//1.Задача: найти разложение натурального числа на простые множители(факторизация числа).

```
	//Исходный код на языке C++
#include <iostream>
#include <vector>

	using namespace std;

		template <typename intT>
		vector<intT> fact(intT num) {

			intT j = 2;
			vector<intT> res;
			while (num / intT(2) >= j) {
				if (num % j == 0) {
					res.push_back(j);
					num /= j;
					j = intT(2);
				}
				else {
					++j;
				}
			}
			res.push_back(num);
			return res;
		}


		//typedef unsigned long integralT;
		typedef __int64 integralT;

		int main() {
```

```
vector<integralT> v;
integralT n;

while (true) {
	cout << "Enter positive number: ";
	cin >> n;
	v.clear();
	v = fact(n);

	for (auto i = v.begin(); i != v.end(); i++) {
		if (i != v.begin())
			cout << ", ";
		cout << *i;
	}
	cout << endl << endl;
}

return 0;
```

//2.Вычислить оптимальный срок вклада

//selevit 17 мая 2015

//Гражданин открыл счет в банке, вложив 10000 рублей.Через каждый месяц размер вклада увеличивается на p % (годовой процент) от имеющейся суммы(p — вещественное число, 0 < p < 25 годовой процент).

//По данному p определить, через сколько месяцев размер вклада превысит 11000 руб, и вывести найденное количество месяцев k(целое число) и итоговый размер вклада(вещественное число).

//Исходный код на языке C++

```
#include <iostream>
```

```
#include <cmath>

	using namespace std;


int main()
{
	double kap = 10000;
	int p;

	cout << "vvedite procent ot 0 do 25." << endl;
	cin >> p;
	//goto r;
	int mes = 0;

	while (kap <= 11000)
	{
		kap = kap + (kap / 100.0 * p);
		mes++;
	}

	cout << "mesjacov: " << mes << endl;
	cout << "kapital: " << kap << endl;
	//r;
	return 0;
}
//Источник: code - live.ru



//3.Проверить, лежит ли точка на отрезке


//selevit  17 мая 2015
//Ввести с клавиатуры координаты начала и конца отрезка на плоскости.Ввести координаты точки.Проверить, лежит ли эта точка на отрезке.
```

```
//Исходный код на языке C++
#include <iostream>
#include <cstdlib>

using namespace std;

double max(double x, double y)
{
	if (x < y) {
		return x;
	}
	return x;
}

double min(double x, double y)
{
	if (x > y) {
		return y;
	}
	return x;
}



bool thc(double x, double y, double z, double w, double a, double b)
{
	double k, c;

	if (z == x) {
		return (a == x && b >= min(y, w) && x <= max(y, w));
	}
```

```
	k = (w - y) / (z - x);
	c = y - k * x;

	return b == a * k + c;
}

int main(int argc, char* argv[])
{
	setlocale(LC_ALL, "Russian");

	double x, y; // Координаты начала отрезка
	double z, w; // Координаты конца отрезка
	double a, b; // Координаты точки

	bool result;

	cout << "Координаты начала отрезка: ";
	cin >> x >> y;

	cout << "Координаты конца отрезка: ";
	cin >> z >> w;

	cout << "Координаты точки: ";
	cin >> a >> b;

	result = thc(x, y, z, w, a, b);
	cout << result << endl;
	system("pause");

	return 0;
}
//Источник: code - live.ru
```

```
//4.Вывод двоичного представления целого десятичного числа

//porshe  целые числа двоичный код  15 июня 2016
//Вывести двоичное представление данного десятичного целого числа.
//Код взят из темы на форуме.
//Автор идеи : Cranium

//Исходный код на языке C++

#include <cstdio>
#include <string>

//Сразу напишем общую версию для всех целочисленных типов
template <typename T>
std::string intToBin(T val) {
	if (val == 0)
		return "0";  //Здесь сработает конструктор std::string


	//Буфер для записи двоичного представления
	//Всего битов в числе sizeof(T) * 8, и ещё оди
	//н на терминальный символ
	char bary[sizeof(T) * 8 + 1];

	//Индекс записи в буфер выше
	int idigit = 0;

	bool meetOne = false;

	//Спускаемся от старшего к младшему биту
	for (int i = sizeof(T) * 8 - 1; i >= 0; i--)
```

```
        {
                if (val & (T(1) << i)) //Если бит номер i
                        //установлен...
                {
                        //...записываем 1 в соответствующий бит
                        //строкового представления
                        //и устанавливаем флаг встречи с 1
                        meetOne = true;
                        bary[idigit++] = '1';
                }
                else
                {
                        //...записываем ноль, но только если до этого
                        //была единица,
                        //так избавимся от ненужных ведущих нулей
                        if (meetOne)
                        {
                                bary[idigit++] = '0';
                        }
                }
        }
        //Завершение строки терминальным символом
        bary[idigit] = '\0';

        return bary;  //Конструктор std::string
}


int main()
{
        int val;
        while (std::scanf("%d", &val) != EOF)
```

```
        {
                std::printf("%s ", intToBin(val).c_str());
        }

        return 0;// C4996_containers.cpp
// compile with: cl /c /W4 /D_DEBUG C4996_containers.cpp
#include <algorithm>

bool example(
   char const * const left,
   const size_t leftSize,
   char const * const right,
   const size_t rightSize)
{
   bool result = false;
   result = std::equal(left, left + leftSize, right); // C4996
   // To fix, try this form instead:
   // result = std::equal(left, left + leftSize, right, right + rightSize); // OK
   return result;
}
}

//Источник: code - live.ru



//5.Программа работает 21-02-2019 г. Визуал Студио С++

//Разложение натурального числа на простые множители

//Cranium  факторизация числа  30 ноября 2015
//Задача: найти разложение натурального числа на простые множители(факторизация числа).
```

```
//		Исходный код на языке C++
#include <iostream>
#include <vector>

	using namespace std;

template <typename intT>
vector<intT> fact(intT num) {

	intT j = 2;
	vector<intT> res;
	while (num / intT(2) >= j) {
		if (num % j == 0) {
			res.push_back(j);
			num /= j;
			j = intT(2);
		}
		else {
			++j;
		}
	}
	res.push_back(num);
	return res;
}


//typedef unsigned long integralT;
typedef __int64 integralT;

int main() {

	vector<integralT> v;
```

```
	integralT n;

	while (true) {
		cout << "Enter positive number: ";
		cin >> n;
		v.clear();
		v = fact(n);

		for (auto i = v.begin(); i != v.end(); i++) {
			if (i != v.begin())
				cout << ", ";
			cout << *i;
		}
		cout << endl << endl;
	}

	return 0;
}
```

