МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

ОТЧЕТ

по учебной вычислительной (ознакомительной) практике

Минковского Виталия Викторовича

студента 1 курса, 2 группы

специальность «Информатика»

Руководитель практики:

старший преподаватель

В. П. Дубков

Минск, 2024

# ОГЛАВЛЕНИЕ

[ОГЛАВЛЕНИЕ 2](#_Toc171102258)

[Введение 3](#_Toc171102259)

[Глава 1 Теоретические основы 4](#_Toc171102260)

[1.1 Общая характеристика курса «Контроль версий с помощью Git» 4](#_Toc171102261)

[1.1.1 Обзор курса 4](#_Toc171102262)

[1.1.2 Модули курса 4](#_Toc171102263)

[1.2 Общая характеристика курса «Python» 5](#_Toc171102264)

[1.2.1 Обзор курса 5](#_Toc171102265)

[1.2.2 Модули курса 5](#_Toc171102266)

[1.3 Общая характеристика курса «Знакомство с форматом JSON и возможностью работы из приложения с файловой системой». 11](#_Toc171102267)

[1.3.1 Обзор курса 11](#_Toc171102268)

[1.3.2 Модули курса 11](#_Toc171102269)

[1.4 Выводы к главе 1 12](#_Toc171102270)

[Глава 2 Практическая часть курса 13](#_Toc171102271)

[2.1 Решение задач из курса 13](#_Toc171102272)

[2.2 Дополнительный материал 13](#_Toc171102273)

[2.3 Выводы к главе 2 13](#_Toc171102274)

[Глава 3 Индивидуальный проект 14](#_Toc171102275)

[3.1 Функциональное назначение 14](#_Toc171102276)

[3.2 Структура проекта 14](#_Toc171102277)

[3.3 Интерфейс приложения 15](#_Toc171102278)

[3.4 Описание проекта 15](#_Toc171102279)

[3.5 Выводы к главе 3 15](#_Toc171102280)

[Заключение 16](#_Toc171102281)

[Список использованных источников 17](#_Toc171102282)

# Введение

Введение обычно имеет объем 1-2 страницы. В данном отчете введение должно включать:

* цель практики, задачи практики;
* выбранные направления работы, их актуальность;
* краткое представление (наличие) Индивидуального проекта.

Целью учебной вычислительной (ознакомительной) практики является закрепление полученных знаний за первый год обучения посредством выполнения специальных учебных заданий, участия в работе над командным проектом.

Задачи практики:

* Знакомство с IT компанией, возможностями обучения и повышения квалификации, знакомство с разными направлениями профессиональной подготовки IT специалистов.
* Изучение инструментов для совместной разработки проектов и контроля версий.
* Подготовка и настройка программных сред и средств тестирования для прохождения тренинга по выбранному направлению.
* Выполнение стандартных заданий.
* Выполнение заданий повышенного уровня.
* Разработка приложения «Аудио плеер»
* Оформление отчета.

При прохождении практики были выбраны такие направления работы, как, JSON, файловая система, Python. Первые два не теряют своей атуальности, так как являются фундаментальными знаниями. Python – перспективный язык программирования, на котором разрабатывется большое число программного обеспечения.

В совместном проекте «Аудио плеер» продуманы стандартные функции аудиопроигрывателя, возможность работы с плейлистами, ведение статистики. Более подробно приложение будет рассмотрено ниже.

# Глава 1 Теоретические основы

## 1.1 Общая характеристика курса «Контроль версий с помощью Git»

Указать, какие курсы изучены в рамках практики. Их описание, предназначение.

Замечания по оформлению: каждый структурный элемент (введение, заключение, главы, список использованных источников, приложения) начинать с новой страницы!

Недопустимо наличие в главе только одного раздела, а в разделе – только одного подраздела.

