Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное   
образовательное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Лабораторная Работа №1. ЧАСТОТНЫЙ АНАЛИЗАТОР

Отчет по лабораторной работе №1 по учебной дисциплине   
«Методы защиты информации»

по специальности 09.03.01 Информатика и вычислительная техника

|  |  |
| --- | --- |
|  | Руководитель  / Т. В. Жгун  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |
|  | Студент группы 3091  / Р. А. Михайлов  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Оглавление

[План работы 3](#_Toc2259)

[Теоретические положения 4](#_Toc9371)

[Алгоритм работы программы 5](#_Toc11946)

[Приложение А 12](#_Toc14258)

# План работы

1. Реализовать инструмент, обеспечивающий частотный анализ файлов графического (\*.bmp, 3 канала по 8 бит), и текстового формата (\*.txt, 8 бит). Кодировка символов – ASCII представлена в таблице 1. Все форматы представлены числами в диапазоне [0, 255].

Анализ файла представляет собою гистограмму текста и таблицу встречаемости символов. Все гистограммы должны быть реализованы для одного диапазона [0, 255].

Реализованный частотный анализатор должен использоваться при выполнении следующих лабораторных работ.

1. Проанализировать распределение символов для графических файлов различного свойства (картинка с преобладанием одного цвета, картинка в графическом редакторе, фото, …) и текстов разного свойства (текст учебника, статья в СМИ, литературный текст, переписка в соцсетях, …). Объем текстового файла не менее 2 тысяч символов.

В отчете представить не менее 4х гистограмм для файлов каждого типа

1. Вычислить информационную меру рассматриваемых файлов, используя информацию шага 1. Обосновать возможности сжатия текстов

# Теоретические положения

Частотный анализ — это метод, используемый в криптографии и лингвистике для изучения частоты появления различных символов, букв или слов в тексте. Он основан на предположении, что в любом языке определенные символы или комбинации символов встречаются с разной частотой.

В криптографии частотный анализ может быть использован для взлома шифров, особенно простых, таких как шифр Цезаря или шифр подстановки. Например, если в зашифрованном тексте буква "E" (которая является самой частой буквой в английском языке) встречается чаще всего, то можно предположить, что она соответствует наиболее часто встречающейся букве в зашифрованном сообщении.

Одной из функций программы будет возможность вычисления энтропии данного файла. Энтропия в теории информации измеряет среднее количество информации, которое содержится в сообщении, и определяется как мера неопределенности, связанной с вероятностным распределением возможных исходов.

Низкая энтропия: если система имеет низкую энтропию, это означает, что она предсказуема. Например, если в тексте часто повторяются одни и те же слова, то его энтропия будет низкой, так как информация в нем менее разнообразна.

Высокая энтропия: высокая энтропия указывает на большую степень неопределенности. Например, если в тексте используются множество уникальных слов с равной вероятностью, то его энтропия будет высокой, так как предсказать следующее слово будет сложно.

Энтропия вычисляется по следующей формуле:

# Алгоритм работы программы

Для реализации частотного анализатора алгоритмическим языком был выбран C++, язык для написания графического интерфейса – C++/CLI (платформа Microsoft Windows Forms), код программы представлен в Приложении А.

Для работы с программой необходимо запустить FrequencyAnalysis.exe. После запуска на экране будет представлен графический интерфейс приложения:

– Координатная плоскость, где ось x – числовое значение символа/пикселя, ось y – количество символов/пикселей с данным числовым значением

– Таблица в формате <числовое значение>. <количество символов/пикселей>, внизу таблицы находится значение энтропии выбранного файла.

– Верхняя часть окна приложения – меню с возможностью выбора файла, настройкой гистограммы и кнопкой выхода из приложения.

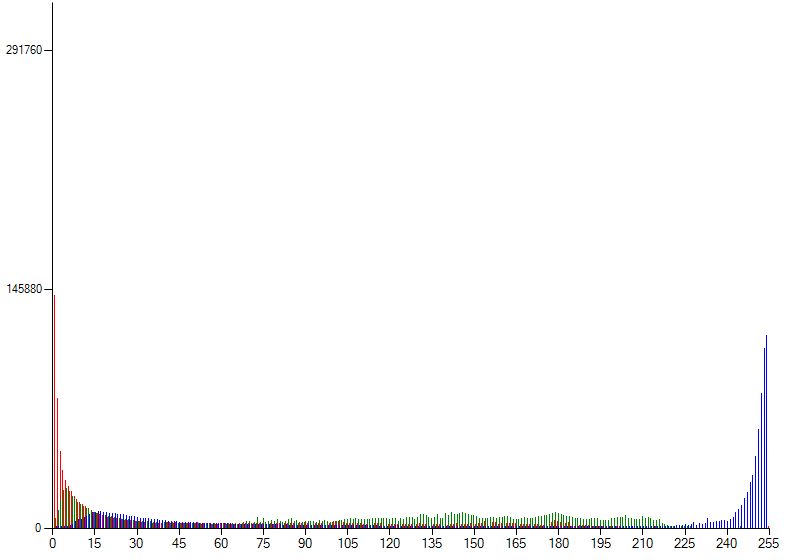
После выбора файла программа проходит по данному тексту/изображению и сохраняет в структуру данных std::map (ключ – значение, в нашем случае ключ это числовое значение пикселя/символа, а значение это количество символов/пикселей с данным числовым значением) количество символов с данным значением.

После того, как файл был проанализирован, программа отображает полученные данные в виде гистограммы, полученные гистограммы представлены в Приложении Б.

В таблице будут выведены числовые значения и количество символов/пикселей с такими числовыми значениями, внизу таблицы выведена энтропия данного файла.

Ниже представлены гистограммы полученные при частотном анализе изображений разного типа и текстов разного стиля.

Используемые тексты представлены в приложении Б.