//6.Задача: найти разложение натурального числа на простые множители(факторизация числа).

```
	//Исходный код на языке C++
#include <iostream>
#include <vector>

	using namespace std;

		template <typename intT>
		vector<intT> fact(intT num) {

			intT j = 2;
			vector<intT> res;
```

```
	while (num / intT(2) >= j) {
		if (num % j == 0) {
			res.push_back(j);
			num /= j;
			j = intT(2);
		}
		else {
			++j;
		}
	}
	res.push_back(num);
	return res;
}


//typedef unsigned long integralT;
typedef __int64 integralT;

int main() {

	vector<integralT> v;
	integralT n;

	while (true) {
		cout << "Enter positive number: ";
		cin >> n;
		v.clear();
		v = fact(n);

		for (auto i = v.begin(); i != v.end(); i++) {
			if (i != v.begin())
				cout << ", ";
```

```
                                        cout << *i;
                                }
                                cout << endl << endl;
                        }

                        return 0;
                }

//7.Рекурсивное вычисление факториала

//selevit  рекурсия  4 мая 2015
//Написать функцию вычисления факториала числа, используя рекурсию.

//Исходный код на языке C++
#include <iostream>
using namespace std;

long double fact(int N)
{
        if (N < 0) // если пользователь ввел отрицательное число
                return 0; // возвращаем ноль
        if (N == 0) // если пользователь ввел ноль,
                return 1; // возвращаем факториал от нуля - не удивляетесь, но это 1 =)
        else // Во всех остальных случаях
                return N * fact(N - 1); // делаем рекурсию.
}

int main()
{
        int N;
        setlocale(0, ""); // Включаем кириллицу
        cout << "Введите число для вычисления факториала: ";
```

```
	cin >> N;
	cout << "Факториал для числа " << N << " = " << fact(N) << endl << endl; // fact(N) - функция для вычисления факториала.
	return 0;
}
```

//Источник: code - live.ru

//8.Поиск максимального элемента в массиве

//selevit массивы 5 мая 2015

//Ввести целочисленный массив из N элементов с клавиатуры.Найти максимальный элемент этого массива.

