**VectorName.cs**

namespace ConsoleApp1

{

class vectorName

{

public string name;

public int id;

public vectorName(string name,int id)

{

this.id = id;

this.name = name;

}

}

}

**Vector.cs**

namespace ConsoleApp1

{

partial class Vector:IComparable

{

private int[] massive;

private int count;

private static int status;

private readonly int ID;

public Vector() : this(new[] { 1, 2, 3, 4, 5 }, 5, 0) { }

public Vector(int[] massive1) : this(massive1, massive1.Length) { }

static Vector() { status = 0; Add(); }

private Vector(int[] massive1, int count1, int status1 = 0) { massive = massive1; count = count1; status = status1; ID = count.GetHashCode(); Add(); }

internal const int random = 15;

public int[] Massive

{

get

{

return massive;

}

set

{

massive = value;

}

}

public int Count

{

get

{

return count;

}

set

{

count = value;

}

}

public int Status

{

get

{

return status;

}

set

{

if (value > 2)

{

Console.WriteLine("Недопустимое значение кода ошибки");

status = 1;

}

else

{

status = value;}}}

public int Id

{

get

{

return ID;

}

}

private static int size = 0;

private static void Add()

{

size++;

}

public static void OutputInfo()

{

Console.WriteLine($"Количество экземпляров класса {size}");

}

public int CompareTo(object obj)

{

throw new NotImplementedException();

}

public int this[int i]

{

get

{

return massive[i];

}

set

{

massive[i] = value;}}}

partial class Vector

{

public static void part()

{

Console.WriteLine("Использовал partial в классе");

}

public override bool Equals(object obj)

{

if (obj == null) return false;

if (obj.GetType() != this.GetType()) return false;

Vector vect = (Vector)obj;

return vect.ID == this.ID;

}

public override int GetHashCode()

{

int hash = 0;

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

{

hash += massive[i];

}

hash = hash / count;

return hash;

}

public override string ToString()

{

return "Type " + base.ToString() + count + " " + ID;

}

public int sum()

{

int summ = 0;

foreach (int k in massive)

{

summ += k;

}

return summ;

}

public int sum(ref int num)//сумма с числом и с ref параметром

{

int summ = sum() + num;

return summ;

}

public int increase()

{

int Increase = 0;

foreach (int k in massive)

{

Increase \*= k;

}

return Increase;

}

public void watchfirstind(ref int index)

{

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

{

Console.Write(massive[i] + " ");

}

}

public bool nullinvector()

{

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

{

if (massive[i] == 0)

{

return true;}}

return false;

}

public double module()

{

double mod = 0;

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

{

mod = massive[i] \* massive[i];

}

mod = Math.Sqrt(mod);

return mod;}}}

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

string[] months = { "Январь", "Февраль", "Март", "Апрель", "Май", "Июнь", "Июль" , "Август","Сентябрь","Октябрь","Ноябрь","Декабрь"};

int n = Convert.ToInt32(Console.ReadLine());

var MonthsLen = from t in months where t.Length == n select t;

foreach(var mon in MonthsLen)

{

Console.WriteLine(mon);

}

Console.WriteLine("/////////");

var summerMon = from t in months where t == "Январь" || t == "Февраль" || t == "Декабрь" || t == "Июнь" || t == "Июль" || t == "Август" select t;

foreach(var mon in summerMon)

{

Console.WriteLine(mon);

}

Console.WriteLine("/////////");

var monthorderby = from t in months

orderby t

select t;

foreach(string t in monthorderby)

{

Console.WriteLine(t);

}

Console.WriteLine("/////////");

var MonthStLeter = from t in months where t.Substring(0,1) == "И" && t.Length <= 4 select t;

foreach(string t in MonthStLeter)

{

Console.WriteLine(t);

}

Console.WriteLine("/////////");

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

list.Add(new Vector(new int[] { 12, 3, 65, 12 }));

list.Add(new Vector(new int[] { 3, 65, 12,123,34 }));

list.Add(new Vector(new int[] { 12, 3, 65, 12,124,641,32,42 }));

list.Add(new Vector(new int[] { 1,2,43,54,5,2,1 }));

list.Add(new Vector(new int[] { 1, 2, 43, 54, 5, 0, 1 }));

var nullvector = from vect in list

where vect.nullinvector() == true

select vect;

Console.WriteLine("Вектора содержащие ноль");

foreach(Vector vect in nullvector)

{

Console.WriteLine(vect.Id);

}

var vectorOrder = from vect in list

orderby vect.Id

select vect;

Console.WriteLine("Вектора по возрастанию");

foreach (Vector vect in vectorOrder)

{

Console.WriteLine(vect.Id);

}

var vectorLength = from vect in list

where vect.Massive.Length==3 || vect.Massive.Length==5 || vect.Massive.Length==7

select vect;

Console.WriteLine("Вектора с заданной длиной");

foreach (Vector vect in vectorLength)

{

Console.WriteLine(vect.Id);

}

var vectorMax = from vect in list

orderby vect.Massive.Length descending

select vect;

Console.WriteLine("Вектор с максимальной длиной: " + vectorMax.First().Massive.Length);

var myinquery = list.Where(s => s.Id < 10).Select(s => s.Massive).Skip(1).Count();

Console.WriteLine(myinquery);

List<vectorName> namesofVectors = new List<vectorName>()

{

new vectorName("a",0),

new vectorName("b",1),

new vectorName("c",2),

new vectorName("d",3),

new vectorName("e",4),

new vectorName("f",5),

new vectorName("r",6),

new vectorName("k",7),

new vectorName("y",8),

};

var listName = list.Join(namesofVectors,

t => t.Id,

p => p.id,

(t, p) => new { Name = p.name, ID = p.id, thisMassive = t.Massive }

);

foreach(var info in listName)

{

Console.WriteLine("Имя вектора " + info.Name + " id вектора " + info.ID);}}}}