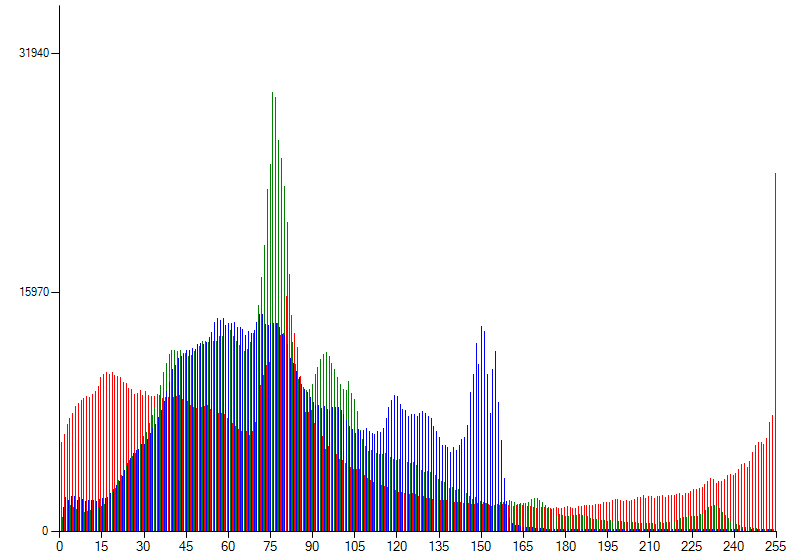


H(А) = 7,332 бит

Рисунок 1 – Гистограмма изображения с преобладанием одного цвета



Рисунок 2 – Изображние с преобладанием одного цвета

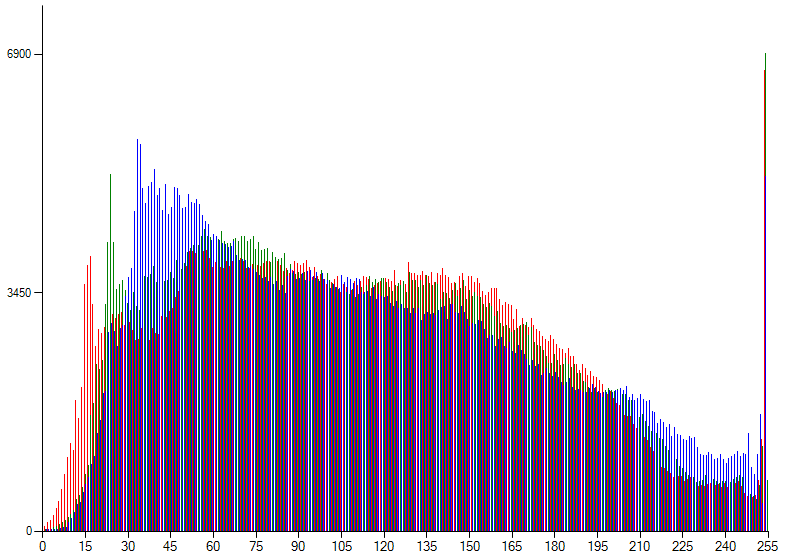


H(x) = 7,532 бит

Рисунок 3 – Гистограмма изображение в графическом редакторе



Рисунок 4 – Изображение в графическом редакторе

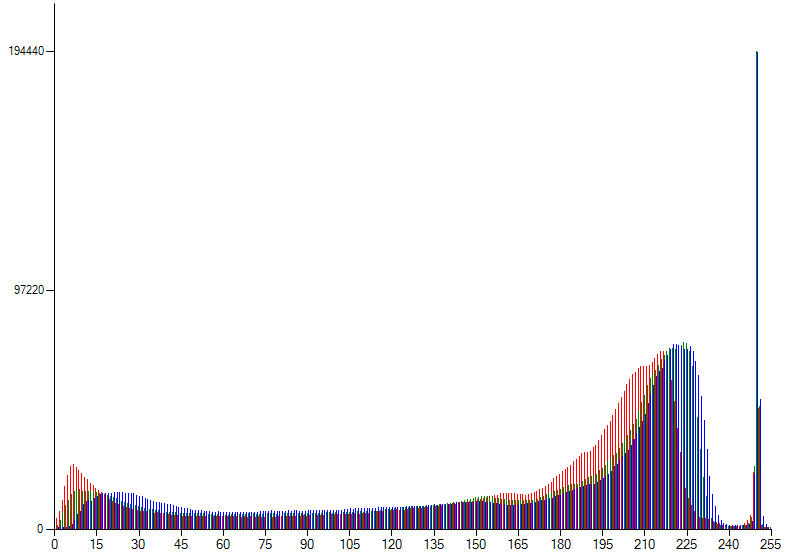


H(x) = 7,819 бит

Рисунок 5 – Гистограмма фотографии №1



Рисунов 6 – Фотография №1

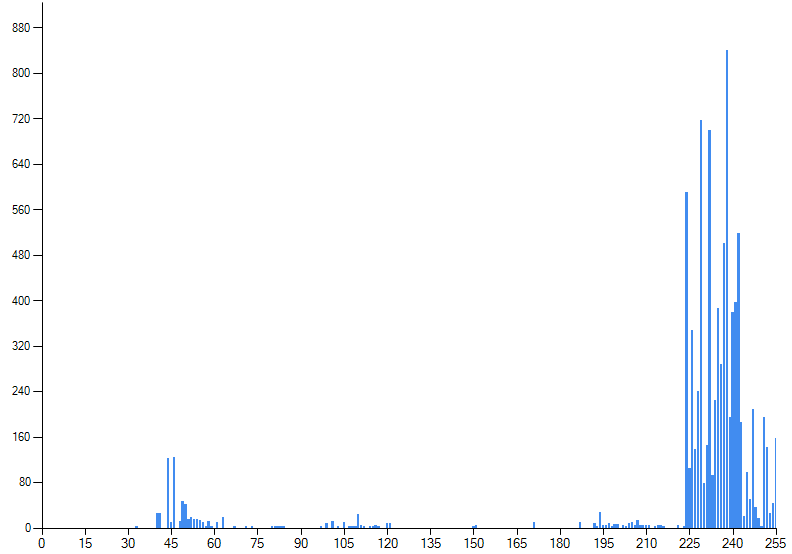


H(x) = 7,346 бит

Рисунок 7 – Гистограмма фотографии №2

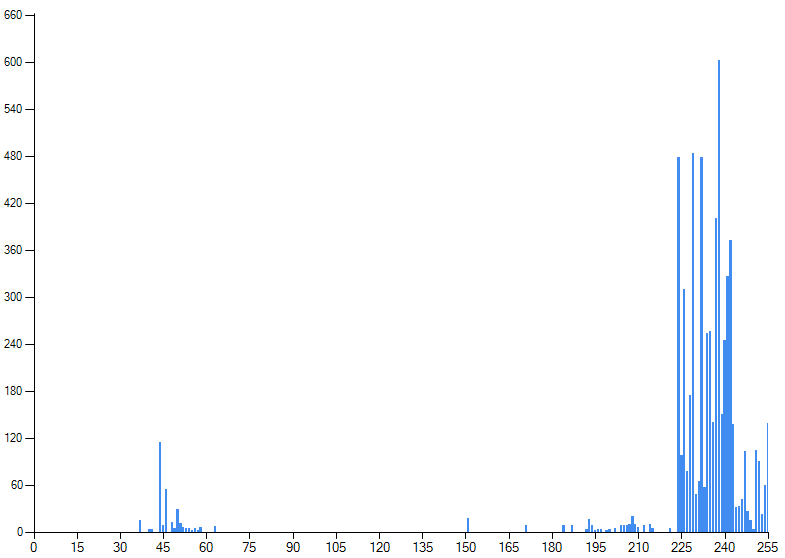


Рисунок 8 – Фотография №2



H(x) = 4,949 бит

Рисунок 5 – Гистограмма текста учебника



H(x) = 4,829 бит

Рисунок 6 – Гистограмма статьи в СМИ

H(x) = 4,965 бит

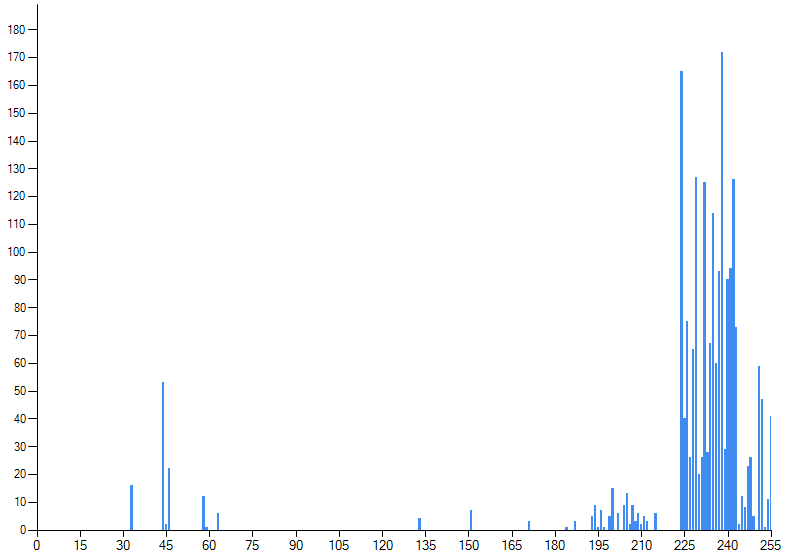
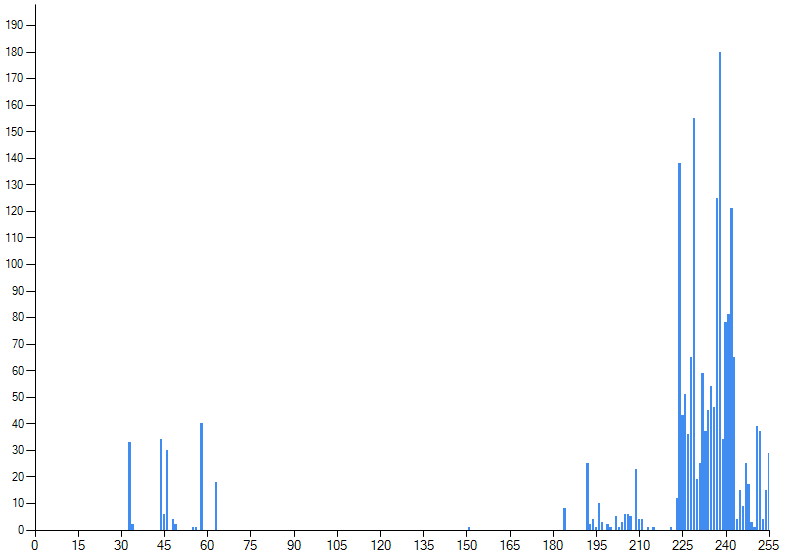


