методы и поля класса Tree** | | | | |
| typetree | Тип дерева | | public TypeTree | public |
| Rad | Радиус дерева | | public int | public |
| GetInfo() | Передает значения всех полей класса Flower в строку | | public override String | public |
| Generate() | Генерирует значения всех полей класса Flower | | public static Tree | public |

|  |  |  |  |
| --- | --- | --- | --- |
| **Переменные метода** button1\_Click | | | |
| Имя | Назначение | тип | Диапазон |
| i | Счетчик | int | от -2147483648 до 2147483647 |
| **Переменные метода** Queue | | | |
| i | счетчик | int | от -2147483648 до 2147483647 |
| plant | Получает тип растения | var | - |
| str | Строка для записи | string | 16 символов |
| **Переменные метода** ShowInfo | | | |
| floawerCount | Количество цветов | int | от -2147483648 до 2147483647 |
| shrubCount | Количество кустарников | int | от -2147483648 до 2147483647 |
| treesCount | Количество деревьев | int | от -2147483648 до 2147483647 |
| a | Получает объект из List | var | - |
| **Переменные метода** button2\_Click | | | |
| plant | Получает объект (растение) | var | - |
| **Переменные метода** GetInfo() | | | |
| str | Строка для записи | string | 16 символов |

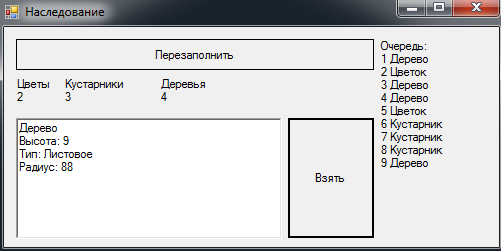
### Таблица тестов

| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Проверка кнопки перезаполнить | - | Очередь:  1 Кустарник  2 Дерево  3 Дерево  4 Кустарник  5 Цветок  6 Кустарник  7 Дерево  8 Цветок  9 Кустарник  10 Кустарник |
| 2 | Проверка кнопки взять | 1 Кустарник  2 Дерево  3 Дерево  4 Кустарник  5 Цветок  6 Кустарник  7 Дерево  8 Цветок  9 Кустарник  10 Кустарник | Кустарник  Наличие цветов: True  Количество веточек: 15  Высота: 6  Дерево  Высота: 24  Тип: Листовое  Радиус: 102  Дерево  Высота: 53  Тип: Хвойное  Радиус: 41  Кустарник  Наличие цветов: False  Количество веточек: 11  Высота: 5  Цветок  Количество липестков: 16  Цвет: желтый  Тип: Тюльпан  Высота: 4  Кустарник  Наличие цветов: True  Количество веточек: 15  Высота: 9  Дерево  Высота: 84  Тип: Листовое  Радиус: 77  Цветок  Количество липестков: 11  Цвет: желтый  Тип: Хризантема  Высота: 3  Кустарник  Наличие цветов: False  Количество веточек: 4  Высота: 1  Кустарник  Наличие цветов: False  Количество веточек: 13  Высота: 4 |
| 3 | Тест чтобы убедиться в правильности данной программы | Очередь:  1 Цветок  2 Кустарник  3 Дерево  4 Цветок  5 Кустарник  6 Дерево  7 Дерево  8 Кустарник  9 Дерево  10 Цветок | Кустарник  Наличие цветов: False  Количество веточек: 13  Высота: 4  Цветок  Количество липестков: 5  Цвет: красный  Тип: Тюльпан  Высота: 5 |

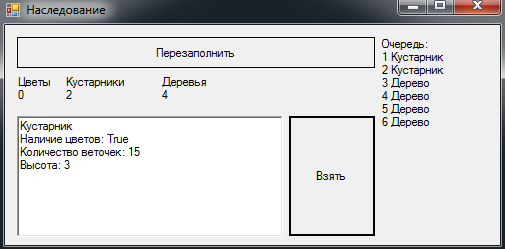
### Результат работы программы



Тест №1.



Тест №2.



