МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет: Информационных технологий

Кафедра: Программной инженерии

Выполнила: студентка 1 курса 5 группы

специальности ПОИТ Дзивнель М.А.

Проверил: преподаватель

Белодед Николай Иванович

**РЕФЕРАТ**

По дисциплине “Основы алгоритмизации и программирования”

На тему “Графика в С++. WinAPI”

Минск

2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ ……………………………………………………………………….2

Подключение……………………………………………………………………...2

Модели цвета………………………………………………………………………2

Настройки курсора. Заголовок консоли………………………………………….3

Графика в консоли. Вывод символов…………………………………………….4

Примеры использования………………………………………………………….5

ВЫВОД …………………………………………………………………………..10

**ВВЕДЕНИЕ**

WinAPI – это набор функций (API – Application Programming Interface), работающих под управлением ОС Windows. Предоставляет прямой способ взаимодействия приложений пользователя с ОС Windows.

При помощи WinAPI можно писать консольные, оконные приложения. Ниже будет подробнее рассмотрена работа с консольными приложениями

**Подключение**

Для работы с WinAPI необходимо подключить заголовочный файл *windows.h* .

Для работы с консолью сначала необходимо получить ее дескриптор. *Дескриптор (описатель) объекта* — служебная структура данных, представляющая собой беззнаковое целое число и служащая для идентификации различных объектов. Дескриптор представляет собой указатель на некоторую системную структуру или индекс в некоторой системной таблице. Получить дескриптор можно функцией GetStdHandle(). Как параметр функция принимает значение константного параметра (ввод, вывод, ошибка). HANDLE – тип данных, используемых в качестве дескриптора.

Пример получения дескрипторов:

HANDLE hInput = GetStdHandle(STD\_INPUT\_HANDLE);

HANDLE hOutput = GetStdHandle(STD\_OUTPUT\_HANDLE);

HANDLE hError = GetStdHandle(STD\_ERROR\_HANDLE);

С помощью WinAPI можно подключать кодовые страницы (использование других языков, по типу русского, китайского и т.д.). Для этого используются функции SetConsoleCP(1251) – для ввода, SetConsoleOutputCP(1251) – для вывода. Где 1251 – номер кодовой страницы для русского языка.

**Модели цвета**

Функции WinAPI позволяют устанавливать произвольные цвета текста и фона ячеек в консоли. Цвета представляются по схеме (I)RGB, где I – интенсивность, R-красный, G-зелёный, B-синий.

Функция SetConsoleTextAttribute() устанавливает атрибуты цвета текста (цвет букв и цвет фона ячеек с буквами). Функция принимает два аргумента. Первый – дескриптор консоли HANDLE, второй – обозначение цвета текста и фона консоли. Чаще всего значения цветов передают при помощи набора констант, так называемых флагов: FOREGROUND\_ для обозначения цвета текста , BACKGROUND\_ для цвета фона. К примеру черный текст на белом фоне мы можем получить при помощи следующей записи:

SetConsoleTextAttribute(hOutput, BACKGROUND\_GREEN | BACKGROUND\_RED | BACKGROUND\_BLUE | BACKGROUND\_INTENSITY);

cout << "черный текст на белом фоне" << endl;

Цвет можно задавать и с помощью чисел, двоичная запись которых показывает, какие цвета мы «включаем», а какие нет (по схеме (I)RGB) . Первые четыре цифры отвечают за цвет фона, последние – за цвет текста. Например число 240 в двоичном коде = 11110000, 1111 означает белый цвет фона, а 0000 черный текст.

SetConsoleTextAttribute(hOutput, 240);