Рисунок 7 – Гистограмма литературного текста



H(x) = 5,057 бит

Рисунок 8 – Гистограмма текста переписки в соцсетях

# Приложение А

Листинг файла Analisys.h

#include <iostream>

#include <string>

#include <fstream>

#include <unordered\_map>

#include <Windows.h>

#include <cmath>

void init\_map(std::unordered\_map<char, int>& data)

{

for (int i = 0; i < 256; i++)

{

data.insert({ (char)i, 0 });

}

}

std::unordered\_map<int, int> init\_map\_RGB()

{

std::unordered\_map<int, int> data;

for (int i = 0; i < 256; i++)

{

data.insert({ i, 0 });

}

return data;

}

int find\_max(std::unordered\_map<int, int> data)

{

int max = data[0];

for (int i = 0; i < 256; i++)

{

if (data[i] > max)

max = data[i];

}

return max;

}

int find\_max(std::unordered\_map<char, int>& data)

{

int max = data['\0'];

for (int i = 0; i < 256; i++)

{

if (data[(char)i] > max)

{

max = data[(char)i];

}

}

return max;

}

long countTotal(std::unordered\_map<int, int> data)

{

long total = 0;

for (int i = 0; i < 256; i++)

{

total += data[i];

}

return total;

}

long countTotal(std::unordered\_map<char, int> data)

{

long total = 0;

for (int i = 0; i < 256; i++)

{

total += data[(char)i];

}

return total;

}

std::string get\_text(std::string path)

{

std::string data = "";

std::ifstream in(path);

if (in.is\_open())

{

std::string temp;

while (std::getline(in, temp))

{

data += temp;

}

}

return data;

}

void text\_to\_diagram(std::string path, std::unordered\_map<char, int>& data)

{

std::string text = get\_text(path);

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

{

data[text[i]]++;

}

}

void ReadBMP(std::string filename, std::unordered\_map<int, int>& R, std::unordered\_map<int, int>& G, std::unordered\_map<int, int>& B)

{

R = init\_map\_RGB();

G = init\_map\_RGB();

B = init\_map\_RGB();

std::ifstream in(filename, std::ios\_base::binary);

if (!in.is\_open())

throw "Argument Exception";

unsigned char info[54]; //Массив заголовка файла bmp

in.read((char\*)info, sizeof(char) \* 54); // Чтение в массив заголовка файла

int width = \*(int\*)&info[18]; // Получение ширины изображения из массив

int height = \*(int\*)&info[22]; // Получение высоты изображения из массив

int row\_padded = (width \* 3 + 3) & (~3);

unsigned char\* data = new unsigned char[row\_padded];

char tmp;

std::string result = "";

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

{

in.read((char\*)data, sizeof(char) \* row\_padded);

for (int j = 0; j < width \* 3; j += 3)

{

tmp = data[j];

data[j] = data[j + 2];

data[j + 2] = tmp;

R[(int)data[j]]++;

G[(int)data[j + 1]]++;

B[(int)data[j + 2]]++;

}

}

in.close();

}

float entropy(std::unordered\_map<int, int> data)

{

long total = countTotal(data);

float result = 0.0;

for (int i = 0; i < 256; i++)

{

if (data[i] != 0)

{

double posibility = (double)data[i] / (double)total;

result += posibility \* log2(1 / posibility);

}

}

return round(result \* 1000) / 1000;

}

float entropy(std::unordered\_map<char, int> data)

{

long total = countTotal(data);

float result = 0.0;

for (int i = 0; i < 256; i++)

{

if (data[(char)i] != 0)

{

double posibility = (double)data[(char)i] / (double)total;

result += posibility \* log2(1 / posibility);

}

}

return round(result \* 1000) / 1000;

}

Листинг файла MyForm.cpp

#pragma optimize("agi", off)

#include "MyForm.h"

#include <Windows.h>

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

int main()

{

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

FreaquencyAnalisys::MyForm form;

Application::Run(% form);

}

Листинг файла MyForm.h

#pragma once

#include "Analisys.h"

#include <msclr\marshal\_cppstd.h>

std::unordered\_map<int, int> R, G, B;

namespace FreaquencyAnalisys {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

using namespace System::Windows::Forms::DataVisualization::Charting;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

openFileDialog1->Filter = "Текстовый файл (\*.txt)|\*.txt|Изображение (\*.bmp)|\*.bmp";

this->chart1->Series["Series1"]->Points->AddXY(0, 0);

this->chart1->ChartAreas[0]->AxisX->ScaleView->Zoomable = true;

this->chart1->ChartAreas[0]->AxisY->ScaleView->Zoomable = true;

this->trackBar1->Value = 5;

this->trackBar2->Value = 10;

this->chart1->Series->Add("R");

this->chart1->Series["R"]->Color = System::Drawing::Color::Red;

this->chart1->Series->Add("G");

this->chart1->Series["G"]->Color = System::Drawing::Color::Green;

this->chart1->Series->Add("B");

this->chart1->Series["B"]->Color = System::Drawing::Color::Blue;

this->скрытьКрасныйRКаналToolStripMenuItem->Visible = false;

this->скрытьЗелёныйGКаналToolStripMenuItem->Visible = false;

this->скрытьСинийBКаналToolStripMenuItem->Visible = false;

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart1;

private: System::Windows::Forms::MenuStrip^ menuStrip1;

private: System::Windows::Forms::ToolStripMenuItem^ файлToolStripMenuItem;

private: System::Windows::Forms::OpenFileDialog^ openFileDialog1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::TrackBar^ trackBar1;

private: System::Windows::Forms::TrackBar^ trackBar2;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::Label^ label7;

private: System::Windows::Forms::ToolStripMenuItem^ закрытьПриложениеToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ настройкаГистрограммыToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ выключитьСеткуToolStripMenuItem;

private: int max = 0;

private: System::Windows::Forms::ToolStripMenuItem^ скрытьКрасныйRКаналToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ скрытьЗелёныйGКаналToolStripMenuItem;

private: System::Windows::Forms::ToolStripMenuItem^ скрытьСинийBКаналToolStripMenuItem;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::Windows::Forms::DataVisualization::Charting::ChartArea^ chartArea1 = (gcnew System::Windows::Forms::DataVisualization::Charting::ChartArea());

System::Windows::Forms::DataVisualization::Charting::Legend^ legend1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Legend());

System::Windows::Forms::DataVisualization::Charting::Series^ series1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());

this->chart1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Chart());

this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());

this->файлToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->настройкаГистрограммыToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->выключитьСеткуToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->скрытьКрасныйRКаналToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->скрытьЗелёныйGКаналToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->скрытьСинийBКаналToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->закрытьПриложениеToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->openFileDialog1 = (gcnew System::Windows::Forms::OpenFileDialog());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->trackBar1 = (gcnew System::Windows::Forms::TrackBar());

this->trackBar2 = (gcnew System::Windows::Forms::TrackBar());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->label7 = (gcnew System::Windows::Forms::Label());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))->BeginInit();

this->menuStrip1->SuspendLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->BeginInit();

this->SuspendLayout();

