

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

ЛАБОРАТОРНАЯ РАБОТА № 2

Выполнили:
Студенты группы: ИП-715
Кузнецов Егор и Винтер Антон

Проверил: доцент кафедры ПМиК
Милешко А.В.

Содержание

1. Задание.....	3
2. Лабораторная работа №2 часть 1.....	4
3. Лабораторная работа №2 часть 2.....	7
4. Лабораторная работа №2 часть 3.....	8
5. Лабораторная работа №2 часть 4.....	11
6. Лабораторная работа №2 часть 5.....	13

1. Задание

LAB 2 Процессы и асинхронное взаимодействие:

1. Тщательно изучить библиотеку VinGraph.
2. Используя функции библиотеки VinGraph, нарисовать абстрактную картину, которой представлены (почти) все доступные графические элементы.
3. Заставить нарисованные элементы двигаться независимо друг от друга с помощью параллельных процессов (можно изменять во времени положение, цвет, размеры, конфигурацию графических элементов). Предусмотреть завершение программы по нажатию на любую клавишу.
4. Нарисовать нечто, движущееся по замкнутой кривой. Организовать изменение траектории движения по нажатию на клавиши (организуя взаимодействие процессов через общую область памяти (shared memory)). В качестве фона можно использовать (оживленную) картину, созданную на предыдущих этапах работы.
5. Затем последнюю программу сделать с помощью нитей в одном процессе.

2. Лабораторная работа №2 часть 1

Библиотека VinGraph представляет такое же простое средство для вывода графики, как Borland под MS-DOS, и даже еще более простое, если говорить об анимации созданных изображений.

Вывод графики производится в окне графического терминала VinGraph. Одновременно может быть запущено множество графических терминалов при условии, что каждому из них задается свое уникальное имя. Терминал VinGraph может быть запущен из командной строки, в этом случае он работает, пока пользователь его не закроет. Терминал VinGraph может быть также запущен из программы пользователя, в этом случае он закрывается при (успешном) завершении программы. Программа пользователя, точнее процесс, прикрепляется к какому-то одному терминалу и работает с ним. Несколько процессов могут одновременно выводить графику на один терминал. Один процесс не может делать вывод сразу на несколько терминалов, да это и не нужно.

Все функции библиотеки VinGraph являются нить-безопасными (thread-safe), т.е. могут свободно вызываться из разных нитей вашего процесса. В архитектурном плане сам терминал VinGraph состоит из двух нитей: одна служит для взаимодействия со средой Photon, другая взаимодействует с прикладными программами посредством интерфейса функций библиотеки VinGraph. Пользователь вызывает библиотечную функцию, и та посылает соответствующее сообщение графическому терминалу (и принимает от него ответ).

Подключение библиотеки к программе:

В начало текста программы нужно включить файл, содержащий объявления прототипов функций и другие нужные вещи:

```
#include <vingraph.h>
```

Вначале выполнения программы, до вызова графических функций, необходимо присоединиться к терминалу VinGraph, а в конце -- отсоединиться от него.

Функция [ConnectGraph\(\)](#) может вызываться с аргументом, задающим имя терминала, если вас не устраивает стандартное имя, или вы хотите работать с несколькими терминалами. Данная функция также запускает терминал VinGraph (с соответствующим именем), если он еще не был запущен. Функция [CloseGraph\(\)](#) закрывает соединение с терминалом, а также завершает работу терминала, если он был запущен вашей программой.

Компиляция программы осуществляется с помощью команды

```
cc prog.cpp -l vg
```

Здесь опция `l` указывает файл библиотеки VinGraph (`libvg.a`). Теперь вы можете запускать вашу программу:

```
./a.out
```

Чтобы запустить терминал VinGraph из командной строки, напечатайте

```
vg
```

Функции рисования

Библиотека VinGraph позволяет выводить на экран следующие элементы графики: точки, прямые и ломаные линии, прямоугольники, многоугольники, окружности, эллипсы, дуги, а также сетки, текстовые надписи и изображения. Кроме того, из этих элементов можно составлять новые композитные элементы (рисунки).

Установка цвета

Цвет задается числом типа `int`, в котором три младших байта кодируют интенсивность трех основных цветов -- красного (`r`), зеленого (`g`) и синего (`b`). Для сведения всех трех составляющих цвета в одно целое число используется макрос `RGB(r, g, b)`, определенный в файле `vingraph.h`. Так, черному цвету соответствует `RGB(0, 0, 0)`, белому -- `RGB(255, 255, 255)`, а промежуточные значения соответствуют всем остальным цветам.

Управление графическими элементами и создание рисунков

Все графические элементы, созданные с помощью функций рисования, являются объектами, т.е. помнят свое состояние и могут выполнять определенные действия. Каждый графический элемент имеет идентификатор -- положительное целое число. Этот идентификатор возвращается функцией рисования.

Анимация

Библиотека VingGraph позволяет изменять такие параметры отображаемых элементов, как цвет, местоположение, размер и др. Это позволяет строить изображения, динамически меняющиеся по алгоритму пользователя.

Определение текущих параметров графических элементов

Параметры всех графических элементов и самого графического терминала находятся под контролем программы пользователя. Но все же в некоторых

случаях, особенно при работе множества нитей, бывает удобно запросить информацию о состоянии непосредственно у графического элемента. Такая информация может также понадобиться при отладке программы.

Ввод с клавиатуры

Для ввода символа с клавиатуры, когда активно окно терминала VinGraph, используется функция `char InputChar()`. Эта функция ждет нажатия на клавишу и возвращает введенный символ. Функция работает в режиме односимвольного ввода без эхо печати (сырой режим в терминологии терминалов UNIX). При нажатии управляющих клавиш формируются нестандартные коды.

