

Программирование на языках высокого уровня

Кафедра
«Прикладная математика и информатика»
Ахмедханла д.М.

Графика C++

В графическом режиме экран представляет собой совокупность точек (пикселей)

(0,0)



(639, 479)

Графика C++

Для использования функций графического режима, необходимо подключить к программе заголовочный файл

<graphics.h>

В графическом режиме присутствует невидимый указатель (курсор)

Функции графического режима

```
initgraph(int *driver, int *mode, char *<путь>);
```

инициализация графического режима

Если в качестве параметра `driver` использовать константу `DETECT`, происходит автоматическое распознавание драйверов.

Функции графического режима

```
intgraphresult ( );
```

-возвращает код ошибки, можно поставить после инициализации. Если функция выполнена успешно – возвращает 0

Функции графического режима

```
outtext (const char *text);
```

- выводит строку символов `text` с текущего положения указателя

Функции графического режима

```
outtextxy (int x, int y, const char *text);
```

 выводит строку символов начиная от точки с координатами (x,y)

Цвет выводимых символов задается функцией `setcolor`, шрифт – функцией `settextstyle`)

Функции графического режима

```
setcolor( int <цвет>);
```

- задает цвет вывода текста, линий и фигур (согласно таблице цветов)

```
setbkcolor ( int <цвет>);
```

- задает цвет фона

Таблица цветов

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Black	Blue	Green	Cyan	Red	Purple	Brown	Grey	Dark Grey	Light Blue	Bright Green	Cyan	Magenta	Light Purple	Yellow	White

Функции графического режима

`getmaxx ();`

 функция возвращает максимальное количество столбцов экрана в графическом режиме

`getmaxy ();`

- максимальное количество строк экрана;

Функции графического режима

```
setfillstyle (int <стиль>, int <цвет>);
```



устанавливает стиль и цвет заливки

```
floodfill (int x, int y, int <цвет>);
```

- заливка замкнутой поверхности от точки с координатами (x, y)

Функции графического режима

`cleardevice ();`

- очистка экрана в графическом режиме

`closegraph ();`

- закрывает графический режим

Пример программы

«Вывести надпись в центре экрана»

```
#include<iostream>
```

```
#include<conio.h>
```

```
#include<graphics.h>
```

```
using namespace std;
```

```
int main()
```

```
{ int k; cin>>k; // цвет надписи
```

Пример

```
int grdriver=ДЕТЕСТ;  
int grmode;  
initgraph(&grdriver, &grmode, "");  
int errorcode = graphresult();  
if (errorcode != grOk)  
{ cout<<"\n ОШИБКА!!! \n";  
  exit(1); }
```

Пример

```
cleardevice();           // Очистка экрана
setbkcolor (3);         // Фон бирюзовый
setcolor (k);           // Цвет символов
int  x = getmaxx()/2;
int  y = getmaxy()/2;
//надпись в центре
outtextxy (x, y, "___ GRAPHIGS ___");
getch();
closegraph();}
```

Построение простейших фигур

`moveto (int x , int y);`

- перемещает указатель в точку (x, y).

`moverel (int dx, int dy);`

- перемещает указатель на dx и dy от текущего положения указателя.

Построение простейших фигур

```
putpixel ( int x, int y, int Цвет);
```

 зажигает точку (x, y) заданным цветом, который устанавливается по таблице цветов

Построение простейших фигур

`line (int x1, int y1, int x2, int y2);`

 рисует линию от точки (x1, y1) до точки (x2, y2)

`lineto (int x, int y);`

- рисует линию от текущего положения курсора-указателя до точки (x, y).

Построение простейших фигур

```
circle ( int x, int y, int R );
```

 окружность радиусом R с центром в точке (x, y) .

Цвет задается функцией `setcolor`

Построение простейших фигур

```
allipse (int z, int y, int Начало, int Конец, int  
        RX, int RY);
```

 рисует эллипс или дугу эллипса с центром в точке (x,y)

Начало и Конец дуги задается в градусах
RX и RY – горизонтальный и вертикальный радиусы

Построение простейших фигур

```
arc ( int x, int y, int Начало,  
      int Конец, int Радиус);
```

- вычерчивает дугу окружности с заданным Радиусом.