//

// chart1

//

this->chart1->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

chartArea1->AlignmentOrientation = System::Windows::Forms::DataVisualization::Charting::AreaAlignmentOrientations::None;

chartArea1->AxisX->Maximum = 255;

chartArea1->AxisY->IsLabelAutoFit = false;

chartArea1->Name = L"ChartArea1";

this->chart1->ChartAreas->Add(chartArea1);

legend1->AutoFitMinFontSize = 14;

legend1->Enabled = false;

legend1->Name = L"Legend1";

this->chart1->Legends->Add(legend1);

this->chart1->Location = System::Drawing::Point(618, 42);

this->chart1->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->chart1->Name = L"chart1";

series1->ChartArea = L"ChartArea1";

series1->Legend = L"Legend1";

series1->LegendText = L"Количесвто символов в тексте";

series1->Name = L"Series1";

this->chart1->Series->Add(series1);

this->chart1->Size = System::Drawing::Size(1278, 1003);

this->chart1->TabIndex = 0;

this->chart1->Text = L"chart1";

//

// menuStrip1

//

this->menuStrip1->GripMargin = System::Windows::Forms::Padding(2, 2, 0, 2);

this->menuStrip1->ImageScalingSize = System::Drawing::Size(24, 24);

this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(3) {

this->файлToolStripMenuItem,

this->настройкаГистрограммыToolStripMenuItem, this->закрытьПриложениеToolStripMenuItem

});

this->menuStrip1->Location = System::Drawing::Point(0, 0);

this->menuStrip1->Name = L"menuStrip1";

this->menuStrip1->Size = System::Drawing::Size(1896, 33);

this->menuStrip1->TabIndex = 1;

this->menuStrip1->Text = L"menuStrip1";

//

// файлToolStripMenuItem

//

this->файлToolStripMenuItem->Name = L"файлToolStripMenuItem";

this->файлToolStripMenuItem->Size = System::Drawing::Size(142, 29);

this->файлToolStripMenuItem->Text = L"Выбрать файл";

this->файлToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::файлToolStripMenuItem\_Click);

//

// настройкаГистрограммыToolStripMenuItem

//

this->настройкаГистрограммыToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {

this->выключитьСеткуToolStripMenuItem,

this->скрытьКрасныйRКаналToolStripMenuItem, this->скрытьЗелёныйGКаналToolStripMenuItem, this->скрытьСинийBКаналToolStripMenuItem

});

this->настройкаГистрограммыToolStripMenuItem->Name = L"настройкаГистрограммыToolStripMenuItem";

this->настройкаГистрограммыToolStripMenuItem->Size = System::Drawing::Size(229, 29);

this->настройкаГистрограммыToolStripMenuItem->Text = L"Настройка гистограммы";

//

// выключитьСеткуToolStripMenuItem

//

this->выключитьСеткуToolStripMenuItem->Name = L"выключитьСеткуToolStripMenuItem";

this->выключитьСеткуToolStripMenuItem->Size = System::Drawing::Size(326, 34);

this->выключитьСеткуToolStripMenuItem->Text = L"Отключить сетку";

this->выключитьСеткуToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::выключитьСеткуToolStripMenuItem\_Click);

//

// скрытьКрасныйRКаналToolStripMenuItem

//

this->скрытьКрасныйRКаналToolStripMenuItem->Name = L"скрытьКрасныйRКаналToolStripMenuItem";

this->скрытьКрасныйRКаналToolStripMenuItem->Size = System::Drawing::Size(326, 34);

this->скрытьКрасныйRКаналToolStripMenuItem->Text = L"Скрыть красный (R) канал";

this->скрытьКрасныйRКаналToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::скрытьКрасныйRКаналToolStripMenuItem\_Click);

//

// скрытьЗелёныйGКаналToolStripMenuItem

//

this->скрытьЗелёныйGКаналToolStripMenuItem->Name = L"скрытьЗелёныйGКаналToolStripMenuItem";

this->скрытьЗелёныйGКаналToolStripMenuItem->Size = System::Drawing::Size(326, 34);

this->скрытьЗелёныйGКаналToolStripMenuItem->Text = L"Скрыть зелёный (G) канал";

this->скрытьЗелёныйGКаналToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::скрытьЗелёныйGКаналToolStripMenuItem\_Click);

//

// скрытьСинийBКаналToolStripMenuItem

//

this->скрытьСинийBКаналToolStripMenuItem->Name = L"скрытьСинийBКаналToolStripMenuItem";

this->скрытьСинийBКаналToolStripMenuItem->Size = System::Drawing::Size(326, 34);

this->скрытьСинийBКаналToolStripMenuItem->Text = L"Скрыть синий (B) канал";

this->скрытьСинийBКаналToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::скрытьСинийBКаналToolStripMenuItem\_Click);

//

// закрытьПриложениеToolStripMenuItem

//

this->закрытьПриложениеToolStripMenuItem->Name = L"закрытьПриложениеToolStripMenuItem";

this->закрытьПриложениеToolStripMenuItem->Size = System::Drawing::Size(203, 29);

this->закрытьПриложениеToolStripMenuItem->Text = L"Закрыть приложение";

this->закрытьПриложениеToolStripMenuItem->Click += gcnew System::EventHandler(this, &MyForm::закрытьПриложениеToolStripMenuItem\_Click);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(20, 43);

this->label1->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(235, 20);

this->label1->TabIndex = 2;

this->label1->Text = L"Шаг по оси X (номер символа)";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(20, 140);

this->label2->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(286, 20);

this->label2->TabIndex = 3;

this->label2->Text = L"Шаг по оси Y (количество символов)";

//

// trackBar1

//

this->trackBar1->Location = System::Drawing::Point(24, 68);

this->trackBar1->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->trackBar1->Name = L"trackBar1";

this->trackBar1->Size = System::Drawing::Size(585, 69);

this->trackBar1->TabIndex = 4;

this->trackBar1->Scroll += gcnew System::EventHandler(this, &MyForm::trackBar1\_Scroll);

//

// trackBar2

//

this->trackBar2->Location = System::Drawing::Point(24, 165);

this->trackBar2->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->trackBar2->Name = L"trackBar2";

this->trackBar2->Size = System::Drawing::Size(585, 69);

this->trackBar2->TabIndex = 5;

this->trackBar2->Scroll += gcnew System::EventHandler(this, &MyForm::trackBar2\_Scroll);

//

// textBox1

//

this->textBox1->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left));

this->textBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 14.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox1->Location = System::Drawing::Point(24, 272);

this->textBox1->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->textBox1->Multiline = true;

this->textBox1->Name = L"textBox1";

this->textBox1->ReadOnly = true;

this->textBox1->ScrollBars = System::Windows::Forms::ScrollBars::Vertical;

this->textBox1->Size = System::Drawing::Size(583, 755);

this->textBox1->TabIndex = 6;

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(33, 117);

this->label3->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(18, 20);

this->label3->TabIndex = 7;

this->label3->Text = L"1";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(33, 214);

this->label4->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(18, 20);

this->label4->TabIndex = 8;

this->label4->Text = L"1";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(570, 214);

this->label5->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(36, 20);

this->label5->TabIndex = 9;

this->label5->Text = L"100";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Location = System::Drawing::Point(578, 117);

this->label6->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(27, 20);

this->label6->TabIndex = 10;

this->label6->Text = L"20";

//

// label7

//

this->label7->AutoSize = true;

this->label7->Location = System::Drawing::Point(20, 248);

this->label7->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

this->label7->Name = L"label7";

this->label7->Size = System::Drawing::Size(323, 20);

this->label7->TabIndex = 11;

this->label7->Text = L"Количество вхождений данного символа";

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(9, 20);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(1896, 1048);

this->Controls->Add(this->label7);

this->Controls->Add(this->label6);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->trackBar2);

this->Controls->Add(this->trackBar1);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->chart1);

this->Controls->Add(this->menuStrip1);

this->MainMenuStrip = this->menuStrip1;

this->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);

this->MaximumSize = System::Drawing::Size(2869, 1631);

this->MinimumSize = System::Drawing::Size(1906, 998);

this->Name = L"MyForm";

this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;

this->Text = L"Частотный анализатор";

this->WindowState = System::Windows::Forms::FormWindowState::Maximized;

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))->EndInit();

this->menuStrip1->ResumeLayout(false);

this->menuStrip1->PerformLayout();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->trackBar2))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void файлToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (openFileDialog1->ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

this->chart1->Series["Series1"]->Points->Clear();

this->chart1->Series["R"]->Points->Clear();

this->chart1->Series["G"]->Points->Clear();

this->chart1->Series["B"]->Points->Clear();

if (openFileDialog1->FileName->Contains(".txt"))

text();

else if (openFileDialog1->FileName->Contains(".bmp"))