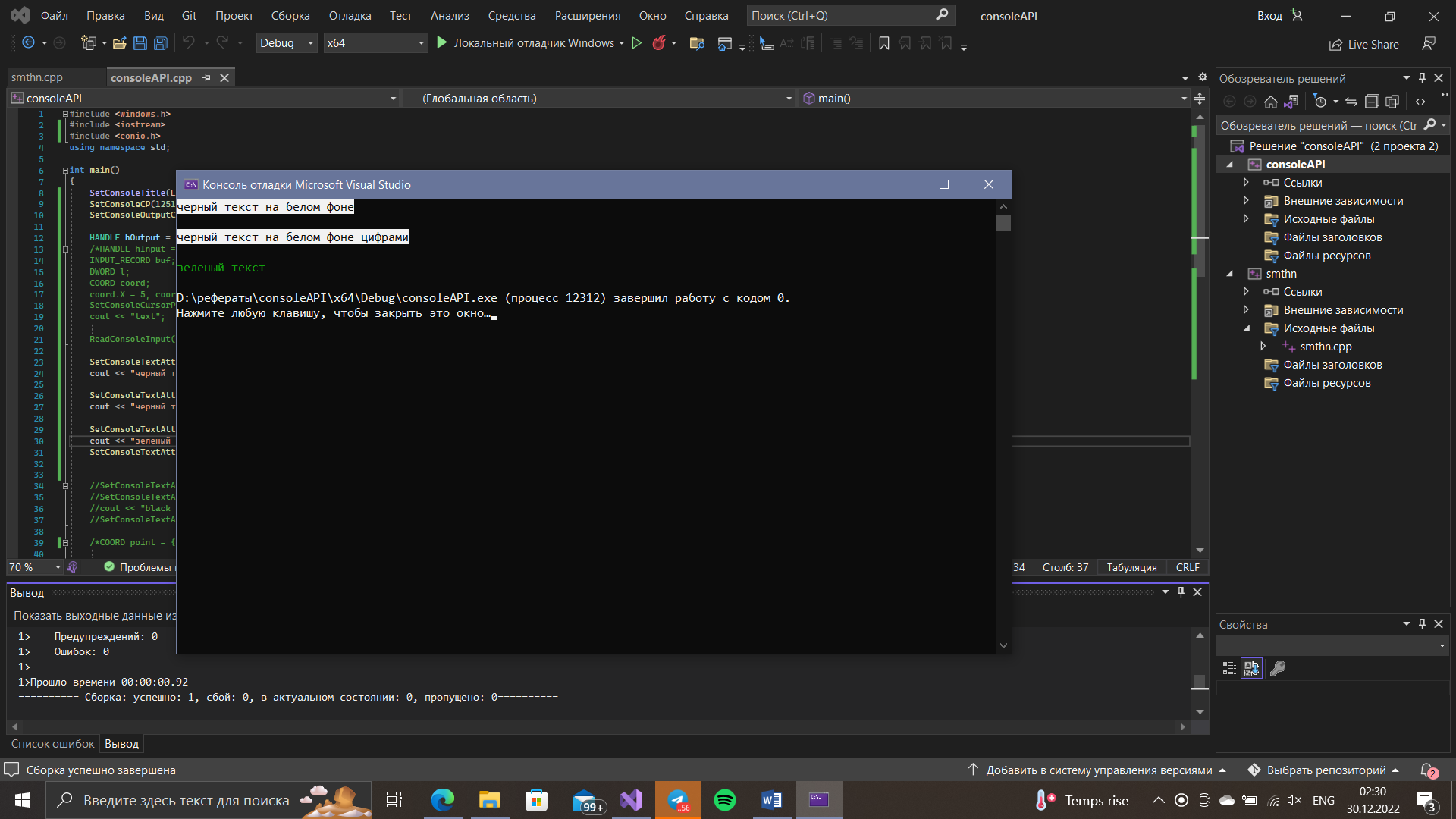
cout << "черный текст на белом фоне цифрами" << endl << endl;

Или число 2, равное 0000 0010. Первые нули дают черный фон, а вот цвет текста будет зеленый, т.к. «включенный» бит стоит на месте G, то есть зеленого цвета.

SetConsoleTextAttribute(hOutput, 2);

cout << "зеленый текст" << endl;

Результат выполнения:



**Настройки курсора. Заголовок консоли**

Функция SetConsoleCursorPosition() устанавливает курсор в указанные координаты в консоли. Функция принимает два аргумента: первый – дескриптор окна, второй – координаты COORD (структура с двумя полями X Y):  
 COORD coord;

coord.X = 5, coord.Y = 2;

