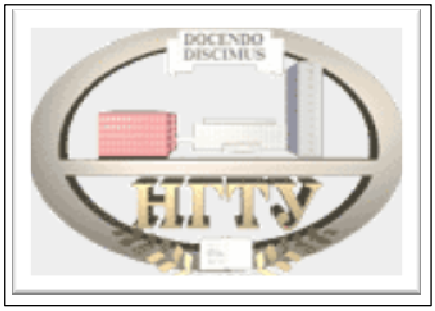
Новосибирский государственный технический университет

Кафедра вычислительной техники



Пояснительная записка к Расчётно-графической работе.

Вариант 2

«Использование графической библиотеки “Graphics.h”»

Группа: АВТ-009
Студент: Мельников Т. А.

Новосибирск
2010.

1. Задание

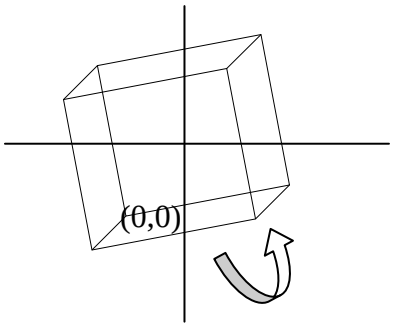
Вариант 2. По экрану движется вращающийся куб, изображаемый в виде граней. Вращение управляется с клавиатуры.

2. Основные идеи и методы решения

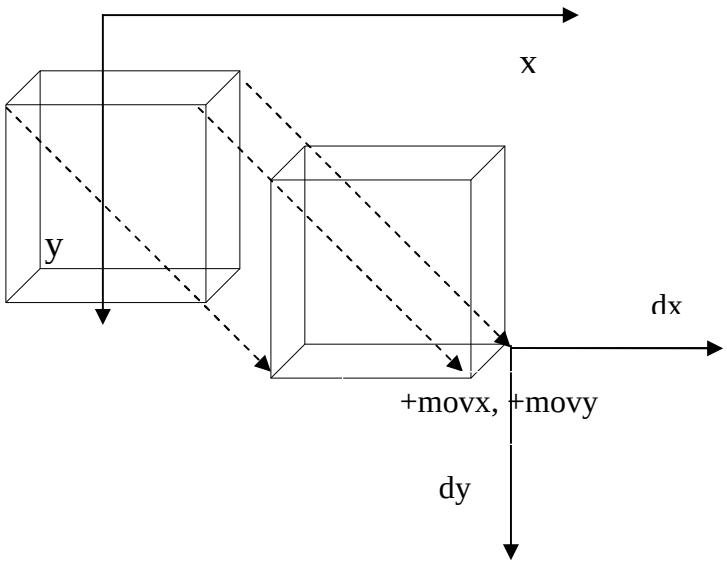
Для реализации движения куба необходимо решить следующие проблемы:

- движение точек в системе координат относительно экрана;
- вращение куба относительно точки и параллельный перенос вершин куба из начала координат;
- изменение направления движения куба и отскока его от границ экрана.

В программе используется две прямоугольные системы координат. При реализации вращательного используется система с началом координат в центре куба.



Затем моделируется движение центра куба в основной системе координат, с соответствующим переносом вершин.



В соответствии с выбранной геометрической моделью в программе реализуюися различные виды движения:

1) ***Вращательное движение*** достигается за счет перемножения каждого трехмерного вектора координаты на матрицу поворота. Для того, чтобы куб вращался вокруг своего центра, вектор координату перемножаем по очереди на матрицу поворота вокруг каждой оси (OX, OY, OZ). Вращение происходит вокруг точки начала координат.
2) ***Линейное движение.*** Т.к. можно вращать куб только вокруг начала координат, то трехмерные координаты не меняются (отсутствует линейное движение) и, следовательно, проецируемые двумерные координаты всегда стоят на месте. Линейное движение происходит за счет прибавления к двумерной координате постоянно увеличивающейся (уменьшающейся) на dx, dy переменной.
3) ***Контроль движения.*** Слежение за процессом движения производится рядом условий, в которых задействованы координаты вершин куба. Условия необходимы для удержания фигуры в ограниченной на экране области. Кроме того, используются две переменные, содержащие информацию о направлении движения и одна для управления направления вращения.

3. Функциональное описание программы

Программа содержит следующие данные.

```
#define A 50      // половина грани куба
double X3[8],Y3[8],Z3[8], X2[8],Y2[8];
                  // глобальные массивы трехмерной и двумерной координаты
double L=0.5, fi=45*(3.14159265358/180.0); //fi угол проецирования координаты
int dx=2,dy=2;          // скорость линейного движения
int k=0;                // счетчик градусов
char c;                 // направление вращательного движения
long double degree;     // градус поворота
```

Программа состоит из главного цикла void main() и из функций cube(), line(),rotate (long double degree), projection (double L, double fi)

- *cube()* – задование трехмерных координат куба;
- *line()* – прорисовка куба;
- *rotate (long double degree)* – вращение куба;
- *projection (double L, double fi)* – проецирование трехмерных координат.