Графический терминал VinGraph

Графический терминал VinGraph -- это приложение, с которым взаимодействуют функции библиотеки VinGraph.

Графический терминал VinGraph запускается автоматически функцией `ConnectGraph()` с параметрами, принятыми по умолчанию. Он может быть также запущен из командной строки следующего формата:

```
vg [-n NAME] [-a X Y W H] [-c R G B]
```

Все аргументы командной строки должны разделяться пробелами.

Квадратные скобки указывают на необязательность соответствующих опций, сами скобки не должны печататься.

Опция `-n` задает имя терминала.

Опция `-a` устанавливает координаты левого верхнего угла, ширину и высоту окна терминала.

Опция `-c` задает цвет фона путем указания трех составляющих цвета.

Одновременно может быть запущено множество терминалов при условии, что им заданы различные имена.

Переменная среды `VGOSC` определяет режим работы терминала: с использованием буфера внеэкрannого контекста или без него. Использование внеэкрannого контекста позволяет убирать мерцание при передвижении графических элементов, однако успешность реализации этого режима зависит от графического адаптера и его драйвера. Этот режим также создает дополнительную нагрузку на центральный процессор.

3. Лабораторная работа №2 часть 2

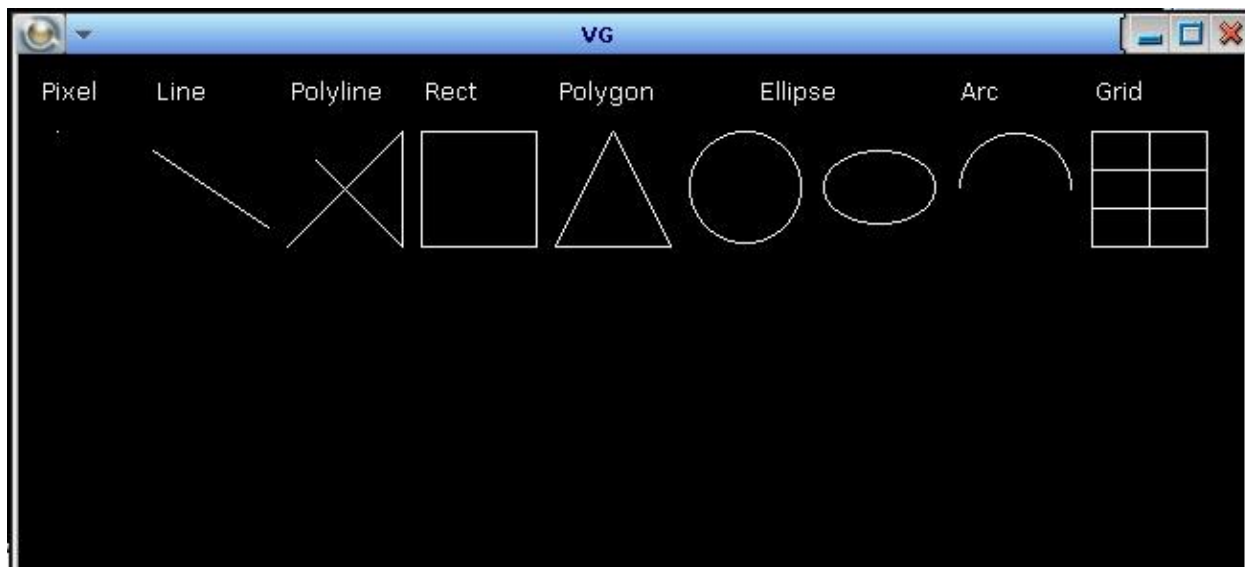


Рисунок 1. Демонстрация работы программы №2

Листинг:

```
prog2.cpp (/home/LAB2/prog2.cpp)
File Edit Search Type Buffer Marker Help
TextFont 9
#include <vingraph.h>
#include <stdio.h>
#include <unistd.h>

int main(){
    ConnectGraph();
    while(1) {
        Text (10, 10,"Pixel");
        Pixel (20, 40);
        delay(1000);

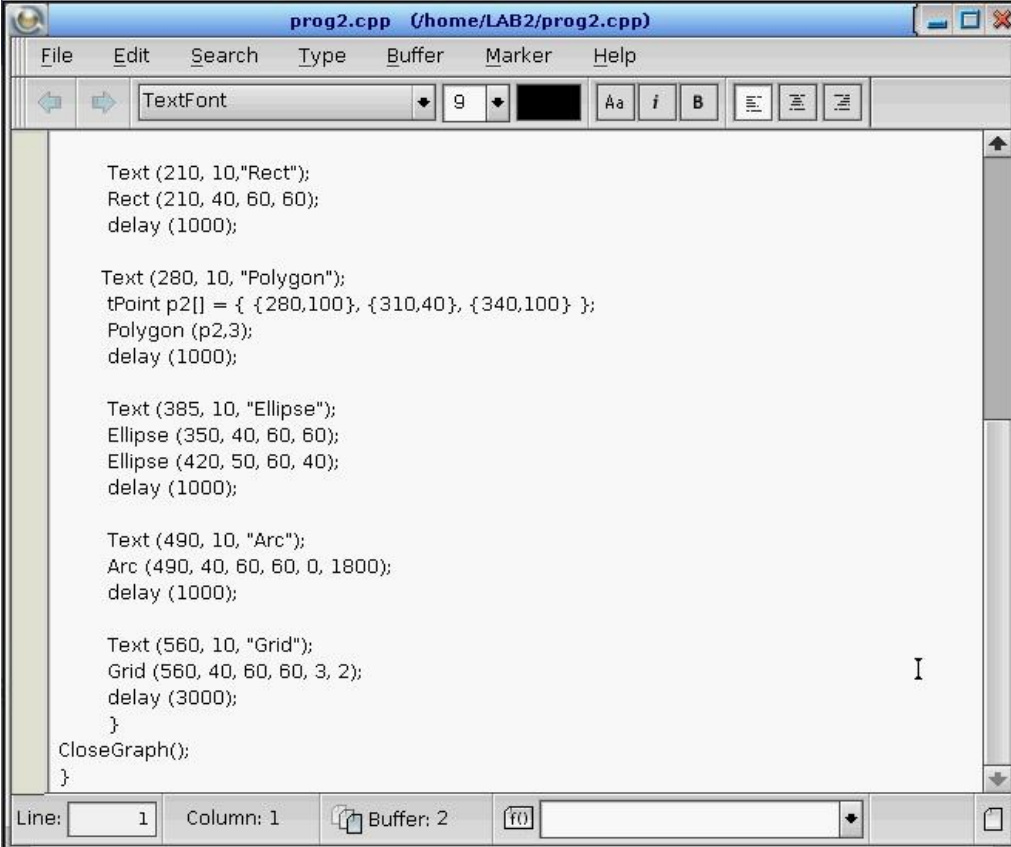
        Text (70, 10,"Line");
        Line (70, 50, 130, 90);
        delay (1000);

        Text (140, 10, "Polyline");
        tPoint p1[] = { { 140,100}, {200,40}, {200,100}, {155,55} };
        Polyline (p1,4);
        delay (1000);

        Text (210, 10,"Rect");
        Rect (210, 40, 60, 60);
        delay (1000);

        Text (280, 10, "Polygon");
    }
}
```