{

image();

this->скрытьКрасныйRКаналToolStripMenuItem->Text = "Скрыть красный (R) канал";

this->скрытьЗелёныйGКаналToolStripMenuItem->Text = "Скрыть зелёный (G) канал";

this->скрытьСинийBКаналToolStripMenuItem->Text = "Скрыть синий (B) канал";

}

}

}

private: void text()

{

this->скрытьКрасныйRКаналToolStripMenuItem->Visible = false;

this->скрытьЗелёныйGКаналToolStripMenuItem->Visible = false;

this->скрытьСинийBКаналToolStripMenuItem->Visible = false;

std::unordered\_map<char, int> data;

String^ path = System::Convert::ToString(openFileDialog1->FileName);

text\_to\_diagram(msclr::interop::marshal\_as<std::string>(path), data);

int max\_value = find\_max(data);

max = max\_value;

this->chart1->ChartAreas["ChartArea1"]->AxisY->Maximum = max\_value + (max\_value \* 0.1);

this->chart1->ChartAreas["ChartArea1"]->AxisX->Maximum = 255;

this->trackBar1->Value = 5;

this->trackBar2->Value = 10;

this->chart1->ChartAreas["ChartArea1"]->AxisY->Interval = this->trackBar2->Value \* (max / 100);

this->chart1->ChartAreas["ChartArea1"]->AxisX->Interval = this->trackBar1->Value;

this->trackBar2->Maximum = 20;

this->trackBar2->Minimum = 1;

this->trackBar1->Maximum = 20;

this->trackBar1->Minimum = 1;

this->label4->Text = Convert::ToString(max / 20);

this->label5->Text = Convert::ToString(Convert::ToInt32(this->label4->Text) \* 20);

for (int i = 255; i > 0; i--)

{

if (data[(char)i] > -1)

this->chart1->Series["Series1"]->Points->AddXY(i, data[(char)i]);

}

this->textBox1->Text = gcnew String(output\_acsii(data).c\_str()) + "\r\nЭнтропия файла: " + Convert::ToString(entropy(data));

}

private: void image()

{

this->скрытьКрасныйRКаналToolStripMenuItem->Visible = true;

this->скрытьЗелёныйGКаналToolStripMenuItem->Visible = true;

this->скрытьСинийBКаналToolStripMenuItem->Visible = true;

String^ path = System::Convert::ToString(openFileDialog1->FileName);

ReadBMP(msclr::interop::marshal\_as<std::string>(path), R, G, B);

int max\_value = find\_max(R);

if (find\_max(G) > max\_value)

max\_value = find\_max(G);

if (find\_max(B) > max\_value)

max\_value = find\_max(B);

max = max\_value;

this->chart1->ChartAreas["ChartArea1"]->AxisY->Maximum = max\_value + (max\_value \* 0.1);

this->chart1->ChartAreas["ChartArea1"]->AxisX->Maximum = 255;

this->trackBar1->Value = 5;

this->trackBar2->Value = 10;

this->label4->Text = Convert::ToString(max / 20);

this->label5->Text = Convert::ToString(Convert::ToInt32(this->label4->Text) \* 20);

this->chart1->ChartAreas["ChartArea1"]->AxisY->Interval = this->trackBar2->Value \* (max / 20);

this->chart1->ChartAreas["ChartArea1"]->AxisX->Interval = this->trackBar1->Value;

this->trackBar2->Maximum = 20;

this->trackBar2->Minimum = 1;

this->trackBar1->Maximum = 20;

this->trackBar1->Minimum = 1;

std::unordered\_map<int, int> data = init\_map\_RGB();

for (int i = 255; i > 0; i--)

{

int temp = 0;

if (R[i] > -1)

{

this->chart1->Series["R"]->Points->AddXY(i, R[i]);

temp += R[i];

}

if (G[i] > -1)

{

this->chart1->Series["G"]->Points->AddXY(i, G[i]);

temp += G[i];

}

if (B[i] > -1)

{

this->chart1->Series["B"]->Points->AddXY(i, B[i]);

temp += B[i];

}

data[i] = temp;

}

String^ result = "";

for (int i = 0; i < 256; i++)

{

result += i + ". " + data[i] + "\r\n";

}

this->textBox1->Text = result + "\r\nЭнтропия файла: " + Convert::ToString(entropy(data));

}

private: System::Void trackBar2\_Scroll(System::Object^ sender, System::EventArgs^ e)

{

this->chart1->ChartAreas["ChartArea1"]->AxisY->Interval = this->trackBar2->Value \* (max / 20);

}

private: System::Void trackBar1\_Scroll(System::Object^ sender, System::EventArgs^ e)

{

this->chart1->ChartAreas["ChartArea1"]->AxisX->Interval = this->trackBar1->Value;

}

private: System::Void закрытьПриложениеToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (MessageBox::Show("Вы уверены, что хотите закрыть приложение?", "Закрыть приложение", MessageBoxButtons::OKCancel) == System::Windows::Forms::DialogResult::OK)

this->Close();

}

private: System::Void выключитьСеткуToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (this->chart1->ChartAreas["ChartArea1"]->AxisX->MajorGrid->Enabled == false)

{

this->выключитьСеткуToolStripMenuItem->Text = "Отключить сетку";

this->chart1->ChartAreas["ChartArea1"]->AxisX->MajorGrid->Enabled = true;

this->chart1->ChartAreas["ChartArea1"]->AxisY->MajorGrid->Enabled = true;

}

else

{

this->выключитьСеткуToolStripMenuItem->Text = "Включить сетку";

this->chart1->ChartAreas["ChartArea1"]->AxisX->MajorGrid->Enabled = false;

this->chart1->ChartAreas["ChartArea1"]->AxisY->MajorGrid->Enabled = false;

}

}

private: System::Void очиститьКоординатнуюПлоскостьToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

this->chart1->Series["Series1"]->Points->Clear();

this->textBox1->Text = "";

this->chart1->Series["R"]->Points->Clear();

this->chart1->Series["G"]->Points->Clear();

this->chart1->Series["B"]->Points->Clear();

this->chart1->Series["Series1"]->Points->AddXY(0, 0);

}

private: System::Void скрытьКрасныйRКаналToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (this->скрытьКрасныйRКаналToolStripMenuItem->Text == "Скрыть красный (R) канал")

{

this->скрытьКрасныйRКаналToolStripMenuItem->Text = "Показать красный (R) канал";

this->chart1->Series["R"]->Points->Clear();

this->chart1->Series["Series1"]->Points->AddXY(0, 0);

}

else

{

this->скрытьКрасныйRКаналToolStripMenuItem->Text = "Скрыть красный (R) канал";

for (int i = 255; i > 0; i--)

{

this->chart1->Series["R"]->Points->AddXY(i, R[i]);

}

}

}

private: System::Void скрытьЗелёныйGКаналToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (this->скрытьЗелёныйGКаналToolStripMenuItem->Text == "Скрыть зелёный (G) канал")

{

this->скрытьЗелёныйGКаналToolStripMenuItem->Text = "Показать зелёный (G) канал";

this->chart1->Series["G"]->Points->Clear();

this->chart1->Series["Series1"]->Points->AddXY(0, 0);

}

else

{

this->скрытьЗелёныйGКаналToolStripMenuItem->Text = "Скрыть зелёный (G) канал";

for (int i = 255; i > 0; i--)

{

this->chart1->Series["G"]->Points->AddXY(i, G[i]);

}

}

}

private: System::Void скрытьСинийBКаналToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (this->скрытьСинийBКаналToolStripMenuItem->Text == "Скрыть синий (B) канал")

{

this->скрытьСинийBКаналToolStripMenuItem->Text = "Показать синий (B) канал";

this->chart1->Series["B"]->Points->Clear();

this->chart1->Series["Series1"]->Points->AddXY(0, 0);

}

else

{

this->скрытьСинийBКаналToolStripMenuItem->Text = "Скрыть синий (B) канал";

for (int i = 255; i > 0; i--)

{

this->chart1->Series["B"]->Points->AddXY(i, B[i]);

}

}

}

};

}

Приложение Б

Текст из учебника:

1.1. Зачем изучать алгоритмы?

