Министерство науки и образования РФ

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический

университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»

(СПбГЭТУ «ЛЭТИ»)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Отчет по лабораторной работе №2

по дисциплине “Организация ЭВМ и систем”

на тему: “Исследование видеосистемы (текстовый режим)”

Выполнил: Хамитов А.К. 9308 Принял: Жандаров В.В.

# Содержание

[Содержание 2](#_Toc54708021)

[Задание 3](#_Toc54708022)

[Краткие сведения о видеосистемах ПЭВМ 4](#_Toc54708023)

[Блок-схема алгоритма 7](#_Toc54708024)

[Листинг программы 8](#_Toc54708025)

[Результат работы программы 10](#_Toc54708026)

# Задание

Общее задание:

1. В зависимости от номера варианта задания написать программу, чтобы в окно с координатами (x1, у1, х2, у2) с шагами Т (секунд) и S (строк) выводилась надпись при всех возможных комбинациях цвета фона и цвета символов. Для каждой комбинации цветов в окне должны выводиться номера или символьные обозначения цветов фона и символов

2. Написать и отладить программу на языке С++, реализующую разработанный алгоритм.

Вариант 24(4)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер варианта | Координаты окна | | | | Обозначение цвета | | Шаг | | Направление |
| Xl | Y1 | X2 | Y2 | Фона | Символа | T | S |
| 4 | 25 | 5 | 55 | 15 | Англ. | Англ. | 0.6 | 1 | Вниз |

# Краткие сведения о видеосистемах ПЭВМ

Видеоадаптеры - это весьма сложные устройства, управляемые собственным микропроцессором, сравнимым по мощности с центральным процессором компьютера.

В самом общем виде видеоадаптер состоит из двух основных частей: контроллера и видеопамяти. Основное назначение видеобуфера - хранение образа информации экрана. Видеоадаптер 25 и более раз в секунду формирует изображение на экране. Так как человеческий глаз не способен уловить такое быстрое мелькание кадров, создается иллюзия

неподвижного изображения на экране монитора. Изображение на экране строится

из небольших точек - так называемых пикселов.

Интегральной характеристикой особенностей работы адаптера является совокупность поддерживаемых им режимов. Поведение адаптера в том или ином режиме является фактическим стандартом и полностью характеризует все особенности адаптера, доступные для программиста средства управления адаптером и т.п. Режимы принято нумеровать, начиная с нуля. Чем совершеннее

видеоадаптер, тем больше режимов он поддерживает. Как правило, более совершенные адаптеры полностью совместимы со своими предшественниками и с точки зрения прикладной программы отображает информацию точно так же, как и его предшественник.

Режимы работы видеоадаптеров можно объединить в две группы: текстовые и графические. Переключение из текстового режима в графический и наоборот означает полное изменение логики работы видеоадаптера с видеобуфером. Если видеоадаптер включен в текстовый режим, он рассматривает экран как совокупность текселов.

Каждому знакоместу экрана (текселу) в текстовом режиме соответствуют два байта памяти видеобуфера. Один байт хранит ASCII-код символа, а следующий за ним байт кодирует особенности отображения символа на экране: цвет пикселов, из которых формируется очертание символа, цвет всех остальных пикселов знакоместа или цвет фона символа, мерцание символа и необходимость повышения яркости символа при отображении. Этот байт называется байтом атрибута.

Функции консольного ввода-вывода предназначены для облегчения работы по созданию простейшего оконного интерфейса. Эти функции используют

понятие активного окна экрана. Активное окно - это прямоугольная область экрана, в границах которой в данный момент работают функции. Описание активного окна (или, как часто говорят, фрейм)

хранится во внутренней структурной переменной С++. Установку параметров

активного текстового окна выполняет функция window(int , int ,int , int ) Она описывает активное текстовое окно: первая пара аргументов задает соответственно номера столбца и строки левого верхнего угла, вторая пара - правого нижнего угла.

Видеоадаптеры всех типов аппаратно поддерживают курсор, который в текстовых режимах отображается на экране в виде одной или нескольких линий в пределах тексела. Курсор указывает на текущую позицию экрана (строку и столбец тексела), в которую будет записываться или из которой будет читаться средствами BIOS символ. При переключении адаптера в графический режим курсор становится невидимым, но BIOS сохраняет возможность изменять его позицию. Специальные регистры видеоконтроллера храняттекущую позицию и форму курсора. Управление формой курсора находит ограниченное применение в практике программирования. Однако изменением формы курсора можно отображать различные режимы работы программы, например режим "Вставка" или режим. Среди функций консольного ввода-вывода С++ текущей позицией курсора в окне управляет функция gotoxy(int x, int y);. Устанавливает курсор в заданную строку y и столбец x в текущем активном окне экрана. Верхний левый угол окна имеет координаты (1,1). При попытке позиционировать курсор за границы окна он останавливается на границе окна. Особенностью функции является то, что координаты задаются относитель нулевого верхнего угла текущего окна. Текущую позицию x и y курсора в активном текстовом окне можно узнать при вызове соответственно функций wherex () и wherey().