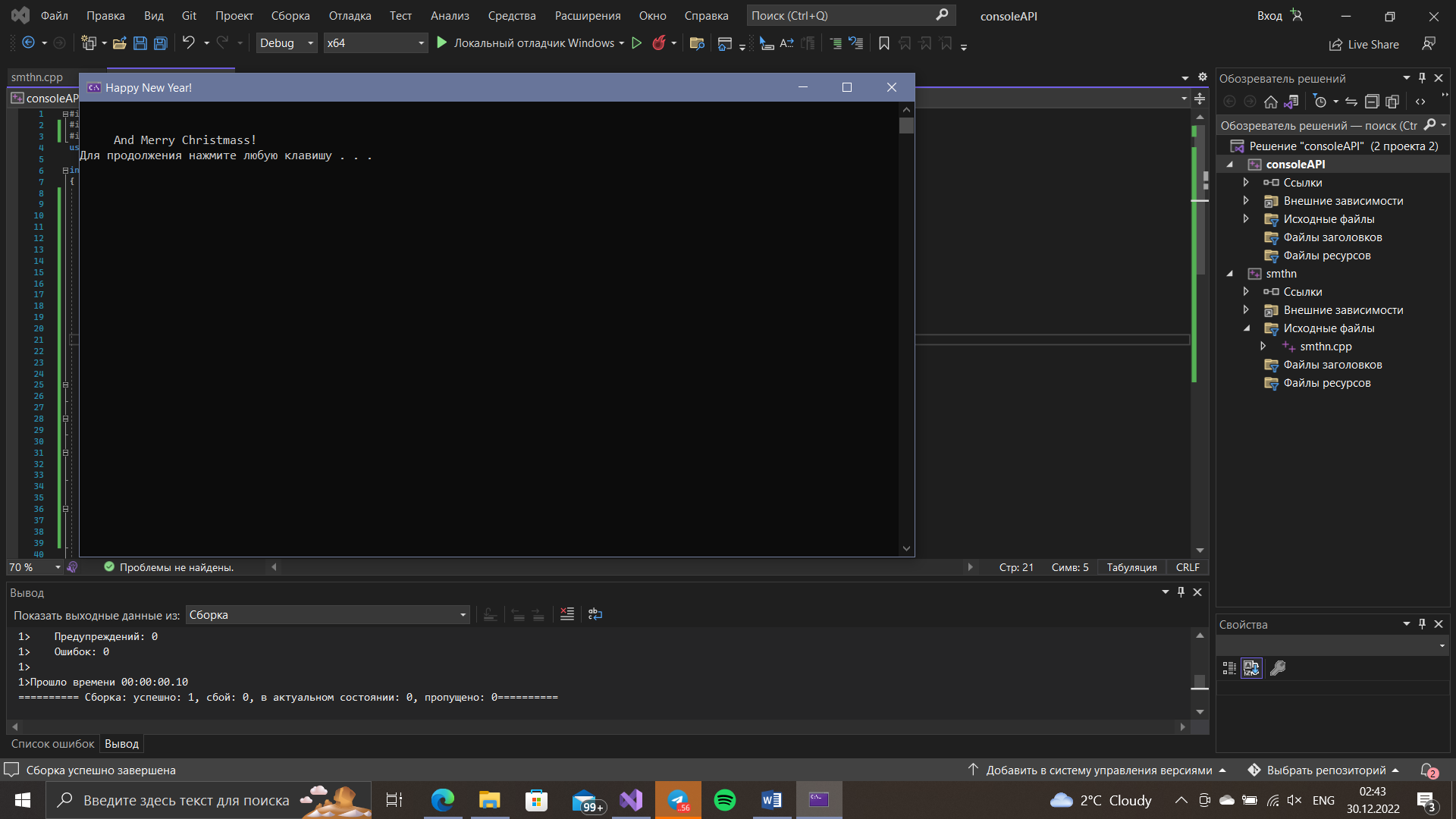
SetConsoleCursorPosition(hOutput, coord);

cout << "And Merry Christmass" << endl;

Функция SetConsoleTitle() устанавливает заголовок консольного окна. В качестве аргумента принимает широкую строку :

SetConsoleTitle(L"Happy New Year!");

Результат выполнения:



**Графика в консоли. Вывод символов**

Функция FillConsoleOutputCharacter() многократно печатает заданный символ в консоли, начиная с указанной позиции. В качестве аргумента принимает:

HANDLE дескриптор консоли

TCHAR символ, который будет выводиться

DWORD количество символов

COORD координаты начала вывода

LPDWORD указатель на переменную, в которой будет количество действительно напечатанных элементов.

COORD coord;

coord.X = 5, coord.Y = 2;