Рисунок 2. Текст программы №2(а)



```
prog2.cpp (/home/LAB2/prog2.cpp)
File Edit Search Type Buffer Marker Help
TextFont 9
Text (210, 10, "Rect");
Rect (210, 40, 60, 60);
delay (1000);

Text (280, 10, "Polygon");
tPoint p2[] = { {280,100}, {310,40}, {340,100} };
Polygon (p2,3);
delay (1000);

Text (385, 10, "Ellipse");
Ellipse (350, 40, 60, 60);
Ellipse (420, 50, 60, 40);
delay (1000);

Text (490, 10, "Arc");
Arc (490, 40, 60, 60, 0, 1800);
delay (1000);

Text (560, 10, "Grid");
Grid (560, 40, 60, 60, 3, 2);
delay (3000);
}
CloseGraph();
}
```

Line: 1 Column: 1 Buffer: 2

Рисунок 3. Текст программы №2(б)

4. Лабораторная работа №2 часть 3

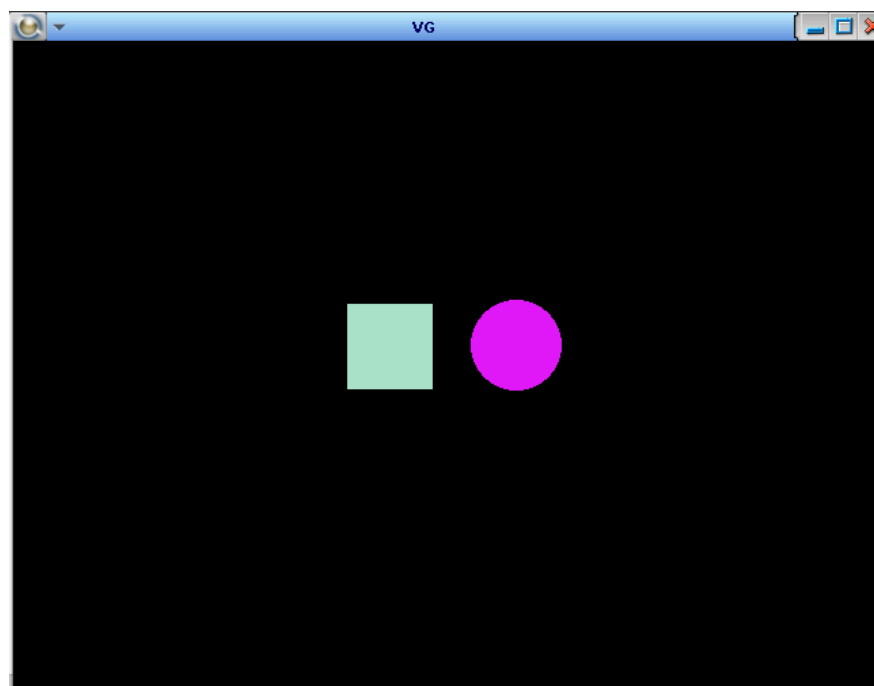


Рисунок 4. Демонстрация работы программы №3(а)

