Лабораторная работа 5 по С#

Цель работы. Освоить механизм перегрузки операций: математических, сравнения, логических, преобразования типов, а также механизм индексирования классов. Продемонстрировать умение создавать классы с расширенной функциональностью, создавать интуитивно понятный и легко читаемый код. Научиться создавать тестовые методы для проверки корректности работы перегруженных операций.

Задание 1

namespace Лабораторная\_работа\_5\_по\_C\_

{

internal class Complex

{

private double a;

private double b;

//свойства с перепроверкой

public double A { get => a; set => a = value; }

public double B { get => b; set => b = value; }

//конструктор

public Complex(double a, double b)

{

A = a;

B = b;

}

//переопределение

public override string? ToString()

{

return $"A={A} B={B}";

}

//индексатор

public double this[int i]

{

get { if(i==0) return A;

if(i==1) return B;

throw new IndexOutOfRangeException("Индекс должен быть 0 или 1");

}

set

{

if (i == 0) A=value;

if (i == 1) B=value;

throw new IndexOutOfRangeException("Индекс должен быть 0 или 1");

}

}

//Перегрузки операций

public static Complex operator + (Complex a, Complex b)

{

return new Complex(a.A+b.A,a.B+b.B);

}

public static Complex operator -(Complex a, Complex b)

{

return new Complex(a.A - b.A, a.B - b.B);

}

public static Complex operator \*(Complex a, Complex b)

{

return new Complex(a.A \* b.A-a.B\*b.B, a.A\*b.B + a.B\*b.A);

}

public static Complex operator /(Complex a, Complex b)

{

var denominator=b.A\*b.A+b.B\*b.B;

if(denominator!=0)

{

return new Complex((a.A \* b.A + a.B \* b.B)/denominator, (a.B \* b.A - a.A \* b.B)/denominator);

}

else

{

throw new DivideByZeroException("делить на 0 нельзя");

}

}

//перегрузка для операций с double

public static Complex operator +(Complex a, double b)

{

return new Complex(a.A+b,a.B);

}

public static Complex operator -(Complex a, double b)

{

return new Complex(a.A - b, a.B);

}

public static Complex operator \*(Complex a, double b)

{

return new Complex(a.A \* b, a.B\*b);

}

public static Complex operator /(Complex a, double b)

{

if (b == 0) throw new DivideByZeroException("делить на 0 нельзя");

return new Complex(a.A / b, a.B / b);

}

private double Modul()

{

return Math.Sqrt(A\*A+B\*B);

}

//Сравнения

public static bool operator ==(Complex a, Complex b)

{

return a.Modul()==b.Modul();

}

public static bool operator !=(Complex a, Complex b)

{

return a.Modul() != b.Modul();

}

public static bool operator >(Complex a, Complex b)

{

return a.Modul() > b.Modul();

}

public static bool operator <(Complex a, Complex b)

{

return a.Modul() < b.Modul();

}

//неявное преобразование

public static implicit operator bool(Complex a)

{

return a.A!=0 || a.B!=0;

}

//явное преобразование

public static explicit operator double(Complex a)

{

return (double)a.Modul();

}

}

}

using Лабораторная\_работа\_5\_по\_C\_;

Console.WriteLine("Лабораторная работа 5 по С#");

Complex c1 = new Complex(5, 8);

Complex c2 = new Complex(4, -7);

Complex c3 = new Complex(0, 0);

// Демонстрация использования свойств

Console.WriteLine("C1: " + c1);

Console.WriteLine("C2: " + c2);

Console.WriteLine("C3: " + c3);

// Демонстрация работы операций

Console.WriteLine("C1 + C2: " + (c1 + c2));

Console.WriteLine("C1 - C2: " + (c1 - c2));

Console.WriteLine("C1 \* C2: " + (c1 \* c2));

Console.WriteLine("C1 / C2: " + (c1 / c2));

// Работает с double

Console.WriteLine("C1 + 2: " + (c1 + 2));

Console.WriteLine("C2 - 1: " + (c2 - 1));

// Сравнения

Console.WriteLine("C1 == C2: " + (c1 == c2));

Console.WriteLine("C1 != C2: " + (c1 != c2));

Console.WriteLine("C1 < C2: " + (c1 < c2));

Console.WriteLine("C1 > C2: " + (c1 > c2));

// Переход к bool

Console.WriteLine("C3 (false): " + ((bool)c3));