Позвольте мне сначала обосновать актуальность этой книги, указав несколько

причин, почему так важно быть мотивированным в изучении алгоритмов. Так

что же такое алгоритм, в конце концов? Это набор четко сформулированных

правил, в сущности, рецепт для решения некоторой вычислительной задачи.

Может быть, у вас имеется куча чисел и вы хотите перераспределить их так,

чтобы расположить их в отсортированном порядке. Возможно, у вас есть

дорожная карта и необходимо вычислить кратчайший путь от некоторой

исходной точки до определенного места назначения. Вполне вероятно, вам

требуется выполнить несколько задач до наступления установленных сроков,

в таком случае вы заинтересованы в упорядочении выполнения этих задач,

для того чтобы вовремя все их завершить.

Итак, зачем же изучать алгоритмы?

Важность для всех отраслей computer science. Во-первых, понимание основ

алгоритмизации и тесно с ней взаимосвязанной сферы организации структур

данных необходимо для выполнения серьезной работы практически в любой

отрасли информатики. Например, в Стэнфордском университете для полу-

1.1. Зачем изучать алгоритмы? 23

чения любой степени по Computer Science (будь то бакалавр, магистр наук

и даже доктор наук) обязательно требуется пройти курс алгоритмов. Приведу

всего несколько примеров:

1. Протоколы маршрутизации в коммуникационных сетях задействуют

классические алгоритмы поиска кратчайшего пути.

2. Криптография с открытым ключом опирается на эффективные теоретико-числовые алгоритмы.

3. Компьютерная графика требует вычислительных примитивов, предоставляемых геометрическими алгоритмами.

4. Индексация в базах данных опирается на структуры данных сбалансированных деревьев поиска.

5. Вычислительная биология использует алгоритмы динамического программирования для измерения сходства геномов.

Приведенный список можно продолжать.

Двигатель технологических инноваций. Во-вторых, алгоритмы играют

ключевую роль в современных технологических инновациях. Приведу только один очевидный пример: поисковые системы используют целую мозаику

алгоритмов для эффективного вычисления релевантности различных вебстраниц заданному поисковому запросу. Наиболее известным подобным

алгоритмом является алгоритм PageRank, используемый в настоящее время

Google. В самом деле, в докладе за декабрь 2010 года для Белого дома

США президентский консультативный совет на науке и технике написал

следующее:

«Все знают Закон Мура — предсказание, сделанное в 1965 г. соучредителем Intel Гордоном Муром о том, что плотность транзисторов

в интегральных схемах будет удваиваться каждые 1–2 года... Во многих

областях прирост производительности за счет улучшения алгоритмов

значительно превысил даже впечатляющий рост производительности

за счет увеличения скорости процессоров»1

.

1 Выдержка из доклада президенту и конгрессу: Проектирование цифрового будущего, декабрь 2010 г. (с. 71).

24 Глава 1. Введение

Новый взгляд на достижения других наук. В-третьих (хотя это выходит

за рамки этой книги), алгоритмы все более и более используются в качестве

«увеличительного стекла», чтобы по-новому взглянуть на научные проблемы

за пределами computer science и IT. Например, исследование квантовых вычислений обеспечило новый вычислительный взгляд на квантовую механику.

Ценовые колебания на экономических рынках могут плодотворно рассматриваться как алгоритмический процесс. Даже эволюцию можно рассматривать

как удивительно эффективный алгоритм поиска.

Гимнастика для мозга. Во времена студенчества моими любимыми всегда

были сложные предметы. После их освоения всегда оставалось ощущение,

что я становился на несколько пунктов IQ умнее, чем в начале изучения. Надеюсь, моя книга позволит вам получить аналогичный опыт.

Удовольствие! Я надеюсь, что к концу этой книги вы поймете, почему разработка и анализ алгоритмов просто приносит удовольствие! Это увлекательное

занятие, которое требует редкого сочетания точности и креативности. Разумеется, временами эта работа может разочаровывать, но она также очень

затягивает. И давайте не будем забывать о том, что обучение алгоритмам, по

сути, сопровождает вас с самого детства.

1.2. Целочисленное умножение

1.2.1. Задачи и решения

Когда вы учились в начальной школе, вы, скорее всего, изучали умножение

двух чисел в столбик, что, по сути, является алгоритмом — четко сформулированным набором правил для преобразования входа (два числа) в выход (их

произведение). Всегда важно понимать разницу между постановкой (описанием) решаемой задачи и описанием метода ее решения (то есть алгоритма

этой задачи). В этой книге мы будем систематически придерживаться схемы,

в которой сначала производится постановка вычислительной задачи (данные

на входе и требуемый результат на выходе), а затем дается описание одного

или нескольких алгоритмов ее решения.

1.2. Целочисленное умножение 25

1.2.2. Задача целочисленного умножения

В задаче целочисленного умножения входными данными являются два

n-разрядных числа, обозначим их x и y. Разрядность (длина) n чисел x и y

может быть любым положительным целым числом. Однако я призываю вас

оперировать большими значениями n, в тысячах или более1

. (Представьте,

что вы разрабатываете некое криптографическое приложение, ведь они манипулируют очень большими числами.) В задаче целочисленного умножения

требуемым выходом является результат произведения x ? y.

ЗАДАЧА. ЦЕЛОЧИСЛЕННОЕ УМНОЖЕНИЕ

Вход: два n-значных неотрицательных целых числа, x и y.

Выход: произведение x ? y.

1.2.3. Алгоритм начальной школы

Точно определив вычислительную задачу, мы опишем алгоритм, который

ее решает, — тот самый алгоритм, который вы изучали в начальной школе.

Мы оценим производительность этого алгоритма числом «примитивных

операций», которые он выполняет, в виде функции от количества знаков n

в каждом входном числе. Пока же давайте представим примитивную операцию как любую из следующих: (i) сложение двух одноразрядных (n = 1)

чисел; (ii) умножение двух одноразрядных чисел или (iii) добавление нуля

к началу или концу числа.

Чтобы освежить вашу память, рассмотрим конкретный пример умножения

x = 5678 на y = 1234 (здесь n = 4) в столбик, см. рис. 1.1. Сначала алгоритм вы1 Если вы хотите перемножить числа с разными длинами (например, 1234 и 56), просто добавьте несколько нулей в начало меньшего числа (например, рассматривайте

56 как 0056). С другой стороны, алгоритмы, которые мы обсудим, могут быть приспособлены для чисел с разной длиной.

26 Глава 1. Введение

числяет «частичное произведение» первого числа и последней цифры второго

числа: 5678 ? 4 = 22 712. Вычисление этого частичного произведения сводится

к умножению каждой цифры первого числа на 4, записи младшего разряда

результата, запоминанию («переносу» на следующий этап) старшего разряда

и добавлению этих «переносов» (если они есть) на следующем умножении1

.

При вычислении следующего частичного произведения (5678 ? 3 = 17 034)

мы делаем то же самое, сдвигая результат на один знак влево (фактически

добавляя «0» в конце). И так далее для оставшихся двух частичных произведений. Заключительный шаг состоит в том, чтобы сложить все частичные

произведения.

Тогда в третьем классе вы, вероятно, согласились, что этот алгоритм является

правильным, имея в виду, что неважно, с каких чисел x и y начинать. При

условии, что все промежуточные вычисления выполняются правильно, алгоритм в конечном итоге заканчивается получением результата произведения

x ? y двух исходных чисел.

Таким образом, вы никогда не получите неправильный ответ, и алгоритм не

может зациклиться.

1 8 ? 4 = 32, 2 пишем, 3 «переносим», 7 ? 4 = 28, плюс 3 равняется 31, 1 пишем,

3 переносим, и так далее...