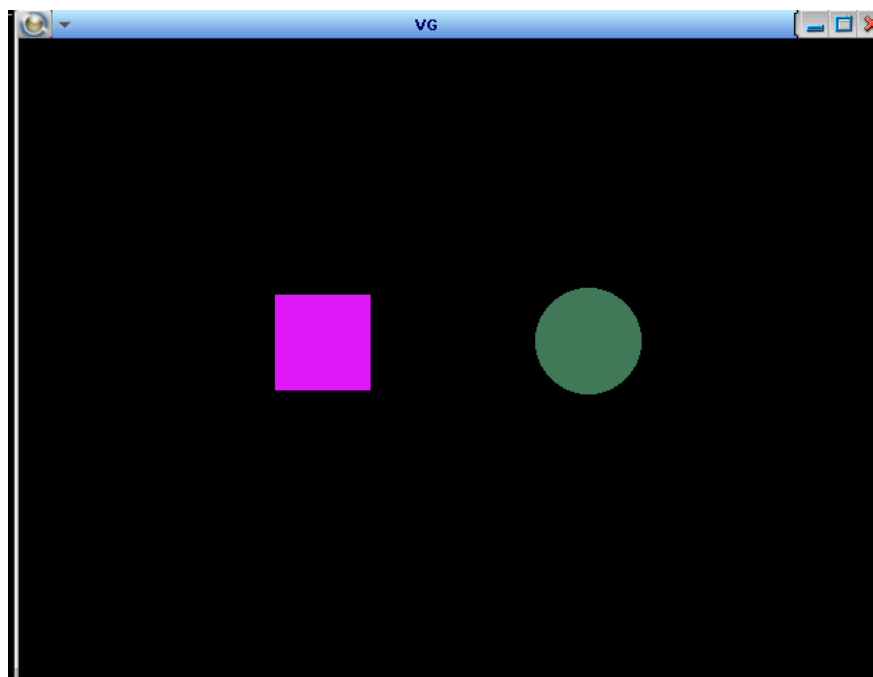


Рисунок 5. Демонстрация работы программы №3(б)

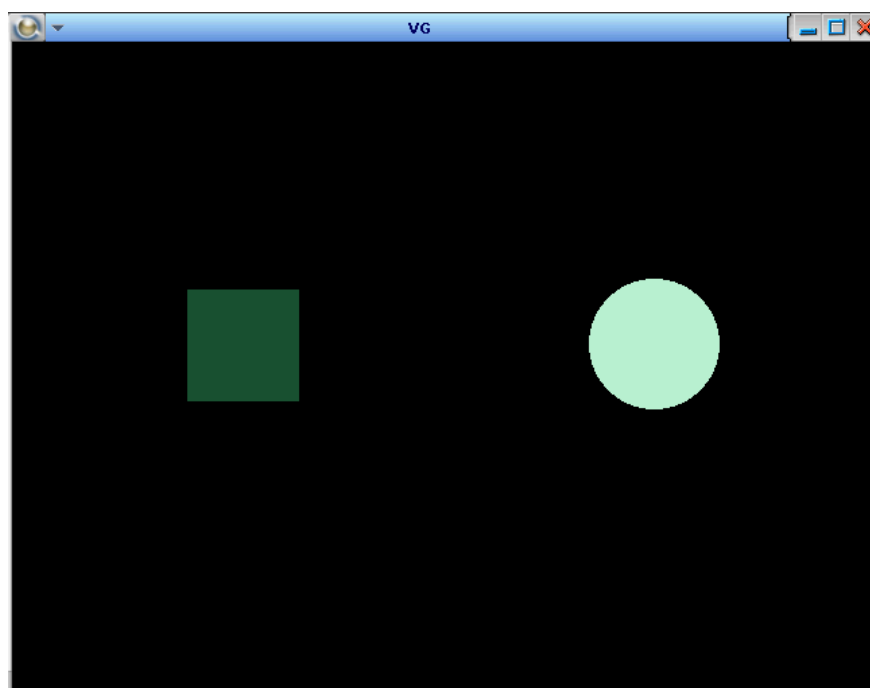


Рисунок 6. Демонстрация работы программы №3(в)

Листинг:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <process.h>
#include <signal.h>
#include <vingraph.h>
int main(void){
    char c;
    int rr = 125;
    int gg = 180;
    int bb = 150;
    ConnectGraph();
    int c1 = RGB(125, 180, 150);
    int f1 = Rect(300, 200, 50, 50, 0, c1);
    Fill(f1, c1);
    int c2 = RGB(135, 206, 250);
    int f2 = Ellipse(300, 200, 50, 50, c2);
    Fill(f2, c2);

    if (fork() == 0){
        for(int i=0; i < 100; i++){
            Move(f1, -2, 0);
            if ((i%5) == 0) {
                rr += 10;
                gg += 10;
                bb += 10;
                c1 = RGB(rr, gg, bb);
            }
            Fill(f1, c1);
            SetColor(f1, c1);
            if ((i % 10) == 0) {
                Enlarge(f1, 2, 2);
            }
            delay(100);
        }
        exit(0);
    }else{
```