//Исходный код на языке C++

```
/*
 * Ввести целочисленный массив из N целых чисел.
 * Найти максимальный элемент массива
*/
#include <iostream>

using namespace std;

int main()
{
	int *arr; // указатель для выделения памяти под массив
	int size; // размер массива

	// Ввод количества элементов массива
	cout << "n = ";
	cin >> size;

	if (size <= 0) {
```

```
		// Размер масива должен быть положитлеьным
		cerr << "Invalid size" << endl;
		return 1;
	}

	arr = new int[size]; // выделение памяти под массив

	// заполнение массива
	for (int i = 0; i < size; i++) {
		cout << "arr[" << i << "] = ";
		cin >> arr[i];
	}

	// Нахождение максимального элемента
	int max = arr[0];
	for (int i = 1; i < size; i++) {
		if (arr[i] > max) {
			max = arr[i];
		}
	}

	// Вывод результата на экран
	cout << "max = " << max << endl;

	delete[] arr; // освобождение памяти

	return 0;
}
```

//9.Сортировка массива методом пузырька

//selevit массивы сортировка 5 мая 2015

//Ввести целочисленный массив из N 'элементов с клавиатуры. Отсортировать его по возрастанию методом пузырька.

//Исходный код на языке C++

```
/*
 * Ввести целочисленный массив из N целых чисел.
 * Отсортировать этот массив по возрастанию методом пузырька
*/
#include <iostream>

using namespace std;

int main()
{
	int *arr; // указатель для выделения памяти под массив
	int size; // размер массива

	// Ввод количества элементов массива
	cout << "n = ";
	cin >> size;

	if (size <= 0) {
		// Размер масива должен быть положитлеьным
		cerr << "Invalid size" << endl;
		return 1;
	}

	arr = new int[size]; // выделение памяти под массив

	// заполнение массива
```

```
	for (int i = 0; i < size; i++) {
		cout << "arr[" << i << "] = ";
		cin >> arr[i];
	}

	int temp; // временная переменная для обмена элементов местами

	// Сортировка массива пузырьком
	for (int i = 0; i < size - 1; i++) {
		for (int j = 0; j < size - i - 1; j++) {
			if (arr[j] > arr[j + 1]) {
				// меняем элементы местами
				temp = arr[j];
				arr[j] = arr[j + 1];
				arr[j + 1] = temp;
			}
		}
	}

	// Вывод отсортированного массива на экран
	for (int i = 0; i < size; i++) {
		cout << arr[i] << " ";
	}
	cout << endl;

	delete[] arr; // освобождение памяти;

	return 0;
}
```

//10.Сохранение данных из текстового файла в строку

//selevit fstream файлы 17 мая 2015

//Открыть текстовый файл example.txt, сохранить его содержимое в строку и вывести на экран.

//Исходный код на языке C++

```
/*
 Чтение данных из текстового файла в строку и вывод на экран
*/
#include <iostream>
#include <fstream>

using namespace std;

int main(int argc, char* argv[])
{
	ifstream fp("example.txt", ios::in);
	if (fp.is_open()) {
		// Если файл открыт успешно
		// Получение размера файла
		fp.seekg(0, fp.end);
		int length = fp.tellg();
		char *buffer = new char[length];
		fp.seekg(0, fp.beg);
		// Считывание данных из файла
		fp.read(buffer, length);
		// Закрытие файла
		fp.close();
		// Вывод данных на экран
		cout << buffer;
		// Удаление буфера, в котором сохранен контент
		// файла
```

```
		delete[] buffer;
	}
	else {
		cerr << "Ошибка открытия файла example.txt" << endl;
	}
	return 0;
}
```

//11.Разбиение натурального числа на слагаемые

//selevit 17 мая 2015

//Построить таблицу всех различных разбиений заданного натурального числа N на сумму трех натуральных слагаемых(разбиения, отличающиеся порядком слагаемых, различными не считаются).