1.2. Целочисленное умножение 27

1.2.4. Анализ числа операций

Ваш школьный учитель, возможно, не обсуждал число примитивных операций, необходимых для завершения процедуры умножения в столбик. В нашем примере, для того чтобы вычислить первое частичное произведение,

мы умножили 4 раза каждую из цифр 5, 6, 7, 8 первого числа. Это четыре

примитивные операции. Мы также выполнили несколько сложений из-за переносов. В общем случае вычисление частичного произведения влечет за собой

n умножений (одно умножение на один знак) и не более n сложений (не более

одного на один знак). Всего получается не более 2n примитивных операций.

Первое частичное произведение ничем не отличается от других, и каждое из

них требует не более 2n операций. Поскольку имеется n частичных произведений — по одному на каждый знак второго числа, — вычисление всех из

них требует не более n ? 2n = 2n2

примитивных операций. Чтобы вычислить

окончательный ответ, нам все еще нужно все частичные произведения сложить

вместе, но для этого требуется сопоставимое число операций (не более чем

еще 2n2

). Подведем итоги:

2

=4

общее количество операций ? ? констант

14243а n .

Рассуждая о том, каким образом объем работы, который этот алгоритм выполняет, возрастает по мере того, как исходные множители становятся все

длиннее и длиннее, мы видим, что объем выполняемых операций растет

квадратически, следуя за увеличением разрядности. Если удвоить длину исходных множителей, то требуемый объем операций подскакивает в 4 раза.

Увеличьте их длину в 4 раза, и она подскочит в 16 раз, и так далее.

1.2.5. Можно ли добиться лучшего?

В зависимости от того, каким третьеклассником вы были, вы вполне могли

принять эту процедуру как уникальный или, по крайней мере, оптимальный способ умножения двух чисел. Если вы захотите стать серьезным

проектировщиком алгоритмов, то нужно будет преодолеть эту робость.

Классическая книга по алгоритмам Ахо, Хопкрофта и Ульмана, после

итеративного рассмотрения целого ряда методик разработки алгоритмов,

говорит следующее:

28 Глава 1. Введение

«Для хорошего разработчика алгоритмов, пожалуй, самый важный принцип состоит в том, чтобы отказаться от соглашательства»1

.

Или как я люблю говорить, что каждый разработчик алгоритмов должен

принять как должное:

Можно ли добиться лучшего?

Этот вопрос особенно актуален, когда вы сталкиваетесь с очевидным или

прямым решением вычислительной задачи. В третьем классе вы, возможно,

не задавались вопросом о том, можно ли выстроить путь решения лучше, чем

это делает простой классический алгоритм умножения в столбик. Настало

время задать этот вопрос и дать на него ответ.

Текст статьи в СМИ:

Глава ЦБ РФ Эльвира Набиуллина заявила, что жесткая денежно-кредитная политика, то есть удержание ключевой ставки на высоком уровне, это не прихоть регулятора, а неизбежная реакция на то, что происходит в экономике. Она не противоречит задачам правительства и делает всё, чтобы «усилия не ушли в свисток, не ушли в инфляцию, а работали именно на рост экономики».

Эксперт объяснил причины жесткой политики Банка России и очертил границы ее влияния на инфляцию

Об этом председатель Банка России сообщила на днях, выступая на заседании комитета по финансовому рынку в Госдуме. Между тем, Росстат накануне оценил уровень годовой инфляции в 8,63%. Почему рост цен не удается затормозить, несмотря на все усилия Банка России, «МК» выяснил у начальника аналитического управления банка БКФ Максима Осадчего.

— Ключевую ставку 25 октября снова повысили, причём сразу до 21%. Сейчас она более высокая, чем была в конце февраля и в марте 2022 года, то есть сразу после начала СВО, когда Запад ввел против России первые «ядерные санкции». Неужели сейчас ситуация в экономике хуже, чем была в начале весны 2022 года?

— Причины повышения ключевой ставки два года назад и сейчас совершенно разные. Об этом говорила глава ЦБ РФ на пресс-конференции 25 октября. Сразу после начала СВО ЦБ резко ужесточил денежно-кредитную политику, увеличил ключевую ставку более чем в два раза, с 9,5% до 20%, для того, чтобы прекратить набег вкладчиков на банки, чтобы остановить массовый отток денег с вкладов и обвал рубля, чтобы избежать полномасштабного финансового кризиса. Как только эта угроза миновала, регулятор начал быстро снижать ключевую ставку. Рыночную экономику удалось быстро и эффективно стабилизировать методом, описываемым в учебниках макроэкономики.

Сейчас ситуация совершенно иная. Нет набега вкладчиков на банки, вместо массового оттока вкладов наблюдаем массовый приток, рубль хоть и слабеет, но не обваливается. Но есть проблема: несмотря на то, что ЦБ РФ повысил ключевую ставку уже до 21% и сигнализирует о возможности дальнейшего ее повышения, инфляция не сдается, а угроза кризиса нарастает.

— Почему ключевая ставка превышает инфляцию более чем в два раза, однако инфляция не снижается?

— Действительно, ключевая ставка достигла уровня 21%, а годовая инфляция на 28 октября, по данным Росстата, составляет 8,6%, тогда как ее целевой уровень — 4%. В январе инфляция была 7,44% при ключевой ставке 16%. И как бы ни повышалась ключевая ставка, целевой уровень в 4%, как заколдованный, остается недостижимым. Близок локоток, да не укусишь.

Причина такого «упрямства» очевидна: жесткая политика ЦБ РФ противостоит мягкой политике Минфина. И проинфляционный эффект мягкой политики Минфина оказывается сильнее, чем контрифляционное маневрирование Банка России.

— Что вы имеете в виду?

— Чтобы пояснить, приведу аналогию. Можно сравнить экономику в условиях геополитического конфликта с кораблем. В пробоины днища хлещет вода, трюмные помпы раскалены докрасна, но вода поступает быстрее.

Роль трюмной помпы (осушительного насоса) как раз и играет ключевая ставка: чем она выше, тем больше ставки по вкладам и тем сильнее приток средств населения в банки. Таким образом, происходит «стерилизация ликвидности» или, проще говоря, выкачивание ликвидности (наличных денег) из карманов вкладчиков и из-под их подушек на депозиты банков. Для особо въедливых: остатки на дебетовых (например, зарплатных) картах мало чем отличаются от наличных денег. Чем меньше наличных у населения, тем ниже спрос на товары и, соответственно, тем слабее инфляционное давление.

А вот роль пробоин играют всяческие способы искусственного закачивания в экономику государственных средств, например, в форме субсидий. Одну такую пробоину наконец-то закрыли — льготную ипотеку, но подобных «дыр» — без счета, и не только для населения, но и для предприятий.

3 материала по теме

Проявлением этого искусственного закачивания в экономику государственных средств является огромный дефицит бюджета, который в 2022 году составил 3,3 трлн рублей, в 2023 году — 3,2 трлн, а в 2024 году, по уточненной в сентябре оценке Минфина, составит опять 3,3 трлн рублей.

Если бы не героическая борьба ЦБ с гидрой инфляции, то приток ликвидности затопил бы экономику, и было бы, как в Турции. Там годовая инфляция в сентябре составила 49,38%.

— Но там и ключевая ставка 50%, которую регулятор сохранил без изменений уже семь раз подряд. Правда, и у нас ещё не вечер. В декабре Совет директоров Банка России проведет ещё одно заседание. Ключевая ставка снова может быть повышена до конца года?

— Так как принципиальных решений по преодолению наращивания дефицита бюджета в России не видно, то и рост ключевой ставки может продолжиться. Даже на последнем в этом году заседании Совета директоров ЦБ РФ 20 декабря ключевую ставку могут снова повысить. Эльвира Набиуллина в своем заявлении 25 октября подчеркнула: чтобы сдержать ускорившийся рост цен, потребуется значительно бо?льшая жесткость денежно-кредитной политики в следующем году. По её словам, «более высокая траектория ключевой ставки должна вернуть инфляцию к цели».

Еще в январе, если судить по кривой бескупонной доходности государственных облигаций, рынок ожидал снижения ключевой ставки уже через квартал, а в настоящее время ожидания сместились на срок один год.