## 1.1.1 Обзор курса

В данном курсе были изучены концепции работы с Git, различные инструменты контроля версий. В ходе обучения были предложены наборы видео, в которых специалист подробно рассказывал про работу каждого модуля. В конце каждой подтемы были тесты для закрепления изученного материала. После всего теоретического блока были задания, которые я успешно выполнил. По завершении курса я закрепил имеющиеся знания и приобрёл новые.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

## 1.1.2 Модули курса

В ходе прохождения курса были изучены следующие модули:

* VCS concepts - изучены концепции системы контроля версий
* Version control types - различные типы систем контроля версий
* Why Git – преимущества использования Git
* Downoload, install, configure – руководство по установке Git.
* Create a github repo and clone it – создание удалённого репозитория и клонирования на свою машину.
* Pull from remote – получение изменений с удалённого репозиторий на локальный.
* Git Gui & gitk – возможность использования графического пользовательского интерфейса для работы с Git.
* Inside .git folder – компоненты папки .git folder, их предназначение.
* Undoing changes – отмена действий, внесённых в файловую систему.
* git reset – разбор команды.
* git revert – разбор команды.
* .gitignore – знакомство с этим файлом и его предназначением
* Branching and merge – концепты «веток» и их слияние.
* Conflict solving – решение конфликтов, вызванных слиянием веток.
* Rebase – разбор команды rebase.
* Cherry-pick – разбор команды cherry-pick.
* Tags – тэги и работа с ними.
* Stashing – концепт «стэшинга»
* Remotes – концепт «ремоутов»
* Branching strategies – стратегия использования «веток»
* Extras – советы авторы по дальнейшему образованию.

## 1.2 Общая характеристика курса «Python»

## 1.2.1 Обзор курса

В данном курсе я проходил hanbook от Яндекса по Python, в котором была теоритеческая база с задачами по каждой темы. Этот курс оказался весьма удобный, так как правильность решения заданий проверяется автоматически на сайте. И самих задач было достаточно для хорошего закрепления пройденного материала. В ходе прохождения handbook я закрепил имеющиеся знания по Python и получил новые, прорешал 104 задания. В дальнейшем я продолжу прохождение handbook, так как за время практики не успел завершить его.

## 1.2.2 Модули курса

1. Введение

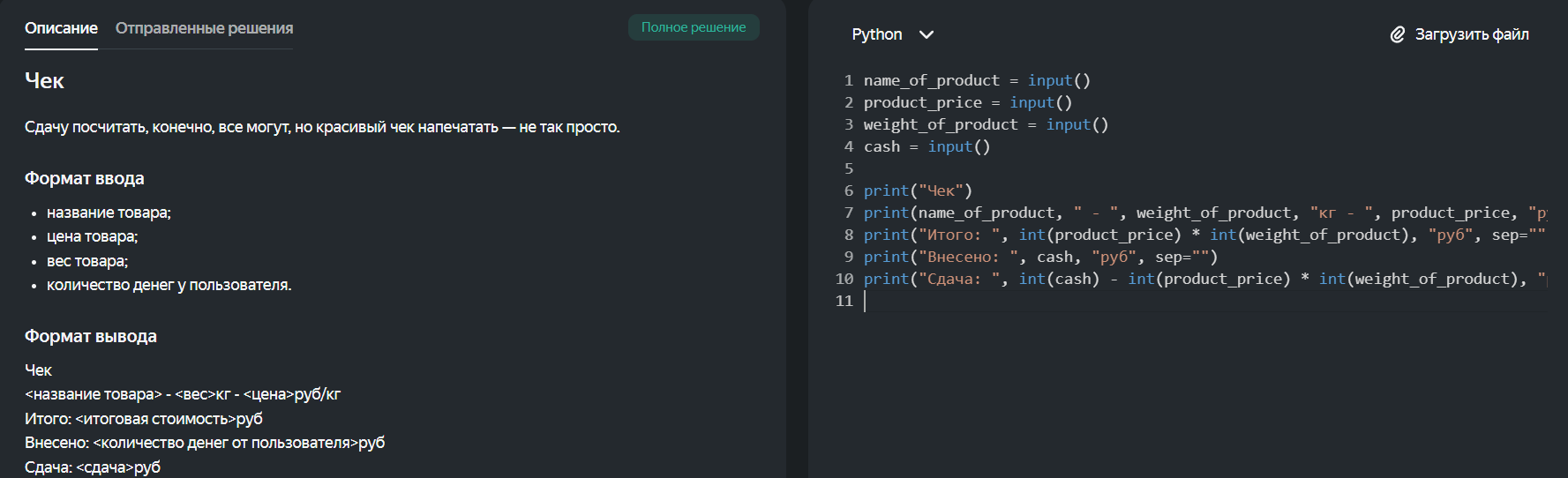
1.1 Как устроен хендбук. Рассказано, как работать с хендбуком.