//Исходный код на языке C++

```
#include <iostream>

int main()
{
	std::cout << "Enter a natural number: ";

	unsigned int N;

	std::cin >> N;

	std::cout << "The number " << N << " can be present by next three summands:\n";

	unsigned int limit_1_summand = (N / 3) + 1;
	unsigned int limit_2_summand;

	for (unsigned int i = 0; i < limit_1_summand; i++) {
```

```
                limit_2_summand = ((N - i) / 2) + 1;

                for (unsigned int j = i; j < limit_2_summand; j++) {

                        std::cout << i << " " << j << " " << N - i - j << std::endl;

                }
        }

        std::cin.get();
        std::cin.get();

        return 0;
}
```

//Источник: code - live.ru

//12.Поиск четырех максимальных элементов в массиве

//selevit массивы сортировка 11 мая 2015

//Для поиска четырех максимальных элементов в массиве, он сначала сортируется с помощью алгоритма быстрой сортировки, а потом берутся 4 крайних элемента.

//Исходный код на языке C++

```
#include <iostream>
using namespace std;

static void quick_sort(int *arr, int low, int high) {
        // Алгоримт быстрой сортировки
        // http://ru.wikipedia.org/wiki/Быстрая_сортировка

        int i = low;
```

```
	int j = high;
	int x = arr[(low + high) / 2];
	int temp;

	do {
		while (arr[i] < x) {
			++i;
		}
		while (arr[j] > x) {
			--j;
		}
		if (i <= j) {
			temp = arr[i];
			arr[i] = arr[j];
			arr[j] = temp;
			i++; j--;
		}
	} while (i <= j);

	if (low < j) {
		quick_sort(arr, low, j);
	}
	if (i < high) {
		quick_sort(arr, i, high);
	}
}

int main()
{
	// размер массива, задается пользователем
	int size;
	cin >> size;
```

```
	int *list = new int[size];

	// заполнить массив руками или из файла
	for (int i = 0; i < size; ++i) {
		cin >> list[i];
	}

	// отсортировать массив по возрастанию
	quick_sort(list, 0, size - 1);

	// вывести последние 4 элемента отсортированного массива
	for (int i = size - 1; i > size - 5; --i) {
		cout << list[i] << endl;
	}
	return 0;
}
//Источник: code - live.ru

//13.Программа змейка
#include <iostream>  //стандартная библиотека
#include <time.h> //случайные числа
#include <stdio.h> //для printf
#include <windows.h> // для HANDLE, курсора, цвета
#include <conio.h>  //для kbhit

using namespace std;



HANDLE hConsole;
//HANDLE hStdout, hStdin;
```

```
HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);


void GotoXY(int X, int Y)
{
	COORD coord = { X, Y };
	SetConsoleCursorPosition(hStdOut, coord);
}
//Цвет
enum ConsoleColor
{
	Black = 0,
	Blue = 1,
	Green = 2,
	Cyan = 3,
	Red = 4,
	Magenta = 5,
	Brown = 6,
	LightGray = 7,
	DarkGray = 8,
	LightBlue = 9,
	LightGreen = 10,
	LightCyan = 11,
	LightRed = 12,
	LightMagenta = 13,
	Yellow = 14,
	White = 15
};

void SetColor(ConsoleColor text, ConsoleColor background)
{
	SetConsoleTextAttribute(hStdOut, (WORD)((background << 4) | text));
}
```

```
class Zmeja  // структура змейка
{
public:COORD *t; //точки
public:int PCount; //количество яблок
};
enum uprawlenie { LEFT, UP, RIGHT, DOWN }; //направление змейки
class Game //даные-точности: змейки, яблок, передвижение по X и Y, задержка, направление
{
public:Zmeja gaduka; //змейка
public:COORD jabloko; //яблоко
public:int dx, dy; //передвижение
public:int pause; //задержка
public:int nap; //направление

};


void PlusJabloko(Game &g) //Функция разброски яблок
{
	int i, x, y;
	int n = g.gaduka.PCount;
	do
	{
		x = rand() % 56 + 3; //