Начало и Конец задают круговые координаты начало и конец точек линии дуги (против часовой стрелки от Начало до Конца (в радианах)), цвет линии задается функцией `setcolor`

Построение простейших фигур

```
bar (int x1, int y1, int x2, int y2);
```

-рисует закрашенный прямоугольник

```
bar3d( int x1, int y1, int x2, int y2,  
       int <глубина>, int <грань>);
```



вычерчивает параллелепипед

Цвет и стиль заливки задается функцией `setfillstyle`

Задача

Нарисовать в центре экрана круг
и залить его определенным цветом

Программа

```
#include<iostream>
#include<conio.h>
#include<graphics.h>
using namespace std;
int main()
{int k;
    cout<<" Введите цвет заливки \n"; cin>>k;
```

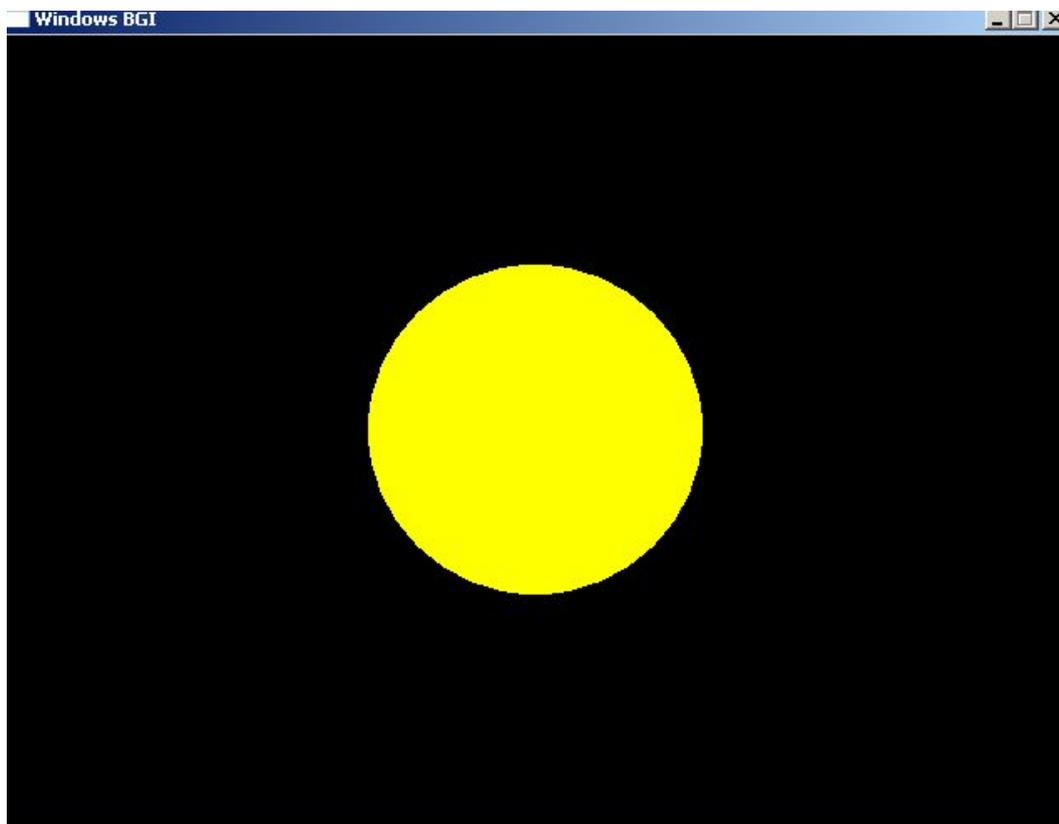
Программа

```
// инициализация графического режима
int grdriver=DETECT;  int grmode;
initgraph(&grdriver, &grmode, " ");
int errorcode = graphresult();
if (errorcode != grOk)
{ cout<<"\n SOS ! \n";
  getch();
  exit(1); }
```

Программа

```
cleardevice();           // очистка экрана
setcolor(k);            // цвет линии
int x = getmaxx()/2;    // координаты центра экрана
int y = getmaxy()/2;
circle ( x, y, 100);    // рисует круг
setfillstyle (SOLID_FILL , k); // стиль и цвет заливки
floodfill (x, y, k);    // заливка замкнутой поверхности
getch();
closegraph();          // закрыть графический режим
}
```

Результат на екране



Задача

Нарисовать в центре экрана окружности разного цвета и радиуса.