Тест №3.

### Код программы

**Класс 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 WindowsFormsApp2

{

public partial class Form1 : Form

{

public static Random rnd = new Random();

List<Plants> plantsList = new List<Plants>();

public Form1()

{

InitializeComponent();

ShowInfo();

}

private void button1\_Click(object sender, EventArgs e)

{

this.plantsList.Clear();

for (var i = 0; i < 10; ++i)

{

switch (rnd.Next() % 3)

{

case 0:

this.plantsList.Add(Flower.Generate());

break;

case 1:

this.plantsList.Add(Shrub.Generate());

break;

case 2:

this.plantsList.Add(Tree.Generate());

break;

}

}

ShowInfo();

Queue();

}

private void Queue()

{

String str = "Очередь:\n";

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

{

var plant = this.plantsList[i];

if (plant is Flower) str += $"{i + 1} " + "Цветок\n";

if (plant is Shrub) str += $"{i + 1} " + "Кустарник\n";

if (plant is Tree) str += $"{i + 1} " + "Дерево\n";

}

richTextBox3.Text = str;

}

private void ShowInfo()

{

int floawerCount = 0;

int shrubCount = 0;

int treesCount = 0;

foreach (var a in this.plantsList)

{

if (a is Flower)

{

floawerCount++;

}

if (a is Shrub)

{

shrubCount++;

}

if (a is Tree)

{

treesCount++;

}

}

richTextBox2.Text = "Цветы\tКустарники\tДеревья";

richTextBox2.Text += "\n";

richTextBox2.Text += String.Format("{0}\t{1}\t\t{2}", floawerCount, shrubCount, treesCount);

}

private void button2\_Click(object sender, EventArgs e)

{

if (this.plantsList.Count == 0)

{

richTextBox4.Text = "";

return;

}

var plant = this.plantsList[0];

this.plantsList.RemoveAt(0);

richTextBox4.Text = plant.GetInfo();

ShowInfo();

Queue();

}

}

}

### Класс Plants.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WindowsFormsApp2

{

public class Plants

{

public static Random rnd = new Random();

public int height = 0;

public virtual String GetInfo()

{

var str = String.Format("\nВысота: {0}", this.height);

return str;

}

}

public enum Color { Белый, Красный, Желтый, Синий };

public enum Type { Роза, Хризантема, Тюльпан };

public class Flower : Plants

{

public int number = 0;

public Color color = Color.Синий;

public Type type = Type.Роза;

public override String GetInfo()

{

var str = "Цветок";

str += String.Format("\nКоличество липестков: {0}\nЦвет: {1}\nТип: {2}", this.number, this.color, this.type);

str += base.GetInfo();

return str;

}

public static Flower Generate()

{

return new Flower

{

number = rnd.Next() % 21,

color = (Color)rnd.Next(4),

type = (Type)rnd.Next(3),

height = rnd.Next() % 5 + 1

};

}

}

public class Shrub : Plants

{

public bool isFlower = false;

public int number = 0;

public override String GetInfo()

{

var str = "Кустарник";

str += String.Format("\nНаличие цветов: {0}\nКоличество веточек: {1}", this.isFlower, this.number);

str += base.GetInfo();

return str;

}

public static Shrub Generate()

{

return new Shrub

{

number = rnd.Next() % 21,

isFlower = rnd.Next() % 2 == 0,

height = rnd.Next() % 10 + 1

};

}

}

public enum TypeTree { Хвойное, Листовое };

public class Tree : Plants

{

public TypeTree typetree = TypeTree.Листовое;

public int Rad = 0;

public override String GetInfo()

{

var str = "Дерево";

str += base.GetInfo();

str += String.Format("\nТип: {0}\nРадиус: {1}", this.typetree, this.Rad);

return str;

}

public static Tree Generate()

{

return new Tree

{

height = rnd.Next() % 100 + 1,

typetree = (TypeTree)rnd.Next(2),

Rad = rnd.Next() % 100 + 20

};

}

}

}

**Класс Programm.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp2

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}