```
        delay(100);
    }
    exit(0);
}else{

    if(fork() == 0){
        for(int i = 0; i < 100; i++) {
            Move(f2, 2, 0);
            if ((i % 5) == 0) {
                rr += 20;
                gg += 20;
                bb += 20;
                c2 = RGB(rr, gg, bb);
                Fill(f2, c2);
                SetColor(f2, c2);
            }
            if ((i % 10) == 0) {
                Enlarge(f2, 3, 3);
            }
            delay(100);
        }
        exit(0);
    }
    else {
        while (1) {
            c = InputChar();
            if (c) {
                CloseGraph();
                exit(0);
            }
        }
    }
}
return 0;
}
```

Рисунок 7. Текст программы №3

5. Лабораторная работа №2 часть 4

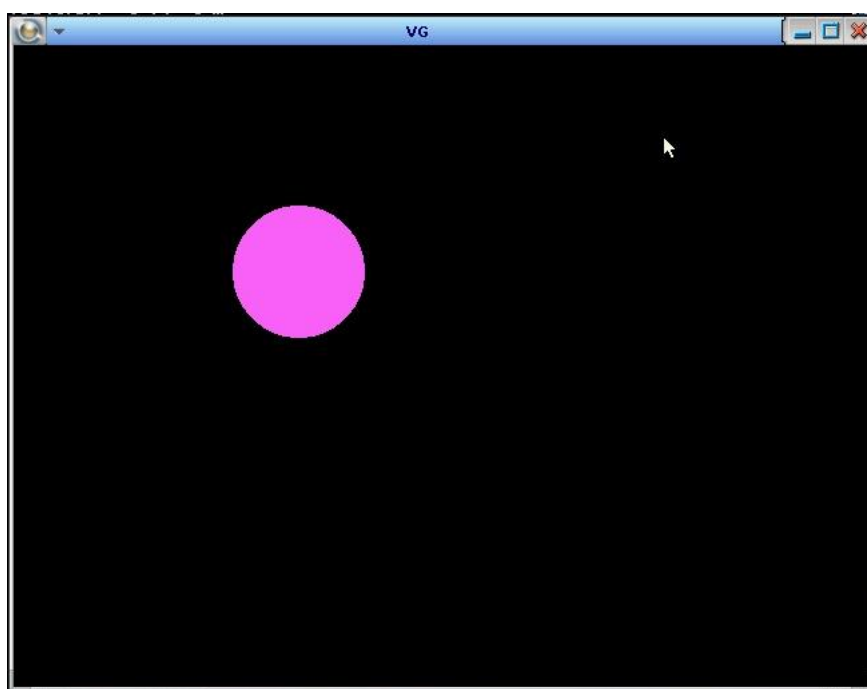


Рисунок 8. Демонстрация работы программы №4(а)

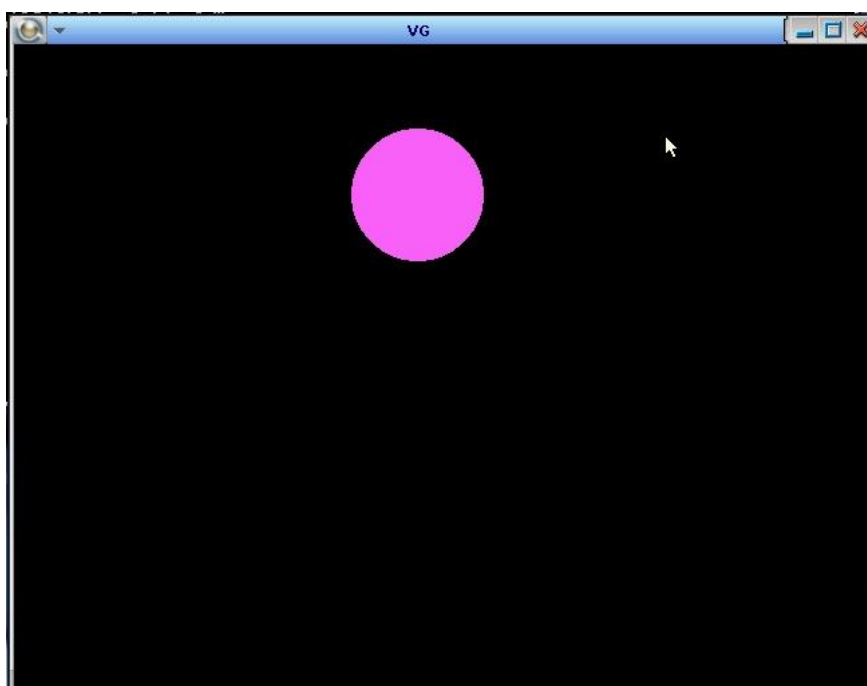
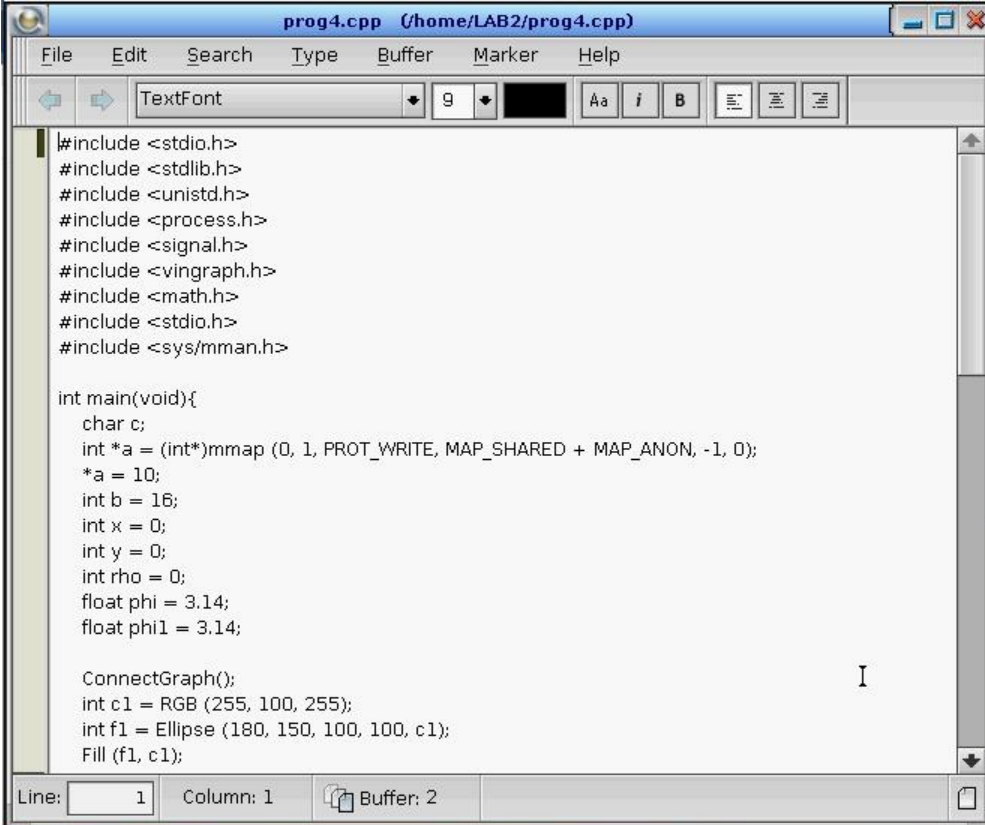


Рисунок 9. Демонстрация работы программы №4(б)

Листинг:



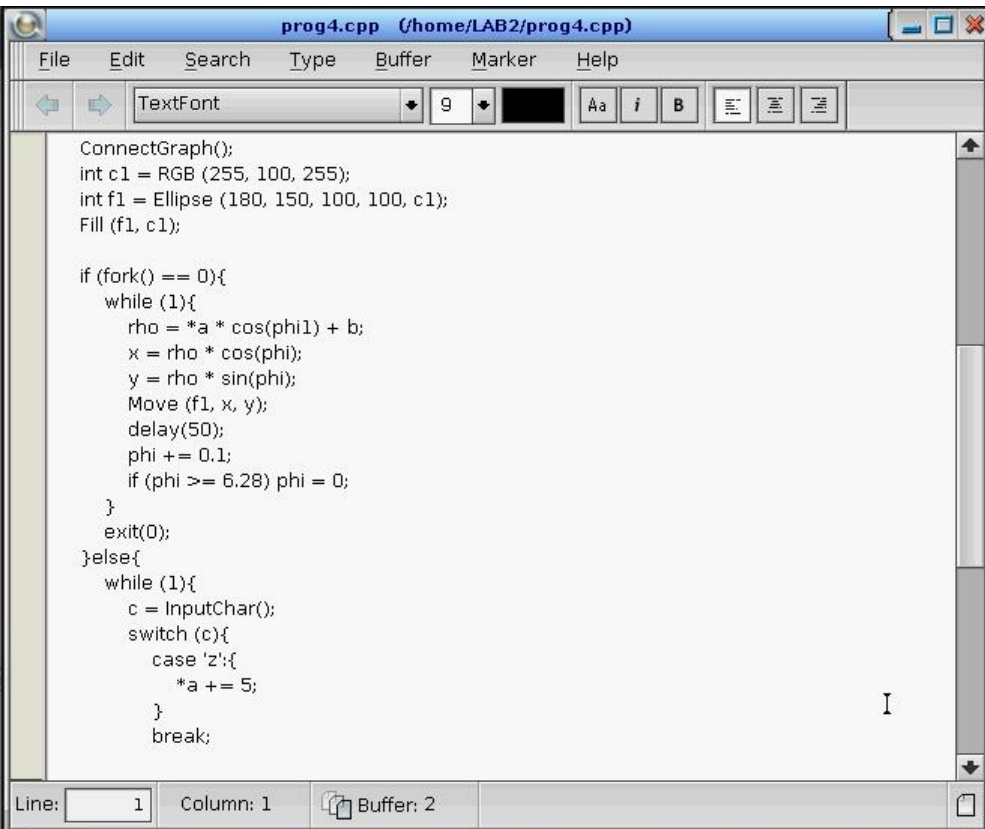
```
prog4.cpp (/home/LAB2/prog4.cpp)
File Edit Search Type Buffer Marker Help
TextFont 9
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <process.h>
#include <signal.h>
#include <vingraph.h>
#include <math.h>
#include <stdio.h>
#include <sys/mman.h>