Программа

```
#include<iostream>
#include<conio.h>
#include<graphics.h>
using namespace std;
int main()
{ int k;
  cout<<" Введите радиус круга\n"; cin>>k;
```

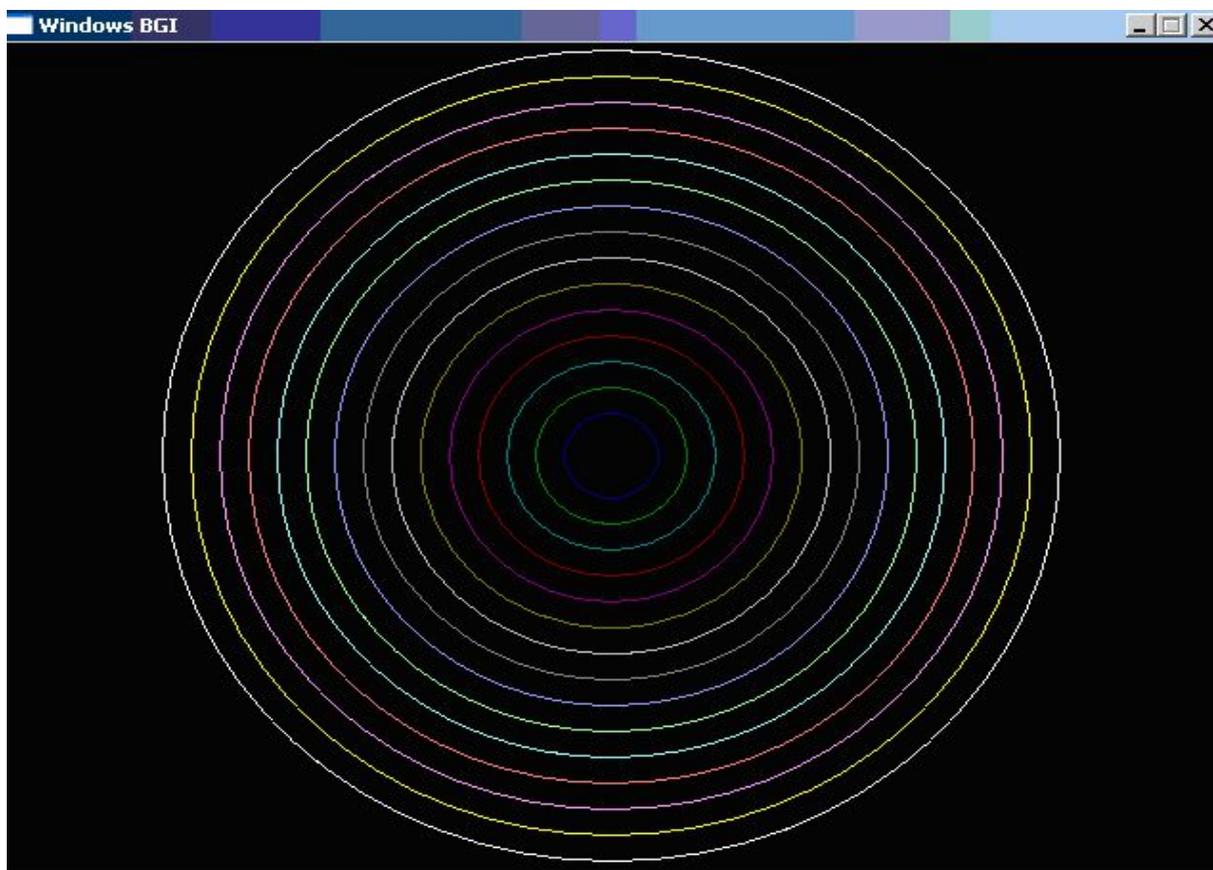
Программа

```
int grdriver=DETECT;
int grmode;
initgraph(&grdriver, &grmode, "");
int errorcode = graphresult();
if (errorcode != grOk)
    { cout<<"\n ОШИБКА!!!\n";
      getch();
      exit(1); }
```

Программа

```
cleardevice(); // Очистка экрана
int x = getmaxx()/2;
int y = getmaxy()/2;
for ( int i =1; i<=15; i++)
    { setcolor (i);
      circle (x, y, k+15*i);
      delay (50); }
getch();
closegraph(); }
```

Результат на екране



Построение движущихся изображений

`imagesize (intx1, inty1, intx2, inty2);`



определяется размер области в которой поместится рисунок ;

`malloc (int S);`

- выделение динамической области памяти под хранение образа рисунка;

Построение движущихся изображений

```
getimage ( int x1, int y1, int x2, int y2, void * p);
```

где

р – указатель на динамическую память,

вычисленную с помощью функции malloc

Данная функция забирает образ рисунка в

динамическую область памяти;