1.2 Введение. Рассказано про курс, Python, среды разработки на данном языке программирования.

2. Базовые конструкции языка

2.1 Ввод и вывод данных. Операции с числами, строками. Форматирование.

Работу данного модуля можно продемонстрировать на примере следующей задачи.



2.2 Условный оператор.

Работу условных операторов можно продемонстрировать на примере следующей задачи.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

2.3 Циклы

Работу цикла while можно продемонстрировать на примере слдедующей задачи.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

А работу цикла for можно продемонстрировать на следующей.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

2.4 Вложенные циклы

Пример исользования данного инструмента:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

3. Коллекции и работа с памятью

3.1 Строки, кортежи, списки.

Пример работы с некоторыми из этих коллекций:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

3.2 Множества, словари.

Пример использования словаря:

Изображение выглядит как снимок экрана, текст, программное обеспечение

Автоматически созданное описание

Пример использования множества:

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

3.3 Списочные выражения. Модель памяти для типов языка Python

Как пример, можно привести следующую задачу:

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

3.4 Встроенные возможности по работе с коллекциями.

Продемунстрируем одну из возможностей.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

3.5 Потоковый ввод/вывод. Работа с текстовыми файлами. JSON.

Продемонстрируем работу с потоковым вводом.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

4. Функции и их особенности в Python.

4.1 Функции. Области видимости. Передача параметров в функции.

Напишем функцию для демонстрации её особенностей в Python.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

4.2 Позиционные и именованные аргументы. Функции высших порядков. Лямбда-функции.

Покажем одну из интересных возможностей передачи аргументов в функцию.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

4.3 Рекурсия. Декораторы. Генераторы.

Суть рекурсии хорошо отражается в следующей задаче.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

5. Объектно-ориентированное программирование.

5.1 Объектная модель Python. Классы, поля и методы.

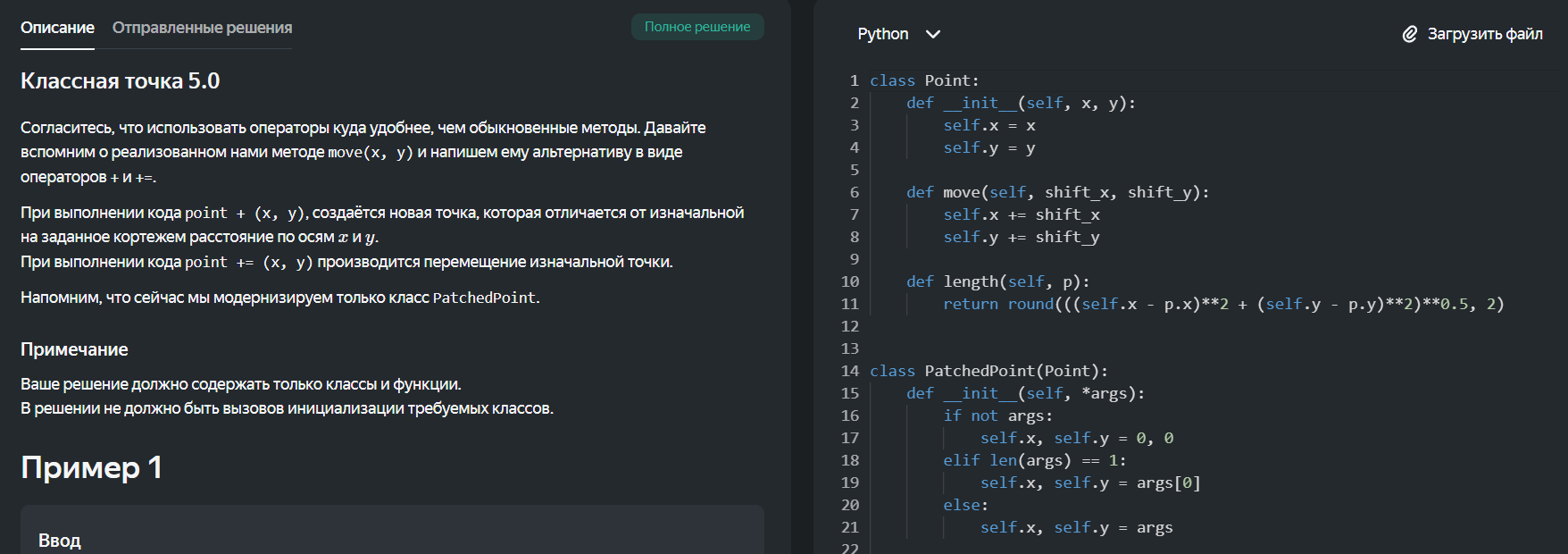
Следующий пример демонстрирует особенность классов в Python.

Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание

5.2 Волшебные методы. Переопределение методов. Наследование.

Покажем пример наследования.



5.3 Модель исключений Python. Try, except, else, finally. Модули.

Следующий пример показывает принцип обработкт исключений.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

5. Билиотеки для получения и обработки данных.

5.1 Модули math и numpy

Продемонстрируем работу некоторых функций из библиотеки math.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

5.2 Модуль Pandas

5.3 Модуль Requests

Последние два модуля я не успел изучить, но сделаю это летом.

## 1.3 Общая характеристика курса «Знакомство с форматом JSON и возможностью работы из приложения с файловой системой».

## 1.3.1 Обзор курса

В данном курсе представлен теоретический материал по формату JSON и работе с файловой системе из приложения. В конце предложены 10 лабораторных заданий для закрепления изученного на практике. В ходе этого курса я освоил инструменты работы с JSON и файловой системы приложения.

## 1.3.2 Модули курса

1. Работа с файлами формата JSON

1.1 Общие сведения о формате JSON

1.2 Структура JSON файла

1.3 JSON и C++

1.4 Класс nlohmann::json

1.5 Класс nlohmann::json и STL

1.6 Методы get, get\_ref и get\_ptr

1.7 Метод dump

1.8 Обработка исключений

2. Работа с файловой системой

2.1 Общие сведения

2.2 Класс std::filesystem::path

2.3 Декомпозиция пути

2.4 Изменение пути

2.5 Обработка ошибок

2.6 Параметры файлов и каталогов

2.7 Управление файлами и каталогами

2.8 Итераторы каталогов

2.9 Содержимое каталогов

Названия модулей говорят сами за себя, о чём там написано.

## 1.4 Выводы к главе 1

При прохождении теоретических основ курсов я изучил:

* концепции и инструменты работы системы контроля версий Git
* python
* формат JSON и работа с файловой системой из приложения

# Глава 2 Практическая часть курса

## 2.1 Решение задач из курса «Знакомство с форматом JSON и возможностью работы из приложения с файловой системой»

В каждой лабораторной работе я продемонстрирую одну функцию.

Лабораторная работа №1. Проверка структуры файлов формата JSON

Задание

Разработать консольное приложение, проверяющее соответствие структуры файла формата JSON структуре, заданной в соответствующем варианте задания. Порядок следования пар ключ:значение в JSON-объектах не учитывается. Обозначения {…} и […] соответствуют JSON-объектам и JSON-массивам с произвольным содержимым. Входные данные приложения: полный путь к объекту файловой системы (файлу формата JSON).

void CheckArgumentsAmount(int arguments\_amount) {

const int required\_arguments = 1;

if (arguments\_amount != required\_arguments) {

throw std::invalid\_argument("Invalid command line arguments amount: current – " +

std::to\_string(arguments\_amount) +

", required – " + std::to\_string(required\_arguments) + "!");