int main(void){
    char c;
    int *a = (int*)mmap (0, 1, PROT_WRITE, MAP_SHARED + MAP_ANON, -1, 0);
    *a = 10;
    int b = 16;
    int x = 0;
    int y = 0;
    int rho = 0;
    float phi = 3.14;
    float phi1 = 3.14;

    ConnectGraph();
    int c1 = RGB (255, 100, 255);
    int f1 = Ellipse (180, 150, 100, 100, c1);
    Fill (f1, c1);
```

Line: 1 Column: 1 Buffer: 2

Рисунок 10. Текст программы №4(а)

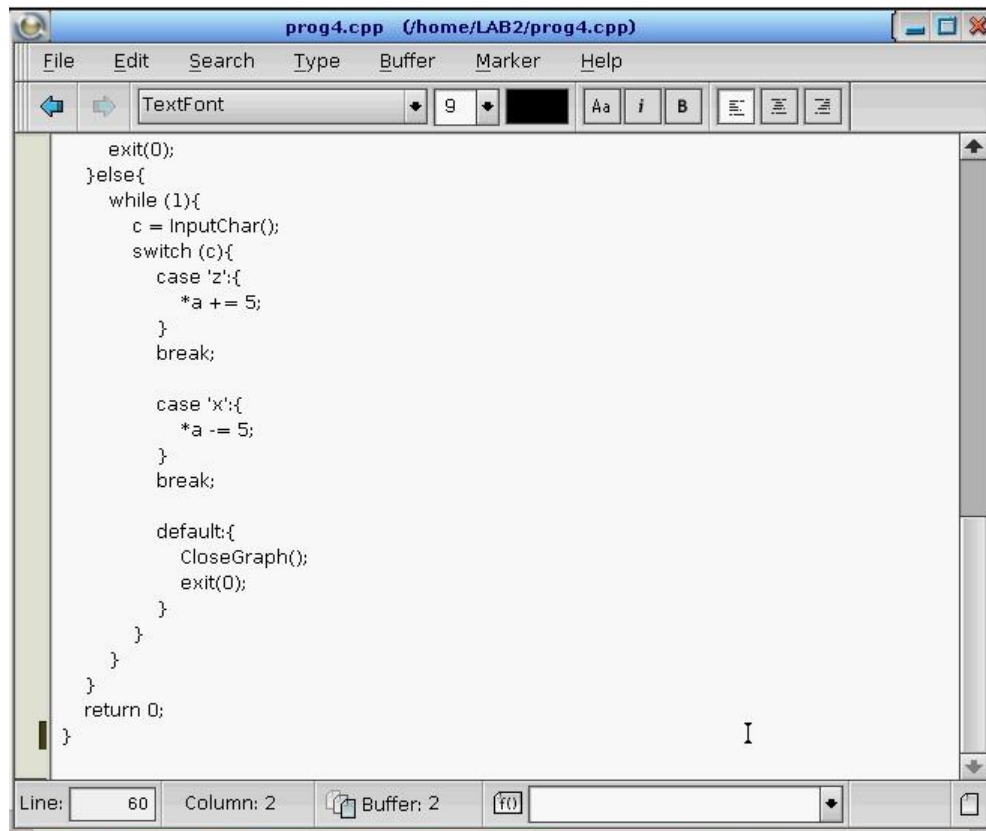