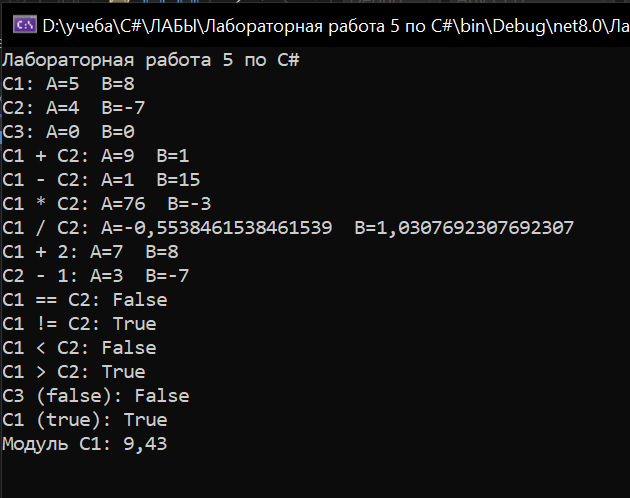
Console.WriteLine("C1 (true): " + ((bool)c1));

// Преобразование в double

double modulC1 = (double)c1;

Console.WriteLine($"Модуль C1: {modulC1:0.00}");

Console.ReadKey();



Задание 2

Вариант 1

using LAB2\_задание\_2;

Account account=new Account("XXL", "13.01.2024", 1000);

Console.WriteLine(account);

account.Put(500);

account = account + 500;

Console.WriteLine(account);

account = account -1500;

Console.WriteLine(account);

Account account1 = new Account("M", "21.03.2023", 500);

Console.WriteLine(account1);

Console.WriteLine();

if (account == account1) Console.WriteLine("Суммы на счетах равны");

else Console.WriteLine("Суммы на счетах не равны");

Console.ReadKey();

namespace LAB2\_задание\_2

{

internal class Account

{

private string name;

private string dateOpen;

private double sum;

public Account(string name, string dateOpen, double sum)

{

Name = name;

DateOpen = dateOpen;

Sum = sum;

}

public string Name { get => name; set => name = value; }

public string DateOpen { get => dateOpen; set => dateOpen = value; }

public double Sum

{

get => sum;

set

{

if(value >= 0)sum = value;

}

}

public static Account operator +(Account a, double b)

{

Console.WriteLine($"Поступило на счет - {b}");

return new Account(a.name,a.dateOpen,a.Sum + b);

}

public static Account operator -(Account a, double b)

{

if (a.Sum >= b)

{

Console.WriteLine($"Снято со счета - {b}");

return new Account(a.name, a.dateOpen, a.Sum - b);

}

else

{

Console.WriteLine("Недостаточно денег на счете");

return a;

}

}

public static bool operator ==(Account a, Account b)

{

return a.Sum == b.Sum;

}

public static bool operator !=(Account a, Account b)

{

return a.Sum != b.Sum;

}

public void Put(double b)

{

Sum=sum + b;

Console.WriteLine($"На счет поступило {b}");

Console.WriteLine(this);

return ;

}

public override string? ToString()

{

string str = "Банковский счет ";

if(Name != null && DateOpen != null)

{

str += $"{Name}, открытый {DateOpen}, с суммой на счете {Sum}";

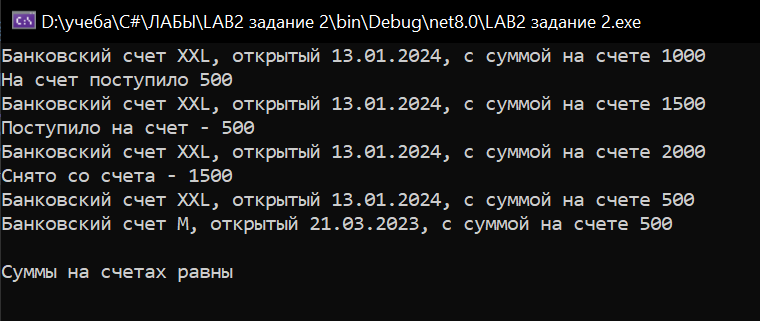
}

return str;

}

}

}



Вариант 2

Group group=new Group(123);

Console.WriteLine(group);

Console.WriteLine(group[3]);

group = group - "Захаров";

Console.WriteLine(group);

group = group + "Петров";

Console.WriteLine(group);

Console.ReadKey();

namespace LAB2\_задание\_2

{

internal class Group

{

private int nomerGroup;

public string[] Student= {"Ермолова","Захаров","Панченок","Лукьянов","Минова" };

public Group(int nomerGroup)

{

NomerGroup = nomerGroup;

}

public int NomerGroup

{

get => nomerGroup;

set

{

if (value > 0)nomerGroup = value;

}

}

public string this[int i]

{

get

{

return Student[i-1];

}

}

public static Group operator -(Group group , string value)

{

bool flag = false;

foreach (string s in group.Student)

{

if (s == value) { flag = true; break; }

}

if (flag)