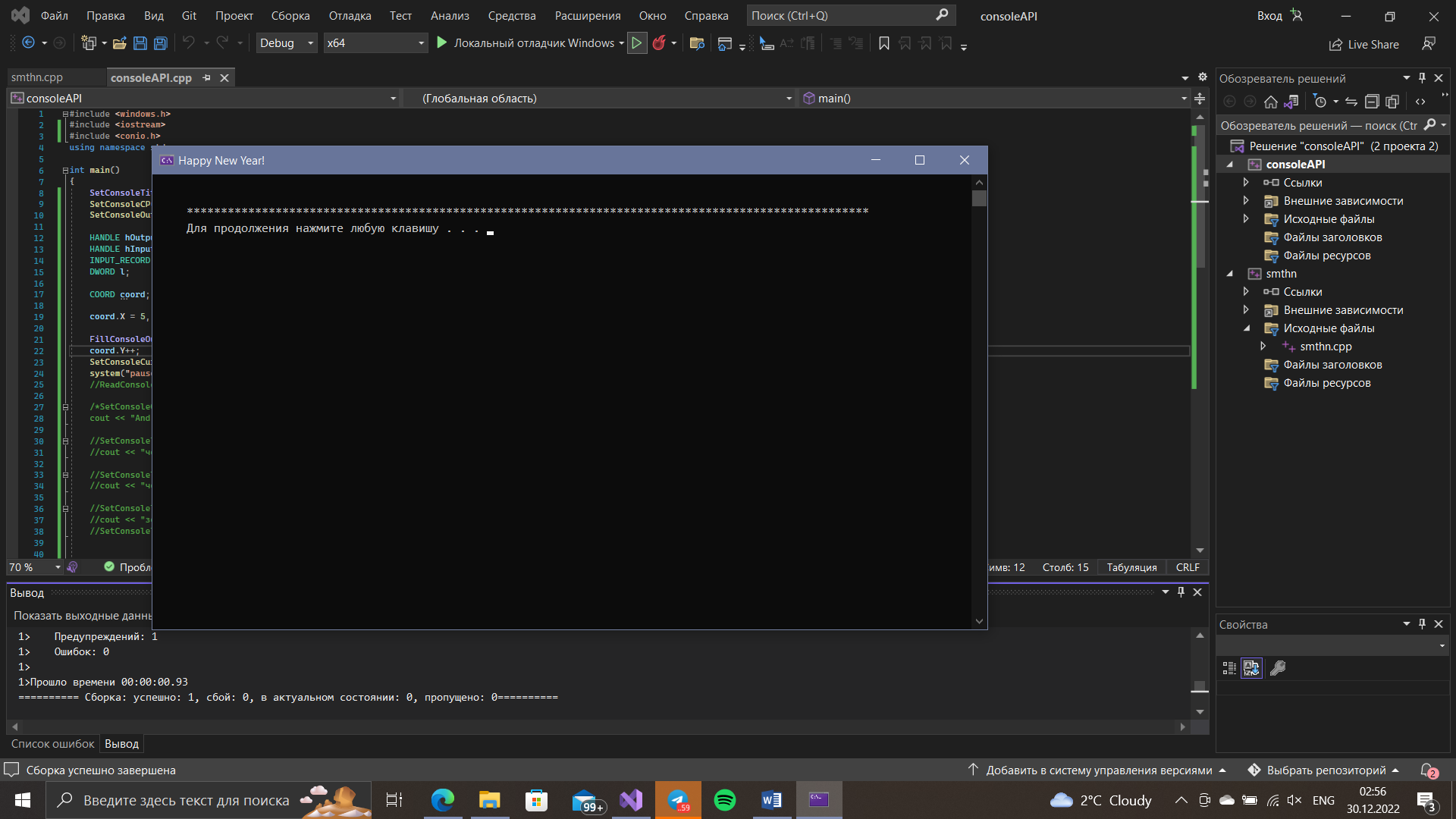
FillConsoleOutputCharacter(hOutput, (WCHAR)'\*', 100, coord, &l);

coord.Y++;

SetConsoleCursorPosition(hOutput, coord);

system("pause");

Пример выполнения кода:



В дополнение к этой функции есть функция FillConsoleOutputAttribute(), которая устанавливает атрибуты цвета для заданного числа ячеек. В качестве аргументов принимает:

HANDLE дескриптор консоли

WORD значения атрибута цветов

DWORD количество символов

COORD координаты начала вывода

LPDWORD указатель на переменную, в которой будет количество действительно заполненных ячеек.