```
prog4.cpp (/home/LAB2/prog4.cpp)
File Edit Search Type Buffer Marker Help
TextFont 9
ConnectGraph();
int c1 = RGB (255, 100, 255);
int f1 = Ellipse (180, 150, 100, 100, c1);
Fill (f1, c1);

if (fork() == 0){
    while (1){
        rho = *a * cos(phi1) + b;
        x = rho * cos(phi);
        y = rho * sin(phi);
        Move (f1, x, y);
        delay(50);
        phi += 0.1;
        if (phi >= 6.28) phi = 0;
    }
    exit(0);
}else{
    while (1){
        c = InputChar();
        switch (c){
            case 'z':{
                *a += 5;
            }
        }
        break;
    }
}
```

Line: 1 Column: 1 Buffer: 2

Рисунок 11. Текст программы №4(б)



The screenshot shows a code editor window titled "prog4.cpp (/home/LAB2/prog4.cpp)". The editor has a menu bar with "File", "Edit", "Search", "Type", "Buffer", "Marker", and "Help". Below the menu is a toolbar with icons for undo, redo, font face (set to "TextFont"), font size (set to "9"), and text color (set to black). The main text area contains the following C++ code:

```
exit(0);
}else{
  while (1){
    c = InputChar();
    switch (c){
      case 'z':{
        *a += 5;
      }
      break;

      case 'x':{
        *a -= 5;
      }
      break;