{

string[] Students = new string[group.Student.Length - 1];

int j = 0;

for (int i = 0; i < group.Student.Length; i++)

{

if (group.Student[i] == value) { continue; j--; }

else

{

if (j <= group.Student.Length - 1)

{

Students[j] = group.Student[i];

}

}

j++;

}

group.Student = Students;

return group;

}

else

{

return group;

}

}

public static Group operator +(Group group , string value)

{

if (value != null)

{

string[] Students = new string[group.Student.Length + 1];

int j = 0;

for (int i = 0; i < group.Student.Length; i++)

{

Students[j] = group.Student[i];

j++;

}

Students[(int)group.Student.Length] = value;

group.Student = Students;

return group;

}

else

{

return group;

}

}

public override string? ToString()

{

string? str = $"Группа {NomerGroup}. Количество учащихся - {Student.Length}\n";

for (int i = 0;i<Student.Length;i++)

{

str += $"\t{i+1} - {Student[i].ToString()}\n";

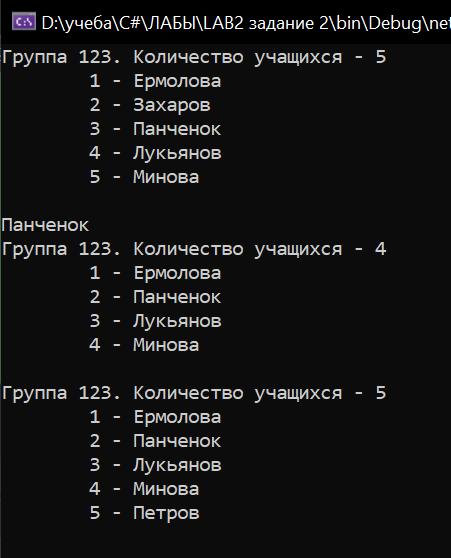
}

return str;

}

}

}



Вариант 3

PriceList priceList=new PriceList("13.12.2024");

Console.WriteLine(priceList);

priceList = priceList + new string[]{"5","Помидор","356" };

Console.WriteLine(priceList);

priceList = priceList - "3";

Console.WriteLine(priceList);

priceList = priceList - "Вода";

Console.WriteLine(priceList);

Console.WriteLine(priceList["1"]);

Console.ReadKey();

namespace LAB2\_задание\_2

{

internal class PriceList

{

private string dateCreate;

public string[] Products = {"1","Персик","134","2","Мишка","543","3","Рис","321","4","Вода","256" };

public PriceList(string dateCreate)

{

DateCreate = dateCreate;

}

public string DateCreate { get => dateCreate; set => dateCreate = value; }

public static PriceList operator+(PriceList priceList1, string[] strings)

{

string[] strings1 = new string[priceList1.Products.Length+3];

int j = 0;

for (int i = 0; i<priceList1.Products.Length; i++)

{

strings1[j++] = priceList1.Products[i];

}

j = 0;

for(int i = priceList1.Products.Length;i<strings1.Length; i++)

{

strings1[i] = strings[j++];

}

priceList1.Products = strings1;

return priceList1 ;

}

public static PriceList operator-(PriceList priceList1, string s)

{

bool flag = false;

for(int i = 0;i<priceList1.Products.Length;i++)

{

if (priceList1.Products[i]==s)flag = true;

}

if(flag)

{

int k=0;

bool flag1 = false;

for(int i = 0; i<priceList1.Products.Length; i++)

{

if (priceList1.Products[i] == s) { k = i; flag1 = true; }

}

if (k % 3 == 0 & flag1==true)

{

string[] strings=new string[priceList1.Products.Length-3];

int h = 0;

for(int i = 0;i<priceList1.Products.Length;i++ )

{

if(i==k || i==k+1 || i==k+2)

{

continue;

}

else

{

strings[h] = priceList1.Products[i];

h++;

}

}

priceList1.Products = strings;

return priceList1 ;

}

else

{

string[] strings = new string[priceList1.Products.Length - 3];

int h = 0;

for (int j = 0; j < priceList1.Products.Length; j++)

{

if (j == k || j == k + 1 || j == k - 1)

{

continue;

}

else

{

strings[h] = priceList1.Products[j];

h++;

}

}

priceList1.Products = strings;

return priceList1;

return priceList1;

}

}

else

{

return priceList1;

}

}

public string this[string s]

{

get

{

string str="";

for(int i = 0;i<Products.Length;i++ )

{

if (Products[i] == s)

str = $"{Products[i]} - {Products[i+1]} - {Products[i+2]} руб";

}

return str;

}

}

public override string? ToString()

{

string str = "Прейскурант ";

if (DateCreate != null)