COORD coord = { 5,2 };

WORD perple = BACKGROUND\_BLUE|BACKGROUND\_RED| BACKGROUND\_INTENSITY;

int num = 8;

for (int i = 0; i < num; i++)

{

FillConsoleOutputAttribute(hOutput, perple, num, coord, &l);

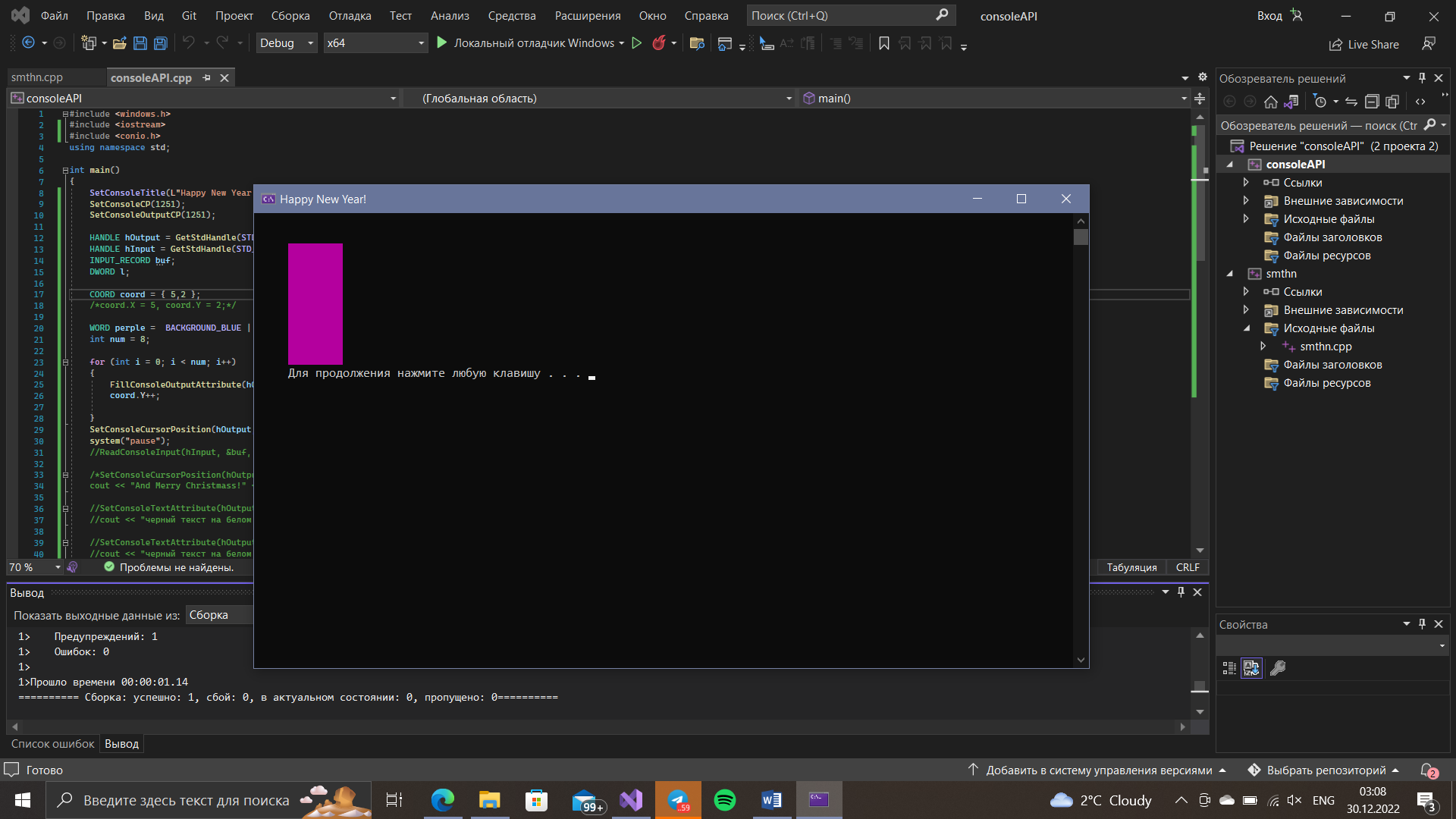
coord.Y++;

}

SetConsoleCursorPosition(hOutput, coord);

system("pause");

Пример выполнения кода:



**Пример использования**

Функции WinApi для консоли можно использовать для множества задач. В основном это работа с текстом, т.к. консоль это текстовый интерфейс. Однако можно делать и небольшие графические программы, по типу графиков, диаграмм и так далее. Ниже следует код программы, выводящей диаграмму, построенную по введенным с клавиатуры числам:

#include <windows.h>

#include <iostream>

using namespace std;

const int WIDTH = 100, HEIGHT = 27;

HANDLE hOutput = GetStdHandle(STD\_OUTPUT\_HANDLE);

HANDLE hInput = GetStdHandle(STD\_INPUT\_HANDLE);

DWORD l;

COORD coord;

WORD colorBl = BACKGROUND\_BLUE | BACKGROUND\_RED;

WORD white = BACKGROUND\_GREEN | BACKGROUND\_RED | BACKGROUND\_BLUE | BACKGROUND\_INTENSITY;

void DrawAxes()