      default:{
        CloseGraph();
        exit(0);
      }
    }
  }
}
return 0;
}
```

At the bottom of the editor, there is a status bar showing "Line: 60", "Column: 2", and "Buffer: 2".

Рисунок 12. Текст программы №4(в)

6. Лабораторная работа №2 часть 5

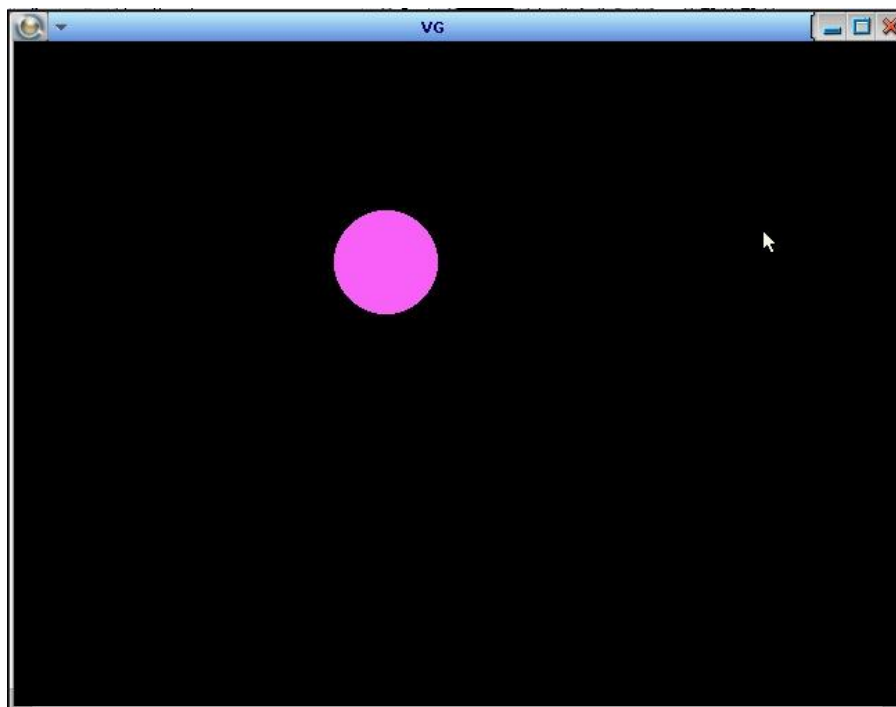
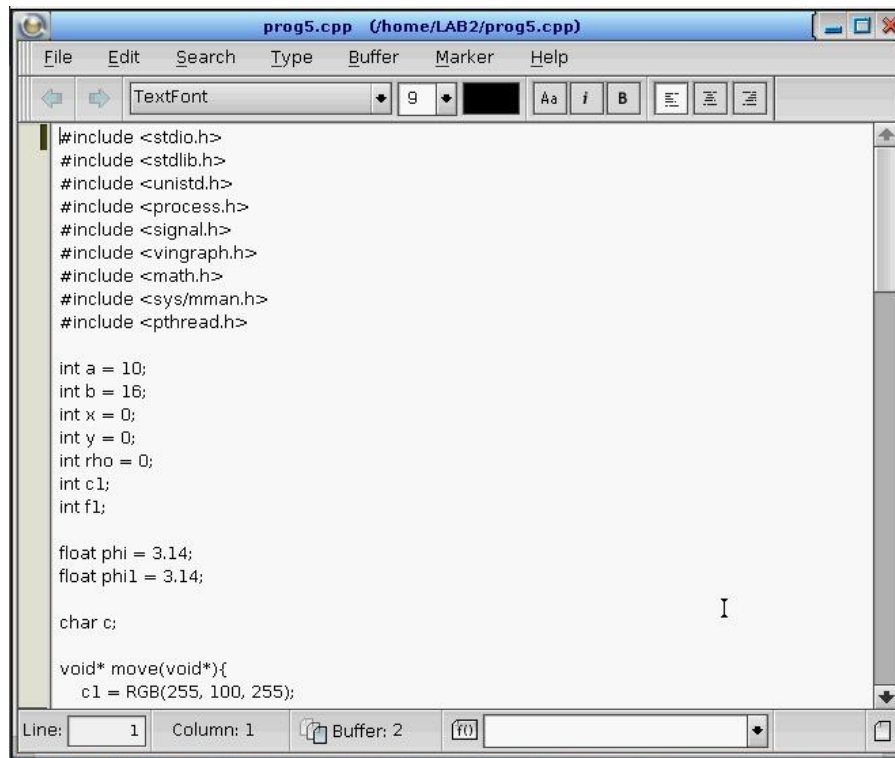


Рисунок 13. Демонстрация работы программы №5

Листинг:



The screenshot shows a text editor window titled "prog5.cpp (/home/LAB2/prog5.cpp)". The menu bar includes File, Edit, Search, Type, Buffer, Marker, and Help. The toolbar shows a font dropdown set to "TextFont", a size dropdown set to "9", and buttons for bold, italic, and underline. The code in the editor is as follows:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <process.h>
#include <signal.h>
#include <vingraph.h>
#include <math.h>
#include <sys/mman.h>
#include <pthread.h>

int a = 10;
int b = 16;
int x = 0;
int y = 0;
int rho = 0;
int c1;
int f1;

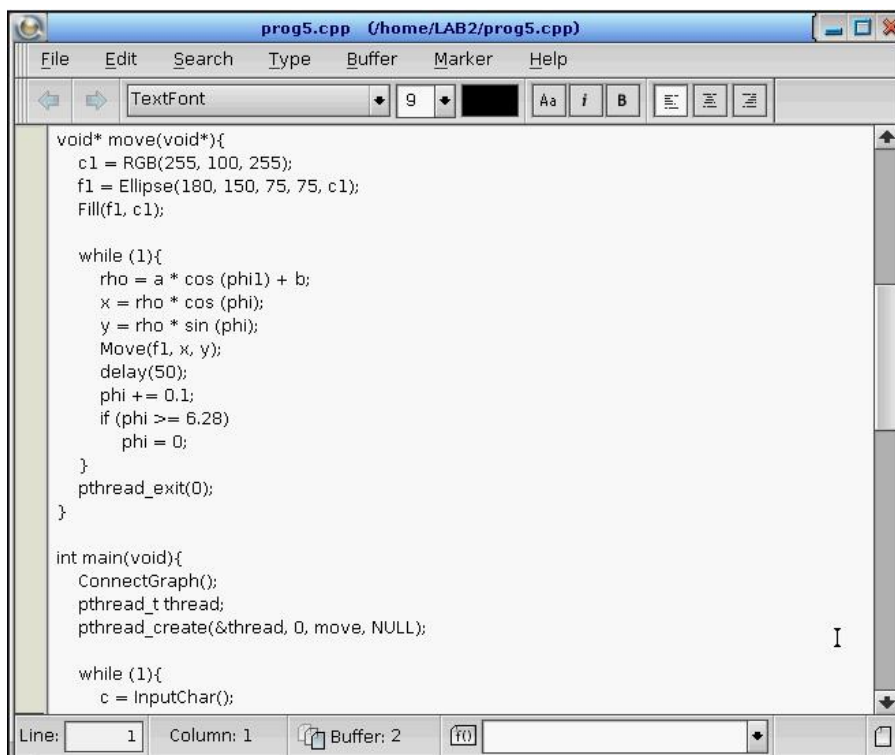
float phi = 3.14;
float phi1 = 3.14;

char c;

void* move(void*){
    c1 = RGB(255, 100, 255);
```

The status bar at the bottom shows "Line: 1", "Column: 1", and "Buffer: 2".

Рисунок 14. Текст программы №5(а)



The screenshot shows the same text editor window as Figure 14, but with more code visible. The code continues as follows:

```
    f1 = Ellipse(180, 150, 75, 75, c1);
    Fill(f1, c1);

    while (1){
        rho = a * cos (phi1) + b;
        x = rho * cos (phi);
        y = rho * sin (phi);
        Move(f1, x, y);
        delay(50);
        phi += 0.1;
        if (phi >= 6.28)
            phi = 0;
    }
    pthread_exit(0);
}

int main(void){
    ConnectGraph();
    pthread_t thread;
    pthread_create(&thread, 0, move, NULL);

    while (1){
        c = InputChar();
```

The status bar at the bottom shows "Line: 1", "Column: 1", and "Buffer: 2".

Рисунок 15. Текст программы №5(б)

```
while (1){  
    c = InputChar();  
    switch (c){  
        case 'z':{  
            a += 5;  
        }  
        break;  
  
        case 'x':{  
            a -= 5;  
        }  
        break;  
  
        default:{  
            CloseGraph();  
            exit(0);  
        }  
    }  
}  
pthread_join(thread, NULL);  
return 0;  
}
```

Рисунок 16. Текст программы №5(в)

