```
//____________________file my.h
#include <iostream> // вектор с целыми координатами,  длина задается в конструктре
#include<stdlib.h>
using namespace std;


class my
{
	int *x,n; //вектор и его длина
	public:
		my(){n=0;x=0;} //конструктор без параметров
		my(int );      //конструктор с параметром
		my (my &&v);   //конструктор перемещения
		my (const my&v);//констрруктор копии
	 ~my();   //деструктор
		my operator +(const my &a);  //сложение покоординатное, длина =min
		my operator /(const my &a);  //деление нацело покоординатное, длина =min
		my &operator =(const my &a);  //присваивание копированием
		my &operator =(my &&a);  //присваивание перемещением
		friend void pr(const my &); //печать
};
//____________________file my.cpp
#include"my.h"
my::my(int N)
{
	n=N;
	x= new int[n];
	for (int i=0;i<N;i++)
		x[i]=rand()%5;
}
my::my (const my&v)//вызывается при создании нового объекта, память под который еще не выделена
{
	n=v.n;
```

```
	x= new int[n];
	for(int i=0;i<n;i++)
		x[i]=v.x[i];
}
my::~my()
{
	if(x) delete []x;
}
my & my::operator=(const my &v)
{
	if(this==&v)return *this;  //самоприсваивание
	if(n<v.n) //если выделенной памяти может не хватить, выделяем новую
	{
		delete []x;
		x= new int [v.n];
	}
	n=v.n;
	for(int i=0;i<n;i++)
		x[i]=v.x[i];
	return *this;
}
my & my::operator=(my &&v)
{
	if(this==&v)return *this;
	if (x)delete []x; // очистка ранее выделенной памяти
	x= v.x; //передаем ресурс  другому
	n=v.n;
	v.x=0; //чтобы деструктор не очистил переданную память
	return *this;
}
my::my(my &&v)
{
```

```
	n=v.n;
	x=v.x; //передаем ресурс  другому
	v.x=0; //чтобы деструктор не очистил переданную память

}
my my::operator +(const my &a)
{
	int y=n<a.n?n:a.n; my tmp(y); //строим временный объект
	for (int i=0;i<tmp.n;i++)
	tmp.x[i]=x[i]+a.x[i];
	return tmp; //ресурс временного объекта может быть передан другому
}
my my::operator /(const my &a)
{
	int y=n<a.n?n:a.n;
	my tmp(y); //строим временный объект
	for (int i=0;i<tmp.n;i++)
		{
			if(a.x[i]==0)
				throw(0); //попытка деления на 0, выброс ошибки
		tmp.x[i]=x[i]/a.x[i];
		}
	return tmp; //ресурс временного объекта может быть передан другому
}
void pr(const my &v)
{
	if(!v.x)
		throw(1); //попытка обращения к объекту, который не сущетвует
	for (int i=0;i<v.n;i++)
		cout<<v.x[i]<<" ";
	cout<<"\n";
}
```

```
//____________________file main.cpp
#include"my.h"


int main()
{
	my v1(15);
	pr(v1);
	try{    // далее блок, который надо отследить на ошибки
		my v2=v1,v3;  //конструктор копии
		v3=my(29)+my(17); //присваивание перемещением
		pr(v3);
		v3=v1/v3+v2/v3/v3; //копирование и присваивание перемещением
		pr(v3);
		pr(v2);
		v1=v1;  //самоприсваивание
		v2=move(v1); //вызов присваивания с перемещением, v1 станет недоступен
		pr(v1); //ошибка
	}
	catch(int i){  //ловим ошибки
		if(i==0)
			cout<<"zERO DIVISION\n";
		if(i==1)
			cout<<"Ego net\n";
		else
			cout<<i<<"\n";
	}
	return 0;
}
```