		y = rand() % 19 + 3; //кординаты яблока
		for (i = 0; i < n; i++)
		{
			if (x == g.gaduka.t[i].X && y == g.gaduka.t[i].Y) // проверка чтоб яблоко не
бросить на змею
				break;
		}
```

```
	} while (i < n);
	g.jabloko.X = x; //
	g.jabloko.Y = y; //запоминаем позицию яблока
	SetConsoleCursorPosition(hConsole, g.jabloko); //переносим курсор в эту позицию
	SetConsoleTextAttribute(hConsole, 0x0c); //цвет яблока
	printf("%c", 4); //рисуем яблоко каким хотим символом

}

void skorostGame(Game &g) // Функцыя старта змейки ее координат и скорости
{
	system("cls");
	g.gaduka.PCount = 3; //сколько точек в змейке
	g.gaduka.t = new COORD[3];//создали точки
	for (int i = 0; i < 3; i++)
	{
		g.gaduka.t[i].X = 20 + i;
		g.gaduka.t[i].Y = 20;

	}
	g.dx = 1;
	g.dy = 0;
	g.pause = 100;//скорость передвижение змеи
	PlusJabloko(g);//рисуем яблока
}
void Level()
{
	GotoXY(10, 10); cout << "Vy nikogda ne vyigraete" << endl; //НАДПИСЬ: Вы никогда не выиграете
	GotoXY(10, 11); cout << "esli ne budete bditelny!" << endl; //НАДПИСЬ: Если не будете бдительны!
}
```

```
void ZmejaStart()
{
	GotoXY(10, 15); cout << "Soberi 50 yablok, togda posmotrim ;)" << endl; //НАДПИСЬ: Собери 50 яблок, тогда посмотрим ;)
}
void STENA_2() //Вся информация, отображаемая на стене
{
	SetColor(LightBlue, Black); GotoXY(20, 0); cout << "Snake game by Danilenko Alexander" << endl;  //НАДПИСЬ: Игра Змейка Даниленко Александра
	GotoXY(64, 2); cout << "Dannue:" << endl; //Данные
	GotoXY(64, 3); cout << "Yablok:0" << endl; //Яблок
	GotoXY(64, 4); cout << "Dlina:3" << endl; //Длина
	GotoXY(64, 5); cout << "Speed:0" << endl; //Скорость
	GotoXY(64, 7); cout << "Uprawlenie:" << endl; //Управление
	GotoXY(64, 8); cout << "Esc:Wuxod" << endl; //Выход
	GotoXY(64, 9); cout << "P:Pause" << endl; //Пауза
	GotoXY(64, 10); cout << "S:Start" << endl; //Старт
	GotoXY(64, 11); cout << "L:Level" << endl; //Уровень
	GotoXY(64, 13); printf("%c", 24); cout << ":Wwerx" << endl; //Вверх
	GotoXY(64, 14); printf("%c", 25); cout << ":Wniz" << endl;   //Вниз
	GotoXY(64, 15); printf("%c", 27); cout << ":Wlewo" << endl;  //Влево
	GotoXY(64, 16); printf("%c", 26); cout << ":Wprawo" << endl; //Вправо
	{SetColor(LightMagenta, Black);

	GotoXY(2, 2); //Рисуем верхнюю горизонтальную линию-стенку
	int m = 0;
	for (m = 0; m < 60; m++)
	{
		printf("*");
	}
	}

	{
```

```
	GotoXY(2, 24); //Рисуем нижнюю горизонтальную линию-стенку
	int m = 0;
	for (m = 0; m < 60; m++)
	{
		printf("*");
	}
}
{	//Рисуем левую вертикальную линию-стенку
	GotoXY(2, 3); cout << "*" << endl;
	GotoXY(2, 4); cout << "*" << endl;
	GotoXY(2, 5); cout << "*" << endl;
	GotoXY(2, 6); cout << "*" << endl;
	GotoXY(2, 7); cout << "*" << endl;
	GotoXY(2, 8); cout << "*" << endl;
	GotoXY(2, 9); cout << "*" << endl;
	GotoXY(2, 10); cout << "*" << endl;
	GotoXY(2, 11); cout << "*" << endl;
	GotoXY(2, 12); cout << "*" << endl;
	GotoXY(2, 13); cout << "*" << endl;
	GotoXY(2, 14); cout << "*" << endl;
	GotoXY(2, 15); cout << "*" << endl;
	GotoXY(2, 16); cout << "*" << endl;
	GotoXY(2, 17); cout << "*" << endl;
	GotoXY(2, 18); cout << "*" << endl;
	GotoXY(2, 19); cout << "*" << endl;
	GotoXY(2, 20); cout << "*" << endl;
	GotoXY(2, 21); cout << "*" << endl;
	GotoXY(2, 22); cout << "*" << endl;
	GotoXY(2, 23); cout << "*" << endl;
}
{  //Рисуем правую вертикальную линию-стенку
	GotoXY(61, 3); cout << "*" << endl;
```

```
                GotoXY(61, 4); cout << "*" << endl;
                GotoXY(61, 5); cout << "*" << endl;