}

}

Лабораторная работа №2. Изменение структуры файлов формата JSON

Задание

Разработать консольное приложение, формирующее из одного входного файла в формате JSON несколько аналогичных файлов специальным образом. Входной файл представляет собой json-объект, каждому ключу которого соответствует также json-объект. Для каждой пары ключ:json-объект необходимо создать файл с именем ключ. В данный файл необходимо записать соответствующий ключу json-объект, содержимое которого должно быть предварительно отредактировано в соответствии с вариантом задания. Входные данные приложения: полный путь к объекту файловой системы (входному файлу в формате JSON).

void ModifyJsonObject(nlohmann::json& jsonObject) {

std::vector<std::string> vect;

for (auto it = jsonObject.begin();

it != jsonObject.end(); ++it) {

if (it.value().is\_null()) {

vect.push\_back(it.key());

}

}

for (const auto& element : vect) {

jsonObject.erase(element);

}

}

Лабораторная работа №3. Сериализация даты и времени в формат JSON

Задание

Разработать консольное приложение, модифицирующее текущие значения даты и времени в соответствии с вариантом задания. Результат необходимо представить в формате JSON, вывести его на экран, а также записать в соответствующий файл. Входные данные приложения: полный путь к объекту 24 файловой системы (файлу, в котором будут записаны значения даты и времени в формате JSON).

tm GetRequiredDateTime() {

time\_t current\_time = time(nullptr);

current\_time += 365 \* 24 \* 60 \* 60;

tm new\_time;

localtime\_s(&new\_time, &current\_time);

return new\_time;

}

Лабораторная работа №4. Сериализация информации об объектах файловой системы в формат JSON

Задание

Разработать консольное приложение, выводящее на экран информацию об объекте файловой системы в формате JSON, а также записывающее эту 26 информацию в файл. Входные данные приложения: полный путь к объекту файловой системы.

nlohmann::json GetRegularFileInfo(const std::filesystem::path& path\_to\_file) {

nlohmann::json file\_info;

file\_info["type"] = "regular\_file";

file\_info["full\_name"] = path\_to\_file.filename().string();

file\_info["name\_without\_extension"] = path\_to\_file.stem().string();

file\_info["extension"] = path\_to\_file.has\_extension() ? path\_to\_file.extension().string() : nullptr;

file\_info["size"] = Size(path\_to\_file);

return file\_info;

}

Лабораторная работа №5. Извлечение информации об объектах файловой системы

Задание

Разработать консольное приложение, выводящее на экран информацию о каталоге как об объекте файловой системы (обобщенную информацию о каталоге, обобщенную информацию о каждом вложенном в каталог объекте 29 файловой системы). Входные данные приложения: полный путь к объекту файловой системы.

std::ostream& operator<<(std::ostream& os, const Info& info) {

os << std::left << std::setw(50) << info.name

<< std::setw(20) << info.type

<< std::setw(20) << info.size;

return os;

}

Лабораторная работа №6. Копирование файлов. Сравнение файлов по содержимому

Задание

Разработать консольное приложение, копирующее файлы из одного каталога в другой. Копирование файла должно происходить при условии отсутствия в каталоге назначения файла с таким же содержимым. При совпадении имени копируемого файла с именем некоторого файла в каталоге назначения последний должен быть перезаписан. Входные данные приложения: два полных пути к объектам файловой системы: − каталог, из которого будут копироваться файлы (исходный каталог); − каталог, в который будут копироваться файлы (каталог назначения).

void CheckDirectoryPath(const std::filesystem::path& path\_to\_directory) {

if (!std::filesystem::exists(path\_to\_directory)) {

throw std::runtime\_error("Filesystem object by path " + path\_to\_directory.string() + " does not exist!");

}

if (!std::filesystem::is\_directory(path\_to\_directory)) {

throw std::runtime\_error("Filesystem object by path " + path\_to\_directory.string() + " is not a directory!");

}

}

Лабораторная работа №7 Копирование файлов. Сравнение файлов по содержимому

Задание

Разработать консольное приложение, копирующее файлы из одного каталога в другой. Копирование файла должно происходить при условии отсутствия в каталоге назначения файла с таким же содержимым. При совпадении имени копируемого файла с именем некоторого файла в каталоге назначения последний должен быть перезаписан. Приложение принимает на вход два полных пути к объектам файловой системы: − каталог, из которого будут копироваться файлы (исходный каталог); − каталог, в который будут копироваться файлы (каталог назначения).