Главный цикл. В теле главного цикла происходит определение направления вращения куба, вызов функций *rotate (long double degree), projection (double L, double fi), cube(), line(),* проверка условий отскакивания от стенок и изменение направления движения при совпадение условия.

```
void main (){
```

```
	int gdriver=DETECT, gmode=VGAHI, errorcode;
	initgraph(&gdriver, &gmode, "e:\\bc31\\bgi");
	errorcode = graphresult();
	if (errorcode != grOk)
	{
		printf("Graphics error: %s\n", grapherrormsg(errorcode));
		printf("Press any key to halt...");
		getch();
		return;
	}

	double L=0.5, fi=45*(3.14159265358/180.0);
	//fi угол проецирования, L коэффициент проецирования
	int maxx=getmaxx();
	int maxy=getmaxy();
	int i;
	int dx=2,dy=2;          // скорость линейного движения
	int movx=100, movy=100 ;
	long double degree;     // градус поворота

	cube();
	char c;             // направление вращательного движения
	while(c!='r') {     // пока не нажата клавиша "r"

	c=getch();
	if (c=='a') {degree= 3.14159265358/180.0;}
	//нажата "a" по часовой стрелке
	if (c=='d') {degree= -3.14159265358/180.0;}
	// нажата "d" против часовой стрелки
	rotate (degree);

	projection (L,fi);       //проецирование трехмерных координат
	for (i=0;i<8;i++) {
		X2[i]=X2[i]+movx;    // "вынос" куба из начала координат
		Y2[i]=Y2[i]+movy;
		if ((X2[i]>=maxx || X2[i]<=0) && (X2[i]+3*dx>=maxx ||
X2[i]+3*dx<=0)) // условие отскока от горизонтальной стенки
			{ dx=-dx;}
		if ((Y2[i]>=maxy || Y2[i]<=0) && (Y2[i]+3*dy>=maxy ||
Y2[i]+3*dy<=0)) // условие отскока от вертикальной стенки
			{ dy=-dy;}
	}
	movx+=dx;   //процесс смещения координат куба
	movy+=dy;
	setfillstyle(1,BLACK);
	bar(0,0,maxx,maxy); // затирание старого куба
	setcolor (WHITE);
	line ();                //создание нового куба
	delay(7);
}
}
```

Функция задания координат *cube()* в качестве параметров получает глобальные массивы трехмерных координат X3, Y3, Z3 и половину длины грани A. Координаты расположены вокруг начала координат.

```
void cube () {
	X3[0]=-A; Y3[0]=-A; Z3[0]=-A;      X3[4]=-A; Y3[4]=-A; Z3[4]=A;
	X3[1]=A; Y3[1]=-A; Z3[1]=-A;    X3[5]=A; Y3[5]=-A; Z3[5]=A;
	X3[2]=A; Y3[2]=A; Z3[2]=-A;  X3[6]=A; Y3[6]=A; Z3[6]=A;
	X3[3]=-A; Y3[3]=A; Z3[3]=-A;    X3[7]=-A; Y3[7]=A; Z3[7]=A;
}
```

Функция прорисовки куба *line()* в качестве параметров получает глобальные массивы двумерных координат X2, Y2.

```
void line () {
      for (int i=0;i<7;i++) { if (i==3) continue;
      line (X2[i],Y2[i],X2[i+1],Y2[i+1]);}
      for(i=0;i<4;i++){
            line (X2[i],Y2[i],X2[i+4],Y2[i+4]);}
      line (X2[0],Y2[0],X2[3],Y2[3]);
      line (X2[4],Y2[4],X2[7],Y2[7]);}
```

Функция вращения куба вокруг точки начала координат *rotate (long double degree)* в качестве параметров получает значения градуса поворота и глобальные массивы трехмерных координат. Производит перемножение вектора координаты на матрицу поворота.

```
void rotate (long double degree) {
      for(int i=0;i<8;i++) {                          //вращение вокруг оси OX
            Y3[i]= Y3[i]*cos(degree)-Z3[i]*sin(degree);
            Z3[i]= Y3[i]*sin(degree)+Z3[i]*cos(degree);
      }
      for(i=0;i<8;i++) {                              //вращение вокруг оси OZ
            X3[i]= X3[i]*cos(degree)-Y3[i]*sin(degree);
            Y3[i]= X3[i]*sin(degree)+Y3[i]*cos(degree);
      }
      for(i=0;i<8;i++) {                              // вращение вокруг OY
            X3[i]= X3[i]*cos(degree)-Z3[i]*sin(degree);
            Z3[i]= X3[i]*sin(degree)+Z3[i]*cos(degree);
      }
}
```

Функция проецирования трехмерных координат *projection (double L, double fi)* в качестве параметров передается угол и коэффициент проецирования. Использует глобальные массивы трехмерных и двумерных координат.

```
void projection (double L, double fi){
      for (int i=0;i<8;i++) {
            X2[i]=X3[i]+Z3[i]*L*cos(fi);
            Y2[i]=Y3[i]+Z3[i]*L*sin(fi);
      }
}
```