                GotoXY(61, 6); cout << "*" << endl;
                GotoXY(61, 7); cout << "*" << endl;
                GotoXY(61, 8); cout << "*" << endl;
                GotoXY(61, 9); cout << "*" << endl;
                GotoXY(61, 10); cout << "*" << endl;
                GotoXY(61, 11); cout << "*" << endl;
                GotoXY(61, 12); cout << "*" << endl;
                GotoXY(61, 13); cout << "*" << endl;
                GotoXY(61, 14); cout << "*" << endl;
                GotoXY(61, 15); cout << "*" << endl;
                GotoXY(61, 16); cout << "*" << endl;
                GotoXY(61, 17); cout << "*" << endl;
                GotoXY(61, 18); cout << "*" << endl;
                GotoXY(61, 19); cout << "*" << endl;
                GotoXY(61, 20); cout << "*" << endl;
                GotoXY(61, 21); cout << "*" << endl;
                GotoXY(61, 22); cout << "*" << endl;
                GotoXY(61, 23); cout << "*" << endl;
        }
}


//Функция которая двигает и рисует
enum { KONEC, STENA, PLUS, MOVE };
int Move(Game &g)
{
        int & n = g.gaduka.PCount;
        COORD head = g.gaduka.t[n - 1]; //голова
        COORD tail = g.gaduka.t[0]; //хвост
        COORD next;
```

```
	next.X = head.X + g.dx;
	next.Y = head.Y + g.dy; //проверка следующей точки по направлению

	if (next.X < 3 || next.Y < 3 || next.X > 60 || next.Y > 23)//не уперлась ли в стену?
		return STENA;

	if (n > 4)
	{
		for (int i = 0; i < n; i++)
			if (next.X == g.gaduka.t[i].X && next.Y == g.gaduka.t[i].Y) //не наехали ли на себя?
				return KONEC;
	}

	if (next.X == g.jabloko.X && next.Y == g.jabloko.Y)
	{
		COORD*temp = new COORD[++n]; //новый масив больший на 1
		for (int i = 0; i < n; i++)
			temp[i] = g.gaduka.t[i]; //перекопируем
		temp[n - 1] = next; //добавляем одну
		delete[] g.gaduka.t;
		g.gaduka.t = temp;

		SetConsoleCursorPosition(hConsole, head);
		SetConsoleTextAttribute(hConsole, 0x0a); //закрашываем яблоко которое сели
		printf("*");
		SetConsoleCursorPosition(hConsole, next);
		SetConsoleTextAttribute(hConsole, 0x0a);
		printf("%c", 1);
		PlusJabloko(g);
		return PLUS;
	}
```

```
	for (int i = 0; i < n - 1; i++)
		g.gaduka.t[i] = g.gaduka.t[i + 1];
	g.gaduka.t[n - 1] = next;
	SetConsoleCursorPosition(hConsole, tail);//закрашиваем хвостик
	printf(" ");

	SetConsoleCursorPosition(hConsole, head);
	SetConsoleTextAttribute(hConsole, 0x0a);//красим хвост змеи в зелений цвет
	printf("*");
	SetConsoleCursorPosition(hConsole, next);
	SetConsoleTextAttribute(hConsole, 0x0f); //красим курсор в белый цвет (голову змеи)
	printf("%c", 1);

	return MOVE;
}

int intro()
{
	GotoXY(3, 10); //Интруха
	printf("ЋЎлз- п §¬Ґ©Є . “Їа ў«Ґ-ЁҐ §¬ҐоЄ®© - бваҐ«®зЄ ¬Ё. Esc - ўле®¤ Ё§ ЁЈал.");
	GotoXY(15, 11);
	printf("…б«Ё б®ЎҐасвҐ 50 пЎ«®Є, в® ў б ¦¤св боаЇаЁ§ ;)");
	GotoXY(18, 15);
	printf("„«п Їа®¤®«¦Ґ-Ёп влЄ-ЁвҐ -  -ЁЄҐ© :D");
	getch();
}

int main()
{

	hConsole = GetStdHandle(STD_OUTPUT_HANDLE); //получаем дескриптор консоли
```

```
intro();
int key = 0, count = 0;
bool Pause = false;
Game g;
skorostGame(g);
STENA_2();
srand(time(0));
bool pause = false;
while (key != 27)
{
	while (!kbhit()) //ждет пока нажмем
	{
		if (Pause == true)
		{
			Sleep(1);
			continue;
		}

		switch (Move(g))//движение
		{

		case PLUS:
			++count;
			g.pause -= 1;
			SetColor(LightBlue, Black);
			GotoXY(64, 2); cout << "Danue:" << endl;
			GotoXY(64, 3); cout << "Jablok:" << count << endl;