Построение движущихся изображений

`putimage (int x, int y, void p, <режим>);`

- функция выводит сохраненный образ рисунка на экран, где (x, y) – координаты левого верхнего угла прямоугольника;

<режимы>-

`COPY_PUT` - изображение видно

`XOR_PUT` - изображение не видно

Программа

Перемещение Куба с надписью по экрану с левого верхнего в правый нижний угол

```
#include<iostream>
```

```
#include<conio.h>
```

```
#include<graphics.h>
```

```
using namespace std;
```

```
int main()
```

Программа

```
{  
    int grdriver=DETECT;  
    int grmode;  
    initgraph(&grdriver, &grmode, "");  
    int errorcode = graphresult();  
    if (errorcode != grOk)  
    { cout<<"\n ОШИБКА!!!\n";  
      getch();  
      exit(1);  
    }  
}
```

Программа

```
setbkcolor (3);           // фон бирюзовый
setcolor (10);           // цвет надписи
setfillstyle (SOLID_FILL, 4);
```

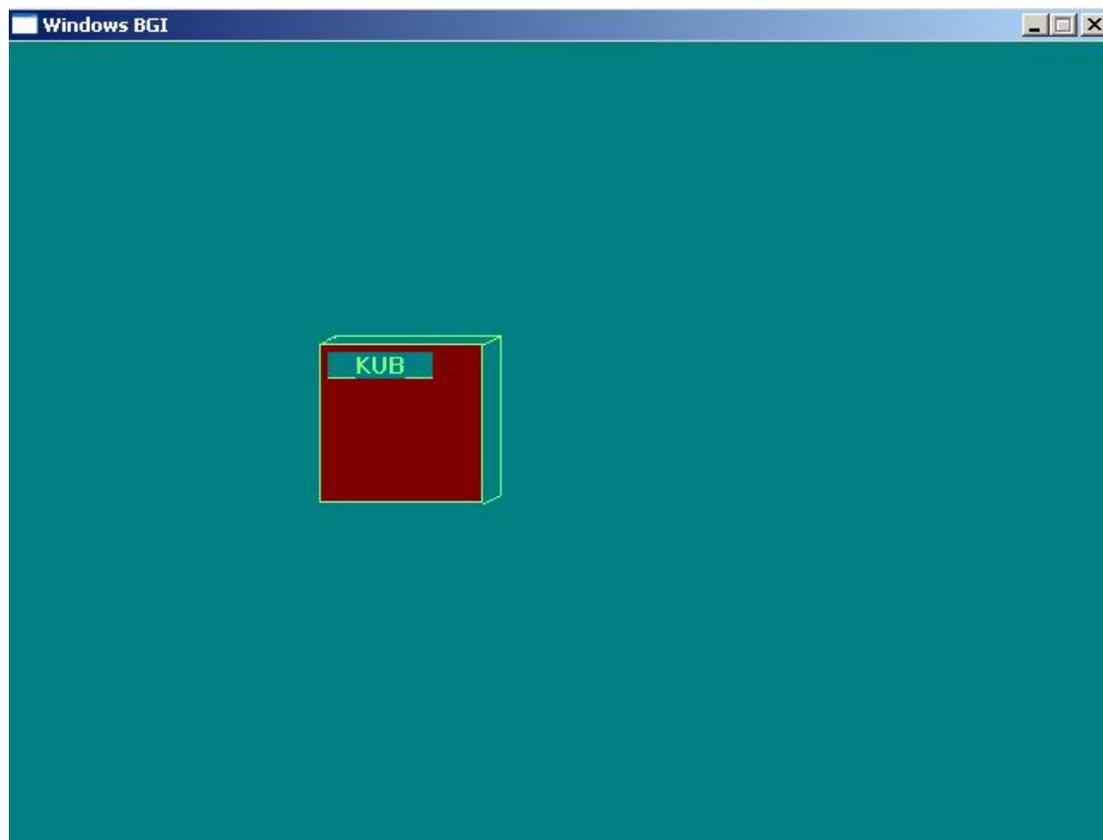
Программа

```
for ( int i =1; i<=200; i++)  
    { cleardevice();                //очистка экрана  
      //закрашенный параллелепипед  
      bar3d (5+i, 5+i, 100+i, 100+i, 10, 10);  
      // надпись  
      outtextxy ( 10+i, 10+i , "__KUB__");  
      delay (50); }                // удаление
```

Программа

```
    getch();  
    closegraph();  
}
```

Результат на екране





Спасибо за внимание!