FilesStorage::FilesStorage(const std::filesystem::path& path\_to\_directory)

: path\_to\_directory\_(path\_to\_directory) {}

Лабораторная работа №8. Удаление файлов. Механизм контрольных сумм

Задание

Разработать консольное приложение, удаляющие дубликаты файлов по содержимому из каталога. Входные данные приложения: полный путь к объекту файловой системы (каталог, из которого необходимо удалить дубликаты).

std::size\_t GetFileContentHash(const std::filesystem::path& path\_to\_file) {

std::string content = ReadFileContent(path\_to\_file);

return std::hash<std::string>{}(content);

}

Лабораторная работа №9. Перемещение файлов

Задание

Разработать консольное приложение, перемещающее файлы из указанного (исходного) каталога по определенному правилу. Входные данные приложения: полный путь к исходному каталогу, содержащему файлы, которые необходимо переместить.

void Move(const std::filesystem::path& path\_to\_file) {

std::filesystem::path new\_path = GetPathToMove(path\_to\_file);

std::filesystem::create\_directories(new\_path.parent\_path());

std::filesystem::rename(path\_to\_file, new\_path);

std::cout << "File by path " << path\_to\_file.string() << " has been moved to " << new\_path.string() << std::endl;

}

Лабораторная работа №10. Перемещение файлов

Задание

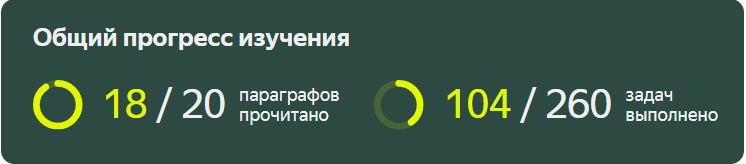
Разработать консольное приложение, перемещающее файлы из указанного (исходного) каталога по определенному правилу. Входные данные приложения: полный путь к исходному каталогу, содержащему файлы, которые необходимо переместить.

bool FileName::IsRemoveRequired() const {

return (year % 5 == 0 && day % 5 == 0);

}

## 2.2 Python



Ниже будет написано количество задач, решённых на каждую тему, и примеры.

- Базовые конструкции Python

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

- Ввод и вывод данных. Операции с числами, строками. Форматирование.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