			GotoXY(64, 4); cout << "Dlina:" << g.gaduka.PCount << endl;
			GotoXY(64, 5); cout << "Speed:" << g.pause << endl;
			GotoXY(64, 7); cout << "Uprawlenie:" << endl;
			GotoXY(64, 8); cout << "Esc:Wuxod" << endl;
			GotoXY(64, 9); cout << "P:Pause" << endl;
```

```
                                GotoXY(64, 10); cout << "S:Start" << endl;
                                GotoXY(64, 11); cout << "L:Level" << endl;
                                GotoXY(64, 13); printf("%c", 24); cout << ":Wwerx" << endl;
                                GotoXY(64, 14); printf("%c", 25); cout << ":Wniz" << endl;
                                GotoXY(64, 15); printf("%c", 27); cout << ":Wlewo" << endl;
                                GotoXY(64, 16); printf("%c", 26); cout << ":Wprawo" << endl;
                                if (count == 50)
                                {
                                        SetColor(White, Black);
                                        GotoXY(24, 1); cout << "Vy vyigrali! Pozdravlyayu! Vy ne
chelovek! xD" << endl; //Вы выиграли
                                        getch();
                                        return(0);
                                }
                                break;

                        case STENA:

                        case KONEC:
                                GotoXY(23, 1); printf(",л Їа®ЁJa «Ё! • -e -e !!!\n\n\t\t\t"); //Вы
проиграли, XA XA XA
                                getch();
                                break;
                        }

                        Sleep(g.pause); //Задержка
                }
                key = getch();

                if (key == 'P' || key == 'p')
                        Pause = !Pause;
                else if (key == 'S' || key == 's')
                        ZmejaStart();
```

```
        else if (key == 'L' || key == 'l')
                Level();
        else if (key == 0 || key == 224)
        {
                key =  getch();

                if (key == 72 && g.nap != DOWN)
                {
                        g.nap = UP;
                        g.dx = 0;
                        g.dy = -1;
                }
                else if (key == 80 && g.nap != UP)
                {
                        g.nap = DOWN;
                        g.dx = 0;
                        g.dy = 1;
                }
                else if (key == 75 && g.nap != RIGHT)
                {
                        g.nap = LEFT;
                        g.dx = -1;
                        g.dy = 0;
                }
                else if (key == 77 && g.nap != LEFT)
                {
                        g.nap = RIGHT;
                        g.dx = 1;
                        g.dy = 0;
                }
        }
}
```

```
}


//14.Найти минимальное и максимальное значений из трех чисел


//selevit  17 мая 2015
//Исходный код на языке C++
#include <iostream>
using namespace std;

int main(int argc, char* argv[])
{
	setlocale(LC_ALL, "");
	int a, b, c;

	cout << "Введите 3 числа через пробел: ";
	cin >> a >> b >> c;

	int min = a;
	int max = a;

	// Нахождениа минимума
	if (b < min) {
		min = b;
	}
	if (c < min) {
		min = c;
	}

	// Нахождение максимума
	if (b > max) {
		max = b;
```

```
        }
        if (c > max) {
                max = c;
        }

        cout << "Max: " << max << endl;
        cout << "Min: " << min << endl;

        return 0;
}
```