МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра Вычислительной техники

ОТЧЕТ

по лабораторной работе № 3

по дисциплине «Организация ЭВМ и систем»

ТЕМА: ИССЛЕДОВАНИЕ ВИДЕОСИСТЕМЫ (ГРАФИЧЕСКИЙ РЕЖИМ)

Студент гр. 3311 _____ Баймухамедов Р. Р.

Преподаватель _____ Гречухин М. Н.

Санкт-Петербург

2024

Краткие сведения о видеосистемах ПЭВМ, графическом режиме их работы и функциях обслуживания графического режима

Видеосистема персонального компьютера (ПЭВМ) является важным компонентом, обеспечивающим отображение графической и текстовой информации на экране. Графический режим работы видеосистемы позволяет выводить на экран изображения, состоящие из пикселов, что делает возможным отображение сложных графических примитивов и изображений. В данном разделе рассмотрены основные аспекты видеосистем ПЭВМ, графического режима их работы и функций обслуживания графического режима.

Видеосистема ПЭВМ включает в себя видеоадаптер и монитор. Видеоадаптер отвечает за обработку графической информации и передачу её на монитор для отображения. Существует несколько типов видеоадаптеров, таких как CGA, EGA и VGA, каждый из которых поддерживает различные графические режимы и разрешения.

Графический режим работы видеосистемы позволяет отображать на экране изображения, состоящие из пикселов. В этом режиме каждый пиксел на экране может иметь свой собственный цвет, что делает возможным отображение сложных графических примитивов и изображений. Графический режим поддерживается различными видеоадаптерами и может быть настроен для работы с различными разрешениями и цветовыми палитрами.

Для работы с графическим режимом видеосистемы используются специальные функции, предоставляемые библиотекой графики. В языке программирования C++ для работы с графикой используется библиотека graphics.h, которая включает в себя набор функций для инициализации, закрытия и управления графическим режимом.

Перед началом работы с графическим режимом необходимо инициализировать систему графики. Для этого используется функция initgraph(), которая загружает соответствующий драйвер и устанавливает видеоадаптер в графический режим.

После инициализации системы графики можно установить цвета пикселов и палитры.

Графическое окно (viewport) - это прямоугольная область экрана, заданная пиксельными координатами левого верхнего и правого нижнего углов. Для задания окна используется функция setviewport().

Библиотека графики позволяет выводить текст в графическом режиме с использованием различных шрифтов. Для вывода текста используются

функции outtext() и outtextxy().

Библиотека графики предоставляет функции для вывода основных графических примитивов, таких как отрезки прямых линий, окружности, эллипсы, прямоугольники и секторы.

Задание на лабораторную работу

Разработать программу для вывода на экран графика функции $Sin^2(x/4)+Sqrt(x)$ на промежутке от $3\pi/2$ до 17π. Произвести разметку осей и проставить истинные значения точек. Найти максимальное значение функции на заданном интервале и вывести в отдельное окно на экране вместе с графиком.