{

str += $"от {DateCreate}. Количество товаров - {Products.Length / 3}\n";

int i = 0;

while(Products.Length > i)

{

str += $"\t{Products[i]} - ";

i++;

str += $"{Products[i]} - ";

i++;

str += $"{Products[i]} руб\n";

i++;

}

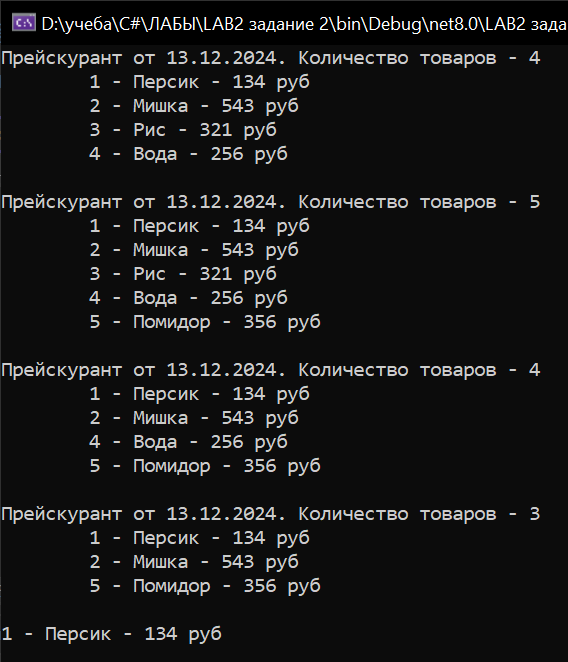
}

return str;

}

}

}



Задание 4

MailingList mailingList= new MailingList("13.01.2024");

Console.WriteLine(mailingList);

mailingList = mailingList + new string[] { "Fedya", "fedya65345@mail.ru" };

Console.WriteLine(mailingList);

mailingList = mailingList - "Fedya";

Console.WriteLine(mailingList);

mailingList = mailingList - "ken@mail.ru";

Console.WriteLine(mailingList);

Console.WriteLine(mailingList["Ben"]);

Console.WriteLine(mailingList["Lena"]);

Console.ReadKey();

namespace LAB2\_задание\_2

{

internal class MailingList

{

private string dateCreate;

public string[] Emails = {"Ken","ken@mail.ru","Gena","gena123@maail.ru","Lena","len123@mail.ru" };

public MailingList(string dateCreate)

{

DateCreate = dateCreate;

}

public string DateCreate { get => dateCreate; set => dateCreate = value; }

public static MailingList operator +(MailingList m,string[] strings)

{

string[] Emails1=new string[m.Emails.Length+strings.Length];

for(int i = 0; i < m.Emails.Length; i++)

{

Emails1[i] = m.Emails[i];

}

int j = 0;

for(int i = m.Emails.Length;i< Emails1.Length; i++)

{

Emails1[i] = strings[j];

j++;

}

m.Emails = Emails1;

return m;

}

public static MailingList operator -(MailingList m, string str)

{

bool flag = false;

int k = 0;

for(int i = 0;i<m.Emails.Length;i++)

{

if (m.Emails[i] == str) { k = i; flag=true; break; }

}

if (flag)

{

string[] strings=new string[m.Emails.Length-2];

if (k % 2 == 0)

{

int j = 0;

for (int i = 0; i < m.Emails.Length; i++)

{

if (i == k || i == k + 1) { continue; }

else

{

strings[j] = m.Emails[i];

j++;

}

}

m.Emails= strings;

return m;

}

else if (k % 2 != 0)

{

int j = 0;

for (int i = 0; i < m.Emails.Length; i++)

{

if (i == k || i == k - 1) { continue; }

else

{

strings[j] = m.Emails[i];

j++;

}

}

m.Emails = strings;

return m;

}

else

{

return m;

}

}

else

{

return m;

}

}

public string this[string str]

{

get

{

bool flag = false;

int k = 0;

for(int i = 0;i<Emails.Length;i++)

{

if (Emails[i] == str) { flag = true; k = i; break; }

}

if (flag)

{

return $"у никнейма {str} емэйл - {Emails[k+1]}";

}

else

{

return $"{str} - Нет такого никнейма";

}

}

}

public override string? ToString()

{

string str = $"Список никнеймов ";

if (DateCreate != null)

{

str += $"и емэйл-адресов от {DateCreate}. Количество - {Emails.Length / 2}\n";

}

int k = 1;

for(int i = 0; i < Emails.Length; i++)

{

if(i%2== 0)

{

str += $"\t{k} - {Emails[i]} - {Emails[i+1]}\n";

k++;

}

}

return str ;

}

}

}