— По словам замглавы Минфина Алексея Моисеева, Центробанк окончательно пройдет путь по смягчению денежно-кредитной политики, то есть снизит ключевую ставку до однозначных значений, только до 2030 года. Из этого можно сделать вывод, что в правительстве понимают политику регулятора?

— О серьезности ситуации говорит то обстоятельство, что сразу после повышения ключевой ставки, в понедельник, 28 октября, состоялось Совещание по экономическим вопросам, которое провел президент РФ в режиме видеоконференции. Отдельно обсуждался вопрос о корпоративном кредитовании. Во вступительном слове Владимир Путин отметил, что задача правительства «на ближайшую, и на более отдалённую перспективу — развивать экономику предложения, увеличивать объём выпуска товаров и услуг именно у нас, в России, запускать новые инвестиционные проекты и создавать современные рабочие места во всех отраслях и секторах экономики». Тезис об «экономике предложения» можно интерпретировать как сохранение политики стимулирования экономики с помощью госсубсидий. Примечтаельно, что первой после вступительного слова президента выступала председатель ЦБ РФ. Стенограмма ее выступления отсутствует, но понятно, что главе Банка России Эльвире Набиуллиной пришлось отстаивать необходимость проведения жесткой денежно-кредитной политики.

— Что можно посоветовать россиянам в таких условиях?

— Сейчас максимальные ставки по депозитам достигают 26%, но ещё не вечер: в ближайшие месяцы они могут превысить и 30%. Если открывать вклад с фиксированной ставкой, то сроком не более, чем на полгода. А лучше искать вклады с плавающей ставкой, привязанной, например, к ключевой ставке. В первую очередь, накопительные счета, позволяющие относительно свободно маневрировать ликвидностью без потери процентов. Также помнить о том, что перед новогодними праздниками, как правило, банки проводят сезонные кампании — предлагают вклады с повышенными ставками. На валютные вклады, в том числе и в юанях даже под очень высокие проценты, не стоит вестись, пока действуют ограничения на валютные вклады, введенные Центробанком в 2022 году.

Литературный текст:

— Скажи-ка, дядя, ведь не даром

Москва, спаленная пожаром,

Французу отдана?

Ведь были ж схватки боевые,

Да, говорят, еще какие!

Недаром помнит вся Россия

Про день Бородина!

— Да, были люди в наше время,

Не то, что нынешнее племя:

Богатыри — не вы!

Плохая им досталась доля:

Немногие вернулись с поля…

Не будь на то господня воля,

Не отдали б Москвы!

Мы долго молча отступали,

Досадно было, боя ждали,

Ворчали старики:

«Что ж мы? на зимние квартиры?

Не смеют, что ли, командиры

Чужие изорвать мундиры

О русские штыки?»

И вот нашли большое поле:

Есть разгуляться где на воле!

Построили редут.

У наших ушки на макушке!

Чуть утро осветило пушки

И леса синие верхушки —

Французы тут как тут.

Забил заряд я в пушку туго

И думал: угощу я друга!

Постой-ка, брат мусью!

Что тут хитрить, пожалуй к бою;

Уж мы пойдем ломить стеною,

Уж постоим мы головою

За родину свою!

Два дня мы были в перестрелке.

Что толку в этакой безделке?

Мы ждали третий день.

Повсюду стали слышны речи:

«Пора добраться до картечи!»

И вот на поле грозной сечи

Ночная пала тень.

Прилег вздремнуть я у лафета,

И слышно было до рассвета,

Как ликовал француз.

Но тих был наш бивак открытый:

Кто кивер чистил весь избитый,

Кто штык точил, ворча сердито,

Кусая длинный ус.

И только небо засветилось,

Все шумно вдруг зашевелилось,

Сверкнул за строем строй.

Полковник наш рожден был хватом:

Слуга царю, отец солдатам…

Да, жаль его: сражен булатом,

Он спит в земле сырой.

И молвил он, сверкнув очами:

«Ребята! не Москва ль за нами?

Умремте ж под Москвой,

Как наши братья умирали!»

И умереть мы обещали,

И клятву верности сдержали

Мы в Бородинский бой.

Ну ж был денек! Сквозь дым летучий

Французы двинулись, как тучи,

И всё на наш редут.

Уланы с пестрыми значками,

Драгуны с конскими хвостами,

Все промелькнули перед нами,

Все побывали тут.

Вам не видать таких сражений!..

Носились знамена, как тени,

В дыму огонь блестел,

Звучал булат, картечь визжала,

Рука бойцов колоть устала,

И ядрам пролетать мешала

Гора кровавых тел.

Изведал враг в тот день немало,

Что значит русский бой удалый,

Наш рукопашный бой!..

Земля тряслась — как наши груди,

Смешались в кучу кони, люди,

И залпы тысячи орудий

Слились в протяжный вой…

Вот смерклось. Были все готовы

Заутра бой затеять новый

И до конца стоять…

Вот затрещали барабаны —

И отступили бусурманы.

Тогда считать мы стали раны,

Товарищей считать.

Да, были люди в наше время,

Могучее, лихое племя:

Богатыри — не вы.

Плохая им досталась доля:

Немногие вернулись с поля.

Когда б на то не божья воля,

Не отдали б Москвы!

Текст переписки в соцсетях:

Анна: Привет, Серёга! Как дела? Что планируешь на выходные? ??

Сергей: Привет, Анна! Всё хорошо, спасибо! В эти выходные собираюсь на концерт в парке. А ты что задумала?

Анна: О, классно! Какой концерт? Я тоже хотела бы сходить, если получится.

Сергей: Будет группа "Звёздный дождь". Они играют такую классную музыку! Начало в 18:00. Ты бы могла взять с собой кого-то?

Анна: Звучит здорово! Я могу позвать Катю, она тоже любит их. А ты с кем пойдёшь?

Сергей: Я пойду с Димой. Мы уже купили билеты. Если ты решишься, можем встретиться перед концертом.

Анна: Отлично! Давай встретимся в 17:00 у входа в парк. Как доберёшься?

Сергей: Я на автобусе, а ты?

Анна: Я на метро, так что доберусь быстро. Кстати, ты слышал, что в воскресенье будет ярмарка на площади?

Сергей: Да, слышал! Там будут разные вкусности и мастер-классы. Думаю, стоит заглянуть.

Анна: Согласна! Может, после концерта сходим на ярмарку?

Сергей: Отличная идея! После концерта как раз будет время.

Анна: Супер! Я возьму с собой немного денег, чтобы попробовать что-то вкусное.

Сергей: Я тоже! Надо будет сделать много фото, чтобы запомнить этот день.

Анна: Да, обязательно! Я так давно не была на концерте, это будет круто!

Сергей: Согласен! Надеюсь, погода будет хорошей.

Анна: Да, по прогнозу обещают солнце. Будем надеяться на лучшее!

Сергей: Если что, можем взять с собой зонт на всякий случай.

Анна: Хорошая идея! Лучше перестраховаться.

Сергей: А ты слышала, что в этом году на ярмарке будут новые стенды с уличной едой?

Анна: Нет, не слышала! Это здорово! Я обожаю уличную еду.

Сергей: Я тоже! Надо будет попробовать что-то новенькое.

Анна: Может, возьмём что-то на двоих и поделимся?

Сергей: Отличный план! Так мы сможем попробовать больше разных блюд.

Анна: Да, и не так много съедим.

Сергей: А ты уже придумала, что наденешь на концерт?

Анна: Думаю, надену свои новые джинсы и яркую футболку. А ты?

Сергей: Я, скорее всего, в спортивной куртке и кроссовках. Удобно будет танцевать!

Анна: Верно! Главное — комфорт.

Сергей: Да, и чтобы не замёрзнуть.

Анна: Точно! Если будет холодно, я возьму с собой шарф.

Сергей: Умно! Я тоже подумаю о тёплой одежде.

Анна: В общем, у нас уже есть план на выходные!

Сергей: Да, жду не дождусь!

Анна: И я! Увидимся в субботу! ??

Сергей: Увидимся! Не забудь позвонить, если что-то изменится.

Анна: Обязательно! Пока!

Сергей: Пока!