- Условный оператор

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

- Циклы

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

- Вложенные циклы

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

- Коллекции и работа с памятью

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

- Строки, кортежи, списки

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

- Множества, словари

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

- Списочные выражения. Модель памяти для типов языка Python

Изображение выглядит как текст, снимок экрана

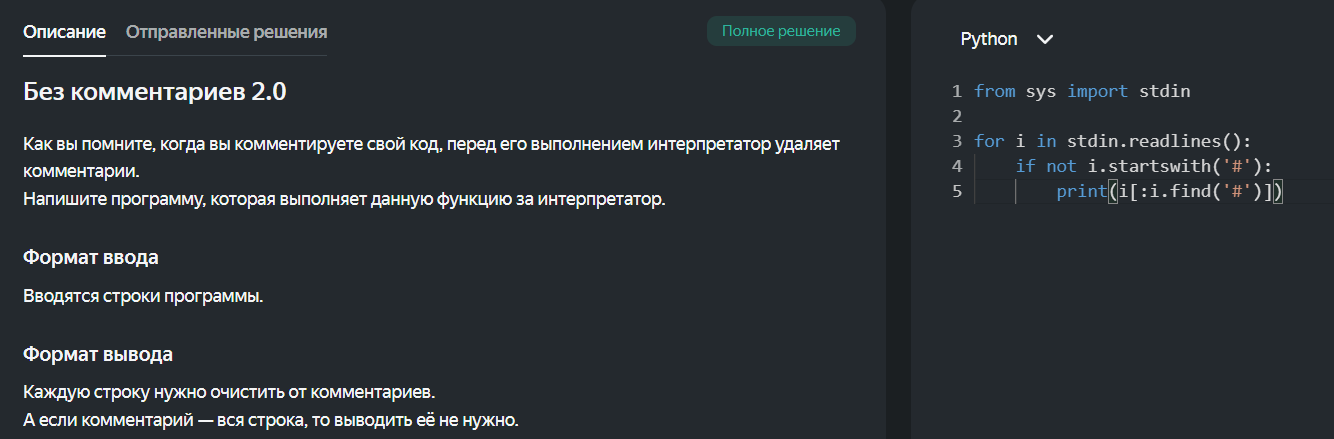
Автоматически созданное описание

- Встроенные выражения по работе с коллекциями

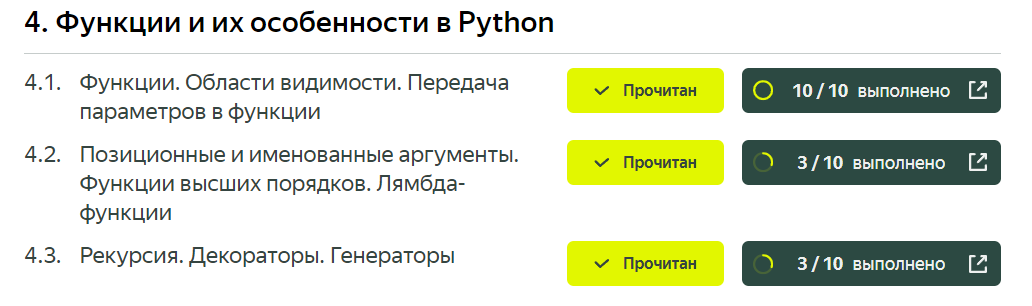
Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

- Потоковый ввод/вывод. Работа с текстовыми файлами. JSON



- Функции и их особенности в Python



- Функции. Области видимости. Передача параметров в функции.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

- Позиционные и именованные аргументы. Функции высших порядков. Лямбда-функции.

Изображение выглядит как текст, снимок экрана, Шрифт

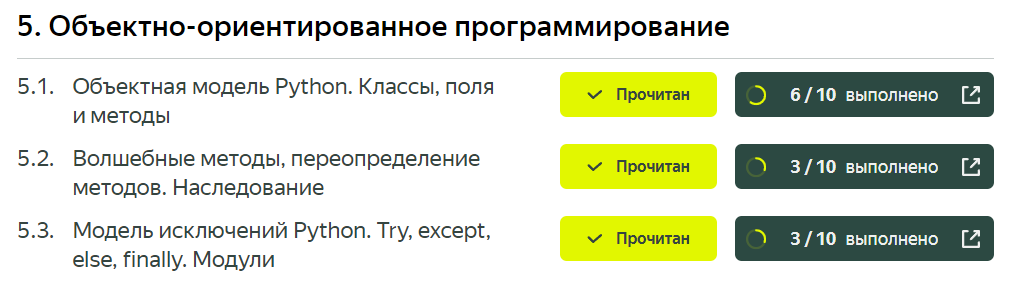
Автоматически созданное описание

- Рекурсия. Декораторы. Генераторы.

