Министерство науки и образования РФ

Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

им. В.И. Ульянова (Ленина)»

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

**Отчет по лабораторной работе № 3**

по дисциплине «Организация ЭВМ и систем» на

тему: «Исследование видеосистемы (графический режим)»

|  |  |
| --- | --- |
| Выполнил студент гр. 9308 | Хамитов А.К. |
|  |  |
| Проверил | Жандаров В.В. |

Санкт-Петербург, 2020

**Оглавление**

[Задание 3](#_Toc57515587)

[Краткие сведения о видеосистемах ПЭВМ 4](#_Toc57515588)

[Текст программы 36](#_Toc57515596)

[Результаты работы программы 38](#_Toc57515597)

[Вывод 40](#_Toc57515598)

# **Задание**

Разработать программу для вывода на экран графика заданной функции.

**Примечание:**

Начало построения графика начинается с x = 3π/2 и заканчивается x = 18π.

|  |  |  |  |
| --- | --- | --- | --- |
| Номер | Функция | Диапазон аргумента | |
| Начало | Конец |
| 24 | Cos2(x/2 ) +Sqrt (x ) | 3π/2 | 18π |

Таблица 1. Заданная функция и ее область определения

# **Краткие сведения о видеосистемах ПЭВМ**

Использование графики в языке С++ - это многошаговый процесс. Прежде всего необходимо определить тип видеоадаптера. Затем устанавливается подходящий режим его работы и выполняется инициализация графической системы в выбранном режиме. После этого становятся доступными для исполь­зования функции графической библиотеки graphicx.h для построения основных графических примитивов: отрезков прямых линий, окружностей, эллипсов, прямоугольников, секторов, дуг и т.д., появляется возможность вывода текста с использованием различных шрифтов.

Прежде чем использовать функции графической библиотеки С++, необходимо инициализировать систему графики - загрузить соответствующий адаптеру или режиму .BGI-драйвер, установить в начальные значения внешние переменные и константы, выбрать шрифт и т.д.

Инициализацию графической модели выполняет функция initgraph().

Если при выполнении инициализации возникает противоречие между за­прашиваемым режимом и типом видеоадаптера, либо отсутствует достаточный объем свободной оперативной памяти и т.п., функция устанавливает код ошибки во внешней переменной, доступной после вызова функции graphresult(). Кроме того, код ошибки передается в точку вызова в ячейке памяти, на которую указывает graphdriver.

Если функции графической библиотеки больше не нужны прикладной программе, следует вызвать функцию closegraph() "закрытия" графического режима и возвращения к текстовому режиму.

Grapherrormsg возвращает указатель на ASCII-строку символов, содержащую сообщение об ошибке, соответствующее внутреннему коду ошибки errorcode функций графики Turbo С.

void setviewport (int left, int top, int right, int bottom, int clip)

Описывает новое графическое окно с координатами (столбец, строка) левого верхнего угла left, top, координатами правого нижнего угла right, bottom и значением флага усечения clip. В качестве начала текущих координат для функций графического вывода устанавливается левый верхний угол.

Вывод текстовой информации через функции библиотеки графики outtext() и outtextxy().

setlinestyle

Устанавливает стиль "рисования" отрезков прямых линий и графических примитивов.

void putpixel

Определяет, лежит ли пиксел с координатами (х, у) в текущем графическом окне, и, если лежит, выводит на экран пиксел, код цвета которого равен pixelcolor. В противном случае цвет пиксела не изменяется.

void line

Выводит отрезок прямой линии между двумя явно специфицированными точками

# **Текст программы**

#include <stdio.h>

#include <dos.h>

#include <graphics.h>

#include <math.h>

#include <stdlib.h>

#include <conio.h>

#define pi 3.1415926536

#define dx 34 // 34 пикселей = pi

#define dy 50 // 50 пикселей = 1

#define x0 7 // Отступ от краев

#define y0 7 // Отступ от краев

#define Maxx 18 // 18pi - макс значение

#define Minx 1.5 // 1.5pi - мин значение

int main(void)

{

int graph\_driver, // Указатель на используемый .BGI-драйвер

graph\_mode, // Режим видеоадаптера

graph\_error\_code, // Внутренний код ошибки

midx,

midy;

int k = 0, j = 0;

char str[5], str2[16];

double x1, y1, max = 0,

mashtabx = dx/pi,

x, y;

clrscr();

graph\_driver = DETECT; // Автоматически определяем тип драйвера

//Инициализация графической модели

initgraph(&graph\_driver, &graph\_mode, "C:\\TURBOC3\\BGI"); // 3 аргумент задает маршрут поиска файла, содержащего .BGI-драйвер

// Определение кода ошибки при выполнении инициализации

graph\_error\_code = graphresult(); // Возвращает значение внутреннего кода ошибки,

if (graph\_error\_code != grOk) // grOk - Отсутствие ошибки

{

printf("error%s\n", grapherrormsg(graph\_error\_code)); // Указатель на сообщение об ошибке

getch();

return 0;

}

midx = getmaxx(); // Ширина экрана для текущего видеорежима

midy = getmaxy(); // Высота экрана для текущего видеорежима

setviewport(0, 0, midx, midy, 1); // Окно: (0,0) и (midx, midy) - координаты левего верхнего и правого нижнего углов

line(x0, midy, midx, midy); // Ось абцисс

line(x0, 0, x0, midy); // Ось ординат

// for (i = dx+x0; k <= Maxx; i += dx)

// {

// k++;

// sprintf(str, "%dpi", k);

// outtextxy(i - 15, midy - 20, str); // Вывод значений делений на оси Ox

// line(i, midy + 5, i, midy - 5); // Деления на оси Ox

// }

sprintf(str, "1.5pi");

outtextxy(Minx \* dx + x0 - 15, midy - 20, str); // Вывод значений делений на оси Ox

line(Minx \* dx + x0, midy + 5, Minx \* dx + x0, midy - 5); // Деления на оси Ox

sprintf(str, "18pi");

outtextxy(Maxx \* dx + x0 - 15, midy - 20, str); // Вывод значений делений на оси Ox

line(Maxx \* dx + x0, midy + 5, Maxx \* dx + x0, midy - 5); // Деления на оси Ox

k = 9;

for (j = y0 + 28; k >= -1; j += dy)

{

sprintf(str, "%d", k);

outtextxy(0, j - 7, str); // Вывод значений делений на оси Oy

line(x0, j, x0 + 5, j); // Деления на оси Oy

k--;

}

setviewport(0, 0, midx, midy, 0);

x1 = Minx \* pi; // Начало отрисовки графика по оси х

do

{

y1 = cos(x1/2) \* cos(x1/2) + sqrt(x1);

x = x1 \* mashtabx + x0;

y = y1 \* dy;

putpixel(x, midy - y, 5); // Вывод пикселя

if (y1 > max)

max = y1;

x1 += 0.0001;

} while (x <= Maxx \* dx + x0); // Maxx \* pi \* dx/pi = Maxx \* dx (Переводим в нужный масштаб)

sprintf(str2, "Max value is %f", max);

outtextxy(400, 40, str2);

setlinestyle(1, 1, 1);

line(Maxx \* dx + x0, 0, Maxx \* dx + x0, midy); // Границы (1.5pi и 18pi)

line(Minx \* dx + x0, 0, Minx \* dx + x0, midy);

getch();

closegraph(); // Закрытие графического режима и возвращение к текстовому

return 0;

}

# **Результаты работы программы**

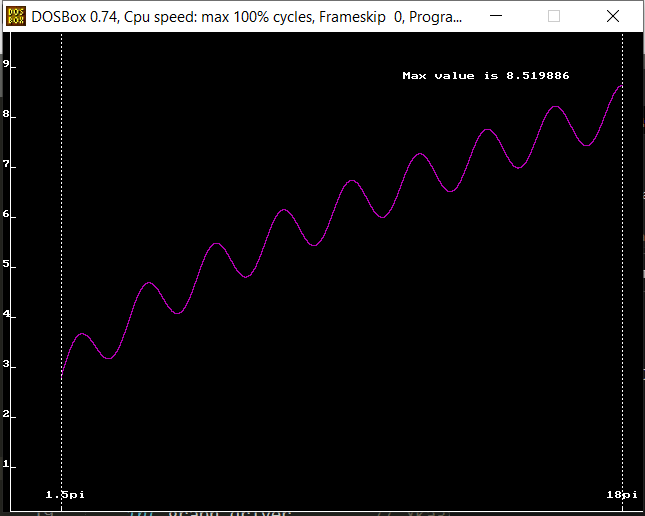


Рисунок 1. Построенный график с выводом максимального значения

# **Вывод**

Была изучена работа с видеосистемой в графическом режиме, включающая построение графика заданной функции с масштабированием и разметкой осей.