С++ включает большой набор функций ввода-вывода информации в окно экрана. Прототипы этих функций помещены в заголовочном файле <conio.h>. В отличие от функций стандартного ввода-вывода они позволяют управлять цветом выводимых символов и не пересекают пределы активного в данный момент окна. При достижении правой вертикальной границы курсор автоматически переходитьна начало следующей строки в пределах окна, а при достижении нижней горизонтальной границы выполняется скроллинг окна вверх. Функция clreol() стирает в текстовом окне строку, на которую установлен курсор, начиная с текущей позиции курсора и до конца строки (до правой вертикальной границы окна). Функция clrscr() очищает все текстовое окно. Цвет "заливки" окна при очистке будет соответствовать значению, установленному символической переменной attribute в описании окна (структурная переменная по шаблону text\_info). Функции управления цветом фона и символа описаны далее. Функция delline() стирает в текстовом окне всю строку текста, на которую установлен курсор.

Функция insline() вставляет пустую строку в текущей позиции курсора со сдвигом всех остальных строк окна на одну строку вниз. При этом самая нижняя строка текста окна теряется. Функция cprintf( const char \*format,...) выполняет вывод информации с преобразованием по заданной форматной строке, на которую указывает format. Является аналогом функции стандартной библиотеки printf (), но выполняет вывод в пределах заданного окна. В отличие от printf () функция cprintf () иначе реагирует на специальный символ '\n': курсор переводится на новую строку, но не возвращается к левой границе окна. Поэтому для перевода курсора на начало новой строки текстового окна следует вывести последовательность символов CR-LF (0x0d, 0x0a). Остальные специальные символы воздействуют на курсор так же, как и в случае функций стандартного ввода-вывода. Функция возвращает число выведенных байтов, а не число обработанных полей, как это делает функция printf Функция putch(int ch) выводит символ в текущей позиции текстового окна экрана. Как и для функций cprintf(), cputs(), специальный символ ‘\n' вызывает только переход курсора на следующую строку текстового окна без возврата к его левой вертикальной границе. Остальные специальные символы воздействуют на курсор так же, как и для функций стандартного ввода-вывода.

# Блок-схема алгоритма

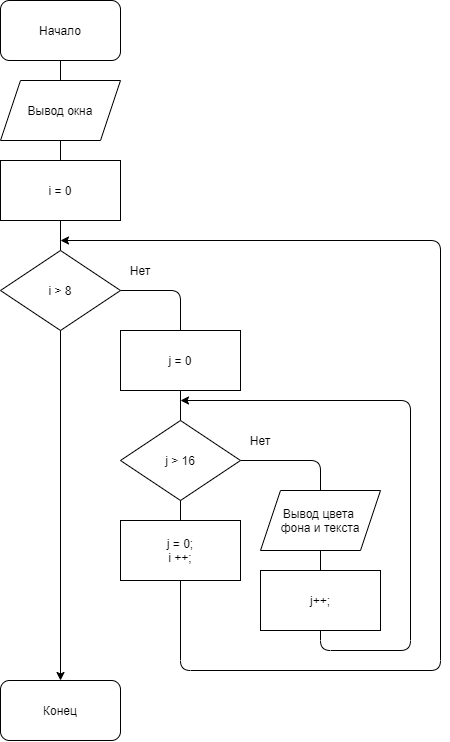


Рисунок 1. Блок-схема алгоритма main.

# Листинг программы

#include <conio.h>

#include <dos.h>

const char\* const colors[]= {

"BLACK",

"BLUE",

"GREEN",

"CYAN",

"RED",

"MAGENTA",

"BROWN",

"LIGHTGRAY",

"DARKGRAY",

"LIGHTBLUE",

"LIGHTGREEN",

"LIGHTCYAN",

"LIGHTRED",

"LIGHTMAGENTA",

"YELLOW",

"WHITE"

};

void main()

{

int i, j;

short T = 600;

int x1 = 25, y1 = 5,

x2 = 55, y2 = 15;

union REGS regs;

regs.h.ah = 7;

regs.h.al = (unsigned char) 1;

regs.h.cl = (unsigned char) (x1-1);

regs.h.ch = (unsigned char) (y1-1);

regs.h.dl = (unsigned char) (x2-1);

regs.h.dh = (unsigned char) (y2-1);

clrscr();

window(x1, y1, x2, y2);

gotoxy(1, 1);

for (j = 0; j < 8; j++)

{

textbackground(j);

for (i = 0; i < 16; i++)

{

textcolor(i);

cprintf("S: %s, B: %s", colors[i], colors[j]);

delay(T);

int86(0x10, & regs, & regs);

gotoxy(1, 1);

}

}

delay(T);

}

# Результат работы программы



Рисунок 3 Пример запуска программы



Рисунок 4 Пример запуска программы