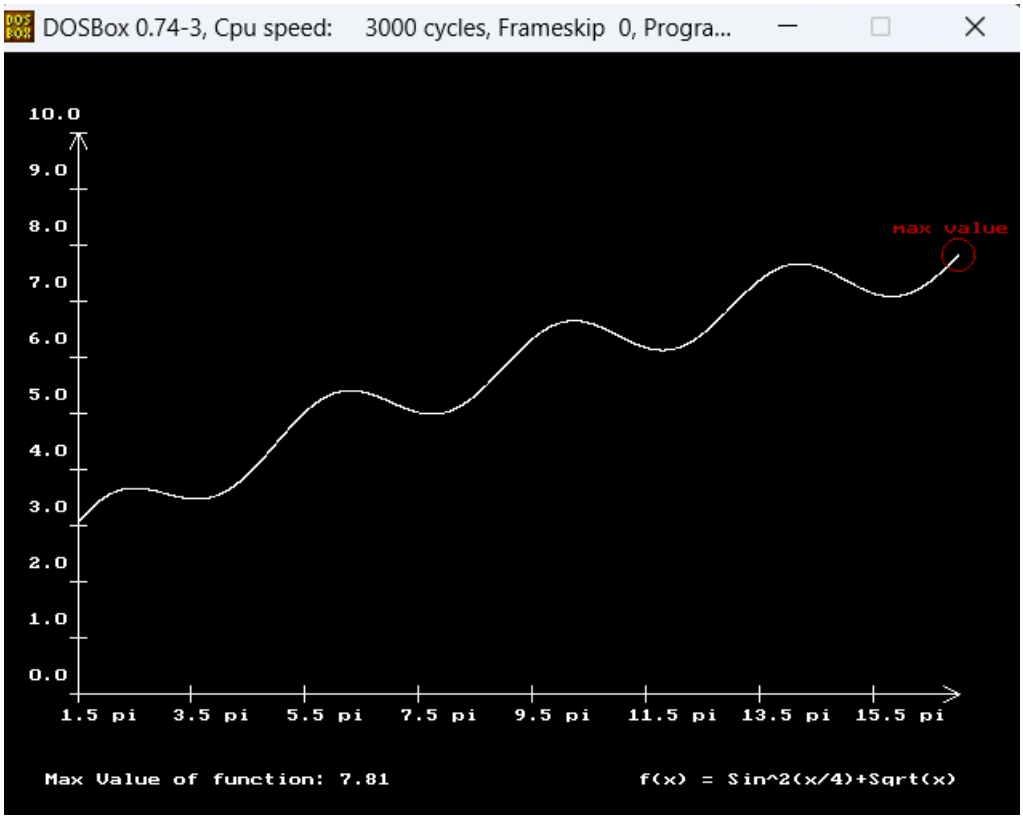
Алгоритмы и тексты отлаженных программ

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define PI M_PI

float func(float x){
    return pow(sin(x/4), 2)+sqrt(x);
    // return pow(sin(x/2), 3) + sqrt(x);
}

void draw_axes_labels(float x_left_border, float x_right_border, float
y_bottom_border, float y_top_border, float x_step, float y_step) {
    char label[50];
    float x, y;
    float screen_x, screen_y;
    float count = 1.5;

    setcolor(WHITE);

    for (x = x_left_border; x <= x_right_border; x += x_step) { // draw labels on
X-axis
        screen_x = 50 + (x-x_left_border)* 550 / (x_right_border -
x_left_border);
        line(screen_x, 405, screen_x, 395); // draw a small line
        sprintf(label, "%.1f pi", count); // convert
        outtextxy(screen_x - 10, 410, label); // draw value of line
        count=count+2;
    }

    for (y = y_bottom_border; y <= y_top_border; y += y_step) { // draw labels on
Y-axis
```

```
        screen_y = 400 - (y - y_bottom_border) * 350 / (y_top_border -
y_bottom_border);
        line(45, screen_y, 55, screen_y); // draw a small line
        sprintf(label, "%.1f", y); // convert
        outtextxy(20, screen_y - 15, label); // draw value of line
    }
}

int main() {
    int gdriver = DETECT, gmode, errorcode; // request auto detection
    float x_left_border = 3*PI/2, x_right_border = 17*PI; // vertical interval
(task)
    float y_bottom_border = 0, y_top_border = 10; // horizontal interval
    float step = 0.01; // step
    float x, y; // variables in loop
    float screen_x, screen_y; // x, y coordinates in loop
    float y_max_value = -100; // max function value
    int x_max_coordinate, y_max_coordinate; // x, y coordinates of max function
value
    char str_max_value[50];

    initgraph(&gdriver, &gmode, "//tc//bgi"); // initialize graphics and this is
unmobile part of libary - driver is listener of system interuptions

    setlinestyle(0, 1, 2); // solid line, upattern useless, thick width
    setcolor(WHITE);
    line (50, 400, 600, 400); // horizontal line x1, y1, x2, y2
    line(50, 400, 50, 50); // vertical line
    line(600, 400, 590, 395); // draw arrow
    line(600, 400, 590, 405);
    line(50, 50, 45, 60); // draw arrow
    line(50, 50, 55, 60);

    draw_axes_labels(x_left_border, x_right_border, y_bottom_border,
y_top_border, 2*PI, 1);

    for(x=x_left_border; x<x_right_border; x+=step){ // x from 3pi/2 to 17pi
        y=func(x); // y = f(x)
        screen_x = 50 + (x - x_left_border) * 550 / (x_right_border -
x_left_border); // '50' is offset to 'oy' // 'x-x_left_border' is change '[xleft,
xright]' to '[0, xright-xleft] // 550 / (x_r_border - x_l_border) is scale (size)
= single step in the interval Δx will be converted to a proportional number of
pixels on the screen
        screen_y = 400 - (y - y_bottom_border) * 350 / (y_top_border -
y_bottom_border); // '400' is offset to 'ox' // 'y-y_bottom_border' is change
'[ybottom, ytop]' to '[0, ytop-ybottom] // 350 / (y_t_border - y_b_border) is
scale (size) = single step in the interval Δx will be converted to a proportional
number of pixels on the screen
        putpixel(screen_x, screen_y, WHITE); // put pixel on screen_x, screen_y
coordinates
```

```
        if(y > y_max_value){
            x_max_coordinate = screen_x;
            y_max_coordinate = screen_y;
            y_max_value = y;
        }
    }

    setcolor(RED);
    circle(x_max_coordinate, y_max_coordinate, 10); // draw a circle in
coordinates with max function value
    outtextxy(x_max_coordinate-40, y_max_coordinate-20, "max value");
    setcolor(WHITE);
    sprintf(str_max_value, "Max Value of function: %.2f", y_max_value);
    outtextxy(30, 450, str_max_value);
    outtextxy(400, 450, "f(x) = Sin^2(x/4)+Sqrt(x)");
    getch(); // wait user action
    closegraph(); // clear and close
    return 0;
}
```

Пример запуска программы



Выводы

Видеосистема ПЭВМ и графический режим её работы являются важными компонентами для отображения графической информации на экране. Библиотека графики в языке программирования C++ предоставляет

широкий набор функций для инициализации, управления и вывода графических примитивов, что делает возможным создание сложных графических приложений.