{

coord = { 2,1 };

for (int i = 0; i < HEIGHT; i++)

{

FillConsoleOutputCharacter(hOutput, (WCHAR)'|', 1, coord, &l);

coord.Y++;

}

coord = { 2, HEIGHT };

for (int i = 0; i < WIDTH; i++)

{

FillConsoleOutputCharacter(hOutput, (WCHAR)'\_', 1, coord, &l);

coord.X++;

}

coord.X = 2;

for (int i = 0; i < HEIGHT; i += 9)

{

coord.Y = i;

FillConsoleOutputCharacter(hOutput, (WCHAR)'\_', WIDTH, coord, &l);

SetConsoleTextAttribute(hOutput, FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_RED);

}

}

void DrawBlocks(int height1, int height2, int height3)

{

coord = { 10, HEIGHT };

for (int i = 0; i < height1; i++)

{

FillConsoleOutputAttribute(hOutput,colorBl , 10, coord, &l);

coord.Y--;

}

coord = { 40, HEIGHT };

for (int i = 0; i < height2; i++)

{

FillConsoleOutputAttribute(hOutput, colorBl, 10, coord, &l);

coord.Y--;

}

coord = { 70, HEIGHT };

for (int i = 0; i < height3; i++)

{

FillConsoleOutputAttribute(hOutput, colorBl, 10, coord, &l);

coord.Y--;

}

}

void PrintNames(int height1, int height2, int height3, int \*arr\_graph, int section)

{

//section name

SetConsoleTextAttribute(hOutput,white);

coord = { 2,0 };

SetConsoleCursorPosition(hOutput, coord);

cout << section \* 3;

coord = { 2, 9 };

SetConsoleCursorPosition(hOutput, coord);

cout << section \* 2;

coord = { 2, 18 };

SetConsoleCursorPosition(hOutput, coord);

cout << section;

//BLOCKS name

SetConsoleTextAttribute(hOutput, white);

coord = { 15,(short)(HEIGHT - height1) };

SetConsoleCursorPosition(hOutput, coord);

cout << arr\_graph[0];

coord = { 45, (short)(HEIGHT - height2) };

SetConsoleCursorPosition(hOutput, coord);

cout << arr\_graph[1];

coord = { 75, (short)(HEIGHT - height3) };

SetConsoleCursorPosition(hOutput, coord);

cout << arr\_graph[2];

}

int main()

{

SetConsoleTitle(L"Graphics");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

SetConsoleTextAttribute(hOutput, white);

int size = 3;

int \*arr\_graph = (int\*)malloc(size \* sizeof(int));

cout << "enter numbers for graphic: " << endl;

for (int i = 0; i < size; i++)

{

cin >> arr\_graph[i];

}

system("cls");

int max = arr\_graph[0];

for (int i = 0; i < size; i++)

{

if (arr\_graph[i] > max)

{

max = arr\_graph[i];

}

}

int countNum = -1;

int num = max;

while (num > 0)

{

num /= 10;

countNum++;

}

if (max == 4)

max++;

int maxG = 0;

if (max % 10 != 0)

{

maxG = floor(max / pow(10, countNum) + 1) \* pow(10, countNum);

}

else

maxG = max+2;

int section = maxG / 3;

double scale = static\_cast<double>(9) / section;

int height1 = arr\_graph[0] \* scale;

int height2 = arr\_graph[1] \* scale;

int height3 = arr\_graph[2] \* scale;

DrawAxes();

DrawBlocks(height1, height2, height3);

PrintNames(height1, height2, height3, arr\_graph, section);

coord = { 0, HEIGHT+2 };