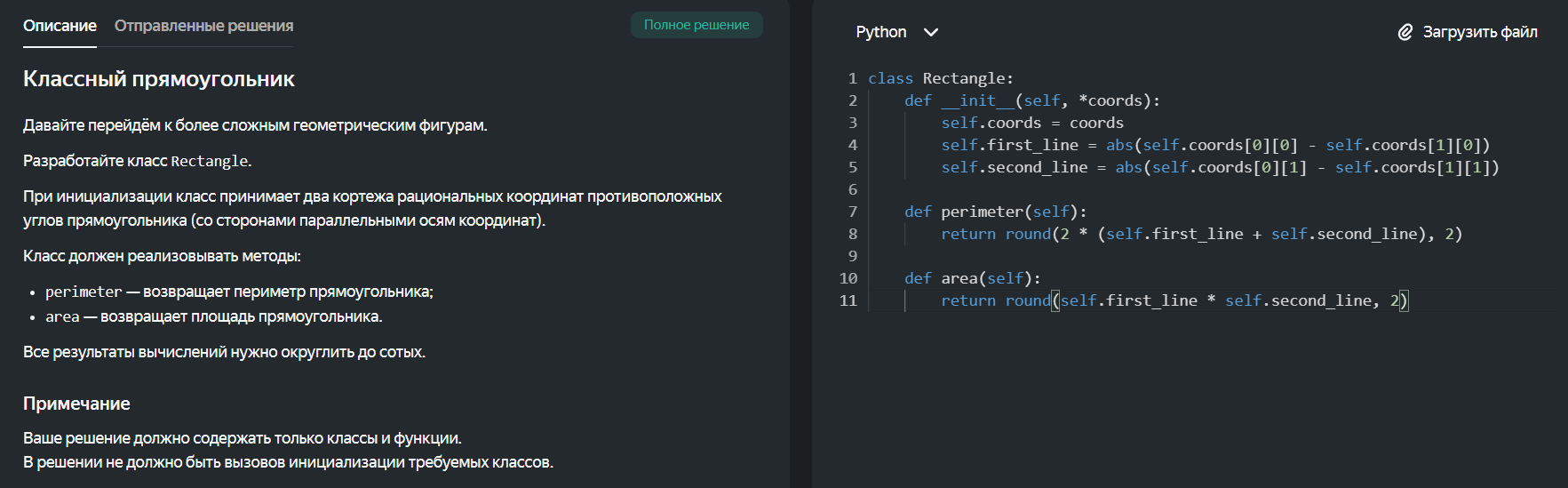
Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

- Объектно-ори­­­ентированное программирование



- Объетная модель Python. Классы, поля и методы



- Волшебные методы, переопределение методов.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

- Модель исключений Python. Try, except, else, finally. Модули

Изображение выглядит как текст, программное обеспечение, Шрифт, Мультимедийное программное обеспечение

Автоматически созданное описание

- Билиотеки для получения данных.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

- Модули math и numpy

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

## 2.3 Выводы к главе 2

Во время выполнения практической части курса я решил курс по Git, сделал все 10 лабораторных работ по JSON и файловой системе, а также 104 задания из хендбука.

# Глава 3 совместный проект «Аудио плеер»

## 3.1 Функциональное назначение

Приложение Аудио плеер – приложение для прослушивания аудио треков, с возможностью работы в разных аккаунтах с плейлистами. Также будет вестись статистика пользователей.

## 3.2 Структура проекта

class MainWindow : public QMainWindow

{

…

private slots:

void on\_buttonPlay\_clicked();

void on\_buttonWhiteHeart\_clicked();

void on\_buttonUserPhoto\_clicked();

void on\_sliderPressed();

void on\_sliderReleased();

void on\_sliderValueChanged(int value);

void on\_sliderMoved(int value);

void onbuttonForward\_10sec\_clicked();

void on\_buttonBack\_10sec\_clicked();

void on\_buttonAddSong\_clicked();

void on\_buttonVolume\_clicked();

void updateLabelNameOfTheSong();

void on\_buttonSong\_clicked(int index);

void on\_buttonMyMusic\_clicked();

void on\_buttonSets\_clicked();

void on\_buttonPlaylists\_clicked();

void on\_buttonForward\_clicked();

void on\_buttonBack\_clicked();

void onMediaStatusChanged(QMediaPlayer::MediaStatus status);

void durationChanged(qint64 duration);

void positionChanged(qint64 position);

void on\_buttonMyPlaylist\_clicked();

void addCurrentSongToFavorites();

void on\_buttonRemoveSong\_clