|  |
| --- |
| https://lh6.googleusercontent.com/QcftzNtI05T0Y6fjdSh1Rr2rt8oqZ1IvnLvbn1jLJ7CCyteVir3k-xBLv4SL1wAgWJsRhmmJSR0UW-RP63_GQenE4vVWv05BRoZTsmIcBccVTnfxwmsnNMvjg599x9SqZd8E3dkd |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт перспективных технологий и индустриального программирования (ИПТИП)

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКИМ РАБОТАМ** | |
| **по дисциплине** | |
| «Технологии создания программного обеспечения» | |
| **Практическая работа №4** | |
| Выполнил студент группы ЭФМО-02-23 | Мурадов Н.Н. |

Москва 2023

**СОДЕРЖАНИЕ**

[Задача 3](#_Toc154315726)

[Решение 5](#_Toc154315727)

[Результаты 9](#_Toc154315728)

Задача

Разработать приложение, которое позволит извлекать данные из json-файла и:

1. Формировать список профессиональных стандартов и

визуализировать их в виде таблицы:

|  |  |
| --- | --- |
| **Код** | **Название** |
| 06.004 | Специалист по тестированию в области информационных технологий |

2. Для указанной универсальной или общепрофессиональной

компетенции формировать список индикаторов достижения:

|  |  |
| --- | --- |
| УК-1. | Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач |
| Знать | методики поиска, сбора и обработки информации; актуальные российские и зарубежные источники информации в сфере профессиональной деятельности; метод системного анализа |
| Уметь | применять методики поиска, сбора и обработки информации; осуществлять критический анализ и  синтез информации, полученной из разных источников; применять системный подход для решения  поставленных задач |
| Владеть | методами поиска, сбора и обработки, критического анализа и синтеза информации; методикой системного подхода для решения поставленных задач |

3. Для выбранной из списка дисциплины выводить полную

информацию о ней:

|  |  |
| --- | --- |
| Б1.Б.Д1 | Философия |
| Цель | Цель изучения дисциплины «Философия» - знание и использование основных законов развития природы, общества, мышления и человека. |
| Компетенции | УК-1, УК-5, УК-6 |
| З.Е. | 4 |
| Семестры (terms) | ☐ ☑ ☐ ☐ ☐ ☐ ☐ ☐ |

4. Формировать список дисциплин, которые ведутся в выбранном из

списка семестре

|  |  |
| --- | --- |
| **Шифр** | **Название дисциплины** |
| Б1.Б.Д2 | История |

5. Формировать график учебного процесса для указанного курса. Неделя длится с понедельника по субботу. Считать первой неделю, на которую приходится 1 сентября.

|  |  |  |
| --- | --- | --- |
| **Вид обучения** | **Продолжительность** | **Количество недель** |
| Теоретическое обучение | 01.09.2020-02.01.2021 | 18 |
| Практика |  |  |
| промежуточная аттестация |  |  |
| Каникулы |  |  |
| Учебная практика |  |  |
| Производственная практика |  |  |
| Научно-исследовательская работа |  |  |
| Государственная итоговая  аттестация |  |  |

Решение

*Листинг 1-5 задачи.*

using System;

using System.Linq;

using System.Collections.Generic;

using System.Runtime.Versioning;

using Bogus;

using System.Xml.Linq;

using System.Text;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using System.Drawing;

using Newtonsoft.Json;

using Newtonsoft.Json.Linq;

// using ConsoleTables;

using MarkdownLog;

using System.Data;

public class Pract4 {

private Faker fakerRu = new("ru");

private Random rnd = new();

public void run() {

Console.WriteLine("Hello World Pract4");

Console.OutputEncoding = System.Text.Encoding.UTF8;

Console.InputEncoding = System.Text.Encoding.UTF8;

using (var sr = new StreamReader("./ОПОП.json"))

{

var obj =JObject.Load(new JsonTextReader(sr));

num1(obj);

num2(obj);

num3(obj);

num4(obj);

num5(obj);

}

Console.WriteLine("Конец Hello World Pract4");

}

private void num1(JObject json) {

Console.WriteLine("num1:");

Console.WriteLine(json["content"]["section4"]["professionalStandards"]

.Where(p => p["content"].Value<string>().IndexOf("06.0") > -1)

.Select(p=>p["content"].Value<string>().Split(" "))

.Select(p=>new {Код = p[0], Название = string.Join(" ", p[1..])})

.ToMarkdownTable());

Console.WriteLine("Конец num1\n");

}

private dynamic dopNum2(JToken jT){

if(jT["title"] == null) {

var sp = jT["content"].Value<string>().Split(": ");

return new {t1 = sp[0], t2 = sp[1]};

} else {

return new {t1 = jT["code"], t2 = jT["title"]};

}

}

private string defConsole(string con, string def) {

return con == String.Empty ? def : con;

}

private int defConsole(string con, int def) {

int x = def;

int.TryParse(con, out x);

return con == String.Empty ? def : x;

}

private void num2(JObject json) {

Console.WriteLine("num2:");

Console.WriteLine("Список компетенций:");

JArray rez = (JArray)json["content"]["section4"]["universalCompetencyRows"];

rez.Merge(json["content"]["section4"]["commonCompetencyRows"]);

Console.WriteLine(rez.Select(e => new { Код = e["competence"]["code"]})

.ToMarkdownTable());

Console.WriteLine("Выберите компетенцию(Пример | по умолчанию, 'УК-1.'): ");

string comp = defConsole(Console.ReadLine(), "УК-1.");

var competency = rez.Where(e => e["competence"]["code"].Value<string>() == comp).First();

var title = competency["competence"];

JArray indicators = (JArray)competency["indicators"];

indicators.First().AddBeforeSelf(title);

Console.WriteLine(indicators.Select(p => dopNum2(p))

.Select(p => new {t1 = p.t1, t2 = p.t2})

.ToMarkdownTable());

Console.WriteLine("Конец num2\n");

}

private void num3(JObject json) {

Console.WriteLine("num3:");

Console.WriteLine("Список дисциплин:");

var disciplines = json["content"]["section5"]["eduPlan"]["block1"]["subrows"];

Console.WriteLine(disciplines.Select(e => new { Дисциплина = e["title"]})

.ToMarkdownTable());

Console.WriteLine("Выберите дисциплину(Пример | по умолчанию, 'Философия'): ");

string nameDis = defConsole(Console.ReadLine(), "Философия");

var dis = disciplines.Where(e => e["title"].Value<string>() == nameDis).First();

var rez = new Dictionary<string, string>

{

{ dis["index"].Value<string>(), dis["title"].Value<string>() },

{ "Цель", dis["description"].Value<string>().Split(".")[0].Replace("<p>", "") },

{ "Компетенции", string.Join(", ", dis["competences"].Select(p => p["code"].Value<string>().Replace(".", "")))},

{ "З.Е.", dis["unitsCost"].Value<string>()},

{ "Семестры (terms)", string.Join(" ", dis["terms"].Select(p => p.Value<bool>() ? "☑" : "☐"))}

};

Console.Write(rez.ToMarkdownTable());

Console.WriteLine("Конец num3\n");

}

private void num4(JObject json) {

Console.WriteLine("num4:");

Console.WriteLine("Всего 8 семестров от 1 до 8 включая");

var disciplines = json["content"]["section5"]["eduPlan"]["block1"]["subrows"];

Console.WriteLine("Выберите семестр(Пример | по умолчанию, 2): ");

int numSem = defConsole(Console.ReadLine(), 2);

var dis = disciplines.Where(e => e["terms"].ElementAt(numSem-1).Value<bool>());

var rez = dis.Select(e=>new{Шифр = e["index"].Value<string>(), Название\_дисциплины = e["title"].Value<string>()});

Console.Write(rez.ToMarkdownTable());

Console.WriteLine("Конец num4\n");

}

private DateTime YearWeekDayToDateTime(int year, DayOfWeek day, int week)

{

DateTime startOfYear = new(year, 1, 1);

int daysToFirstCorrectDay = (((int)day - (int)startOfYear.DayOfWeek) + 7) % 7;

return startOfYear.AddDays(7 \* (week-1) + daysToFirstCorrectDay);

}

class Grafik {

public string t1, t2, t3;

public Grafik(string t1) {

this.t1 = t1;

}

}

private dynamic dopNum5(List<JToken> lJT, Grafik jT, int i, int sem, int year) {

string[] ids = {"Б1", "Б2", "Э", "К", "У", "П", "НИР", "Д"};

int count = lJT.Count(p => p.Value<string>() == ids[i]),

offset = sem != 0 ? 35 : 6,

begI = lJT.FindIndex(e => e.Value<string>() == ids[i]),

endI = lJT.FindLastIndex(e => e.Value<string>() == ids[i]);

if(begI > -1) begI += offset;

if(endI > -1) endI += offset;

DateTime beg = sem != 0 && i == 0 ? new(year, 9, 1) : YearWeekDayToDateTime(year, DayOfWeek.Monday, begI);

DateTime end = sem == 0 && i == 3 ? new(year, 8, 31) : YearWeekDayToDateTime(year, DayOfWeek.Saturday, endI);

return new{t2 = $"{(begI == -1 ? "" : beg.ToString("dd.MM.yyyy"))}-{(endI == -1 ? "" : end.ToString("dd.MM.yyyy"))}", t3 = count};

}

private void num5(JObject json) {

Console.WriteLine("num5:");

Console.WriteLine("Всего 8 семестров от 1 до 8 включая");

Console.WriteLine("Выберите семестр(Пример | по умолчанию, 2): ");

int numSem = defConsole(Console.ReadLine(), 2),

kurs = (int)Math.Ceiling(numSem/2f) - 1,

sem = numSem % 2;

Console.WriteLine("Выберите год(Пример | по умолчанию, 2023): ");

int year = defConsole(Console.ReadLine(), 2023);

var course = json["content"]["section5"]["calendarPlanTable"]["courses"][kurs]["weekActivityIds"];

var courseRez = (sem == 0 ? course.Skip(23) : course.Take(23)).ToList();

Grafik[] t = {

new ("Теоретическое обучение"),

new ("Практика"),

new ("промежуточная аттестация"),

new ("Каникулы"),

new ("Учебная практика"),

new ("Производственная практика"),

new ("Научно-исследовательская работа"),

new ("Государственная итоговая аттестация")

};

Console.WriteLine(t.Select((e, i) => new{t1 = e.t1, t2 = e.t2, t3 = e.t3, dop = dopNum5(courseRez, e, i, sem, year)})

.Select(e => new{Вид\_обучения = e.t1, Продолжительность = e.dop.t2, Количество\_недель = e.dop.t3})

.ToMarkdownTable());

Console.WriteLine("Конец num5\n");

}

}

Результаты

Результаты выполнения кода 1-5 задачи представлены на Рисунке 1.

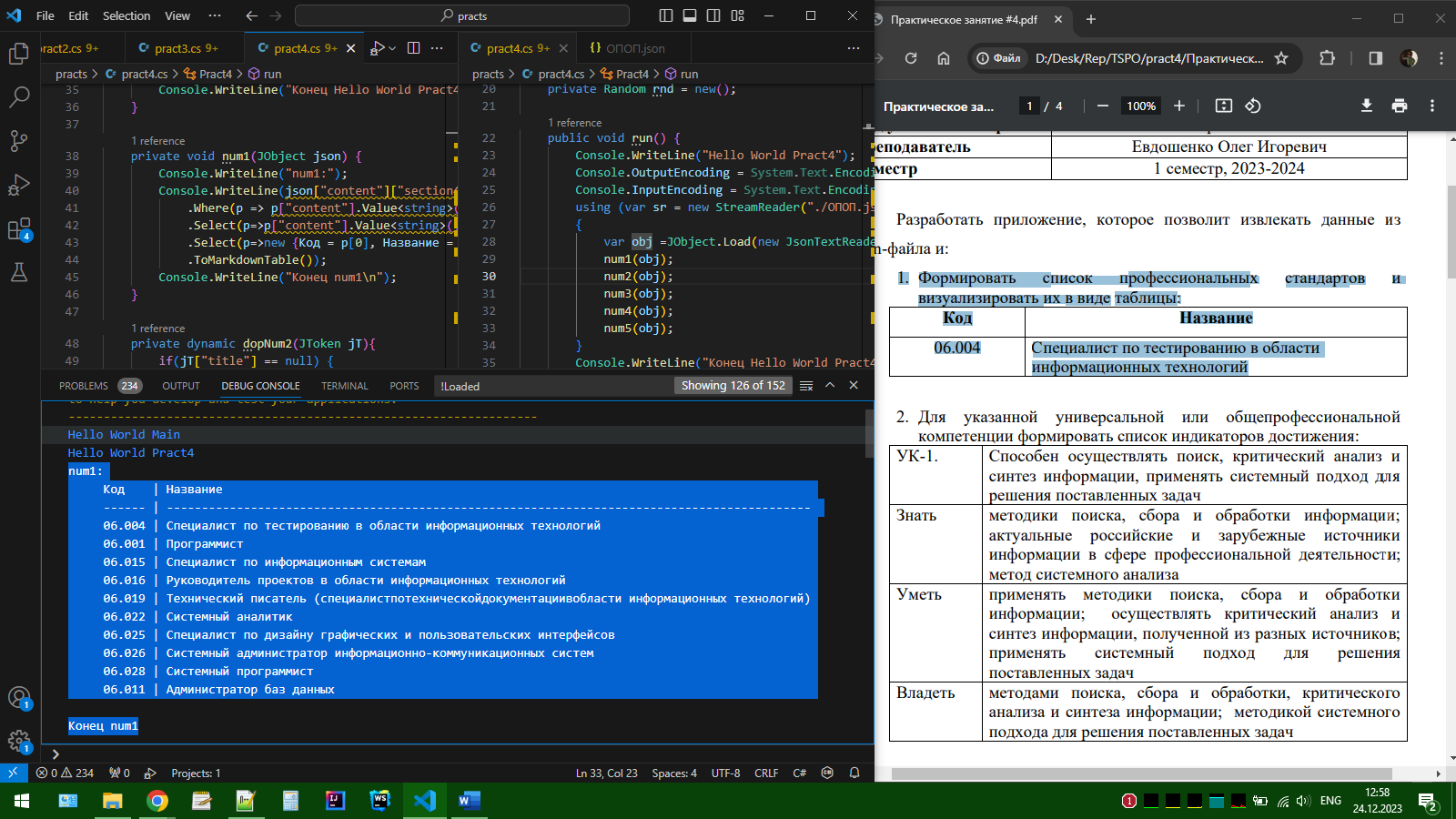


Рисунок 1 – Результаты компиляции кода 1 задачи

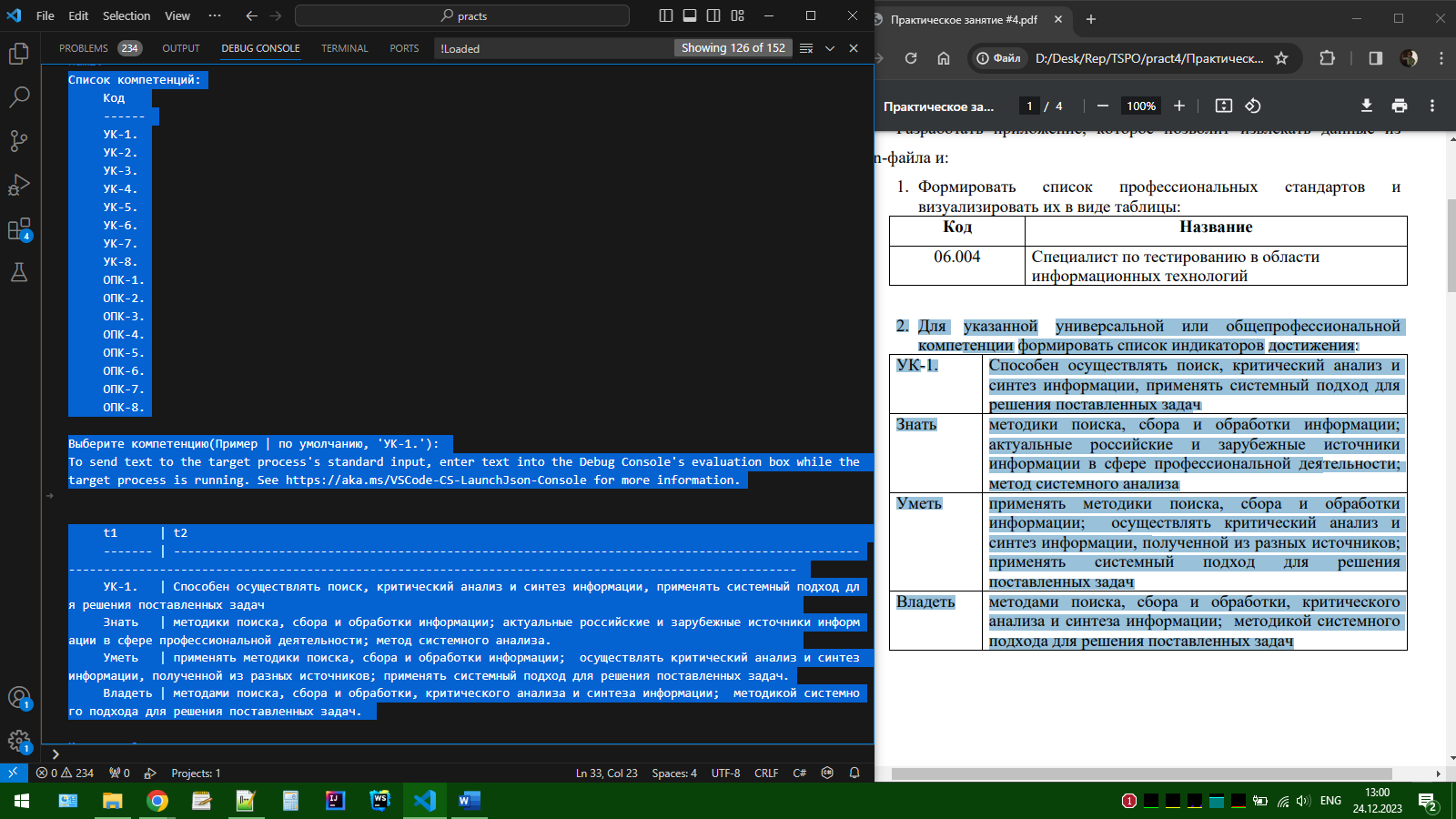


Рисунок 2 – Результаты компиляции кода 2 задачи

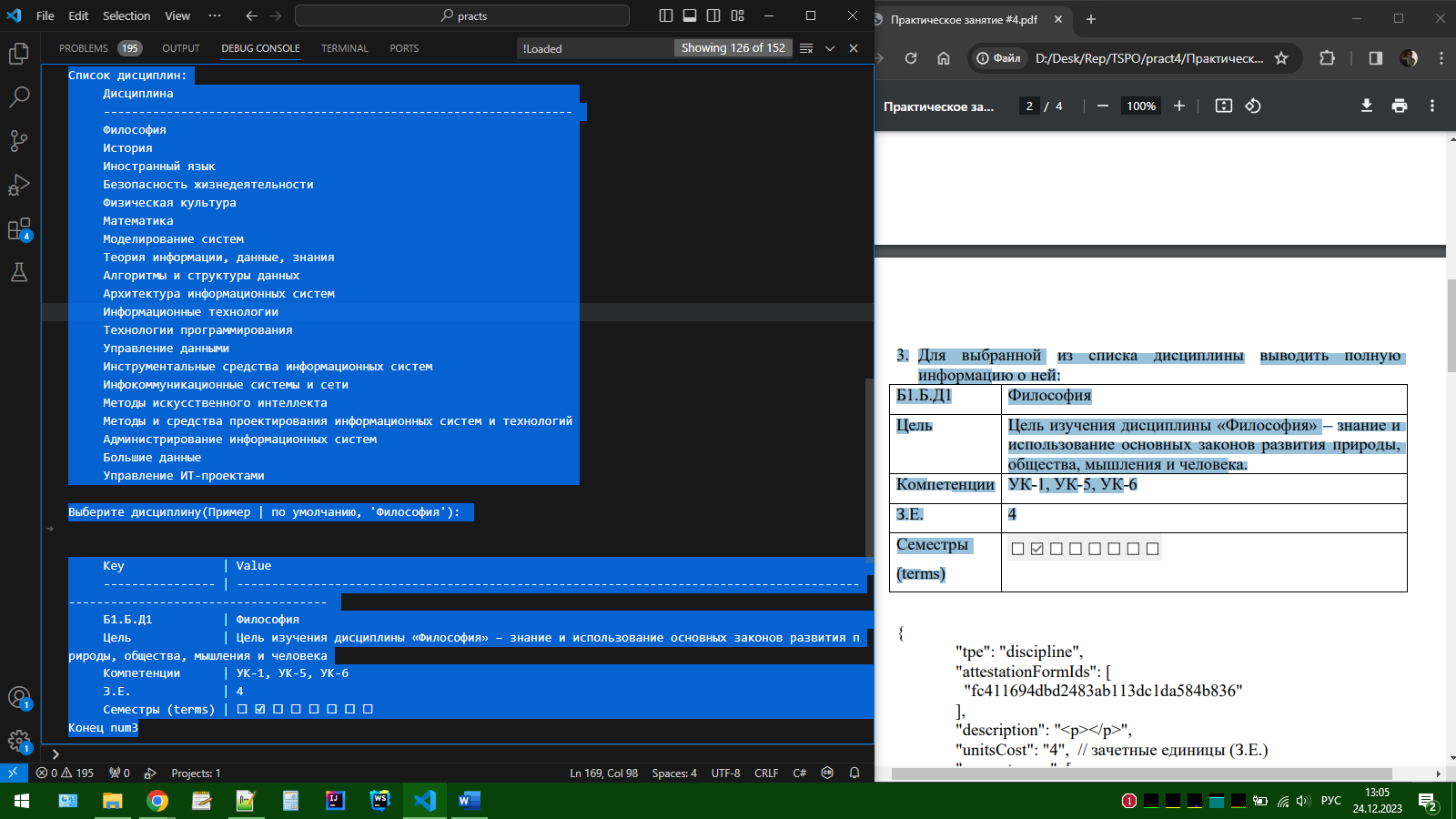


Рисунок 3 – Результаты компиляции кода 3 задачи

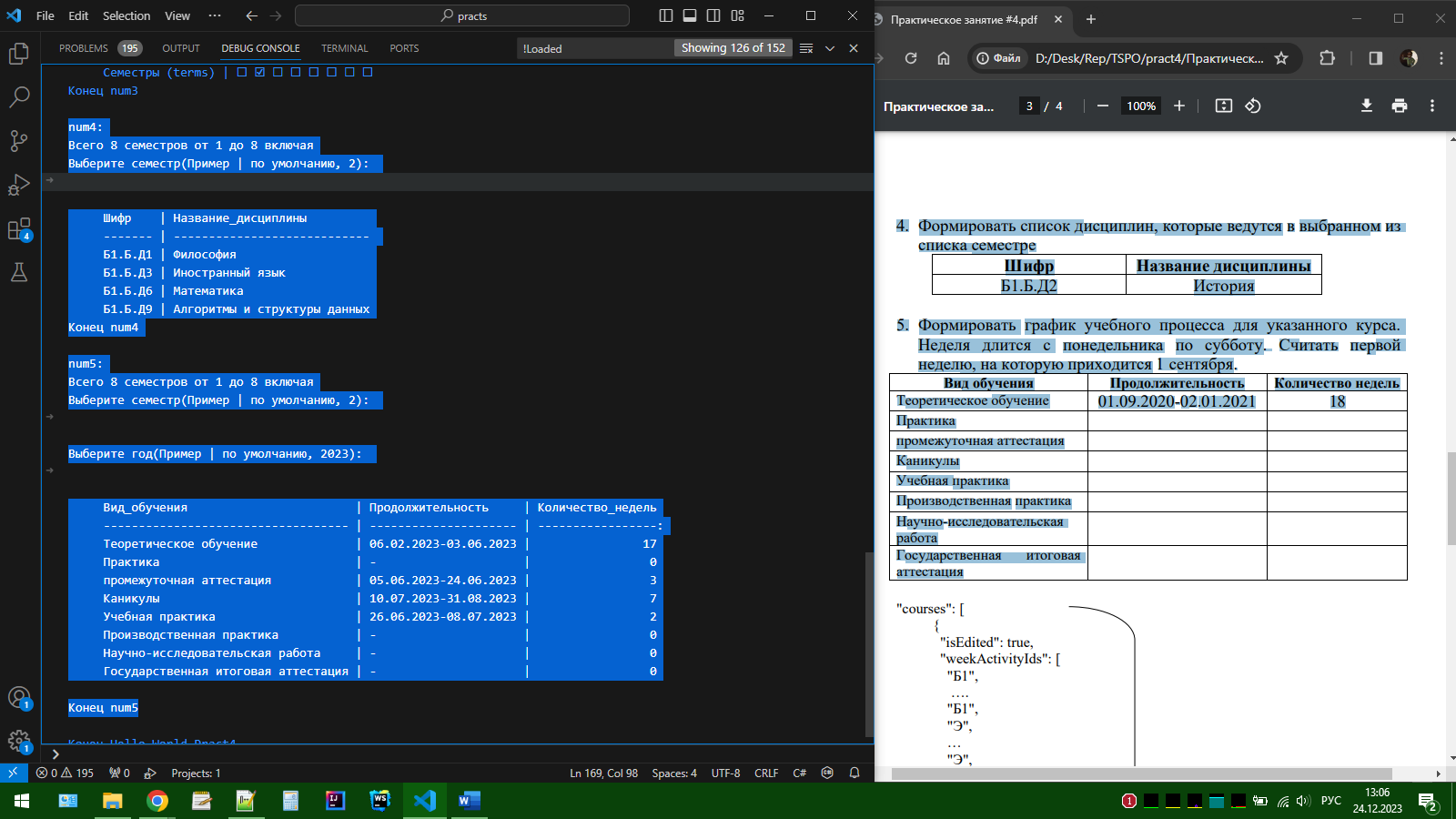


Рисунок 4 – Результаты компиляции кодов 4 и 5 задач