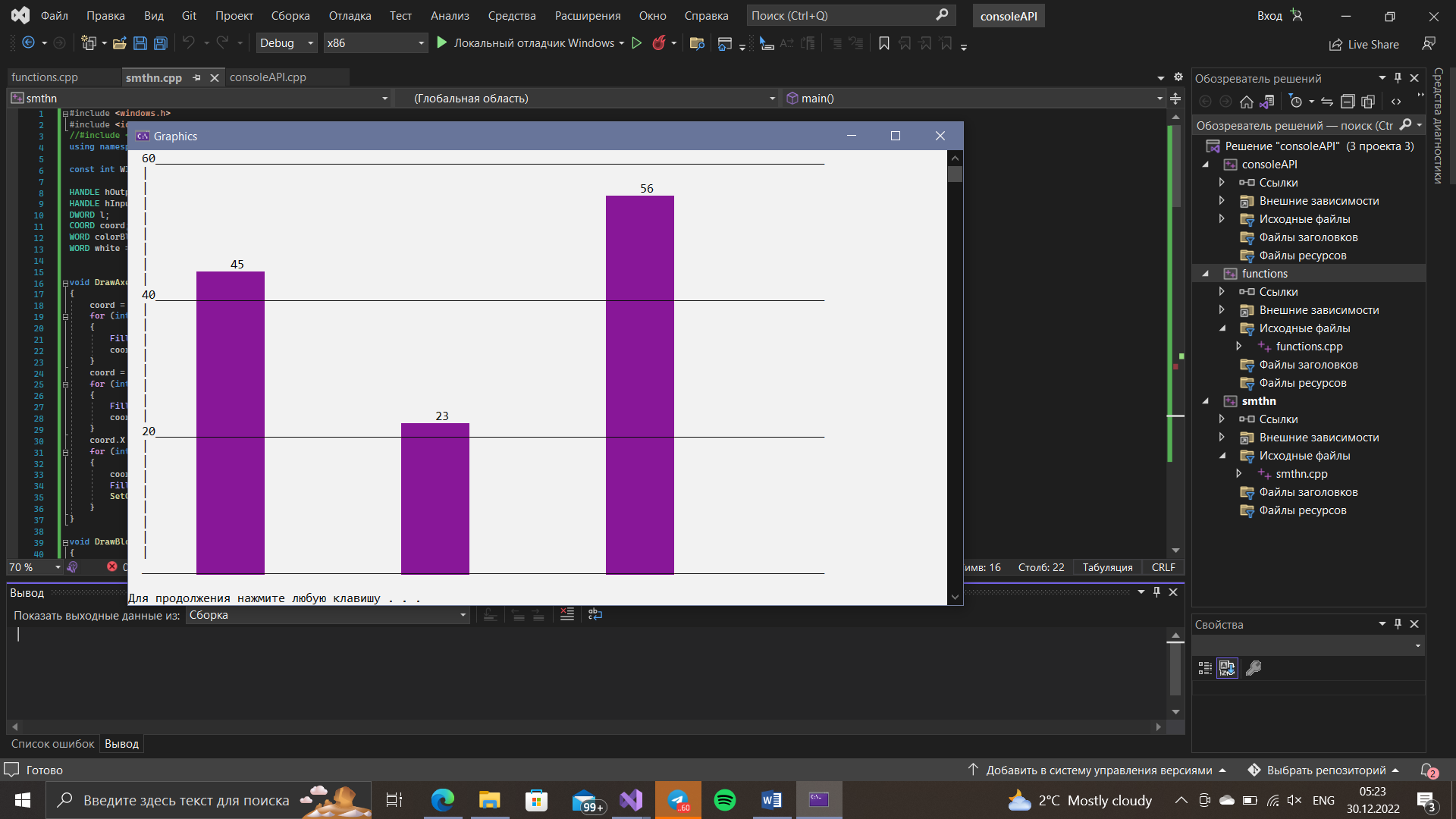
SetConsoleCursorPosition(hOutput, coord);

system("pause");

free(arr\_graph);

}

Результат выполнения программы с введенными значениями 45, 23, 56:



Также можно выводить графики функций. Пример построения графика простейшей линейной функции с введенными k=1, b=3:

#include <windows.h>

#include <iostream>

using namespace std;

const int WIDTH = 70, HEIGHT = 50;

HANDLE hOutput = GetStdHandle(STD\_OUTPUT\_HANDLE);

HANDLE hInput = GetStdHandle(STD\_INPUT\_HANDLE);

COORD coord;

DWORD l;

void DrawAxes()

{

coord = { 35,1 };

for (int i = 0; i < HEIGHT; i++)

{

FillConsoleOutputCharacter(hOutput, (WCHAR)'|', 1, coord, &l);

coord.Y++;

}

coord = { 1, 25 };

for (int i = 0; i < WIDTH; i++)

{

FillConsoleOutputCharacter(hOutput, (WCHAR)'\_', 1, coord, &l);

coord.X++;

}

}

int main()

{

int k , b;

cout << "enter k and b:" << endl;

cin >> k >> b;

system("cls");

int x = 0, y = 0;

DrawAxes();

coord = { 80, 20 };

SetConsoleCursorPosition(hOutput, coord);

cout << "k=" << k << " b=" << b;

for (int x = 0; x < WIDTH; x++)

{

y = k\*x + b;

coord = { (short)x, (short)y };

FillConsoleOutputCharacter(hOutput, (WCHAR)'.', 1, coord, &l);

}

for (int x = WIDTH; x > 35; x--)

{

y = k\*x + b;

coord = { (short)x, (short)y };

FillConsoleOutputCharacter(hOutput, (WCHAR)'.', 1, coord, &l);

}

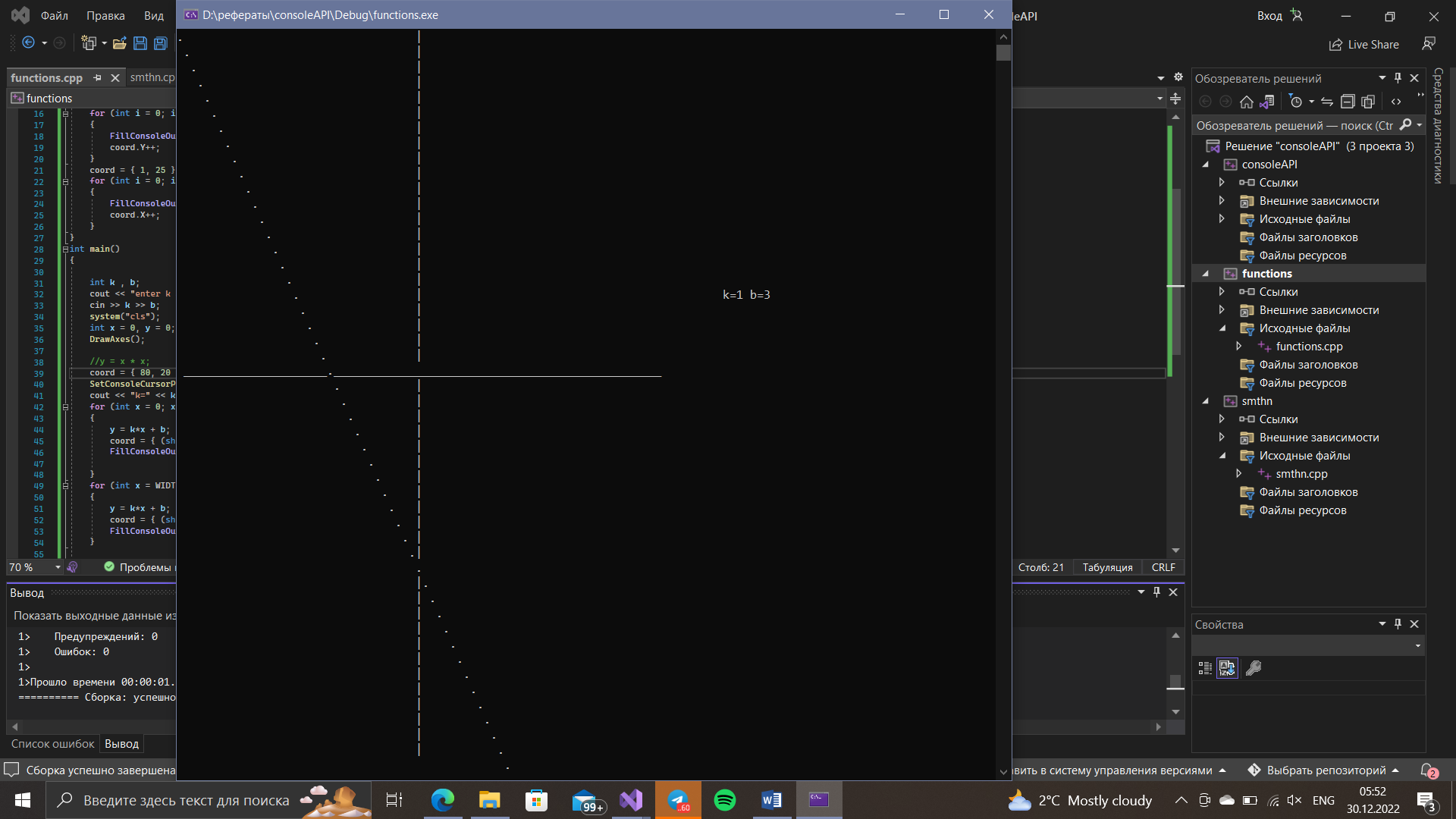
coord = { 0, HEIGHT + 2 };

SetConsoleCursorPosition(hOutput, coord);

system("pause");

}

Результат:



**Вывод**

Функции WinAPI в настоящее время в основном используются для написания оконных приложений, однако и для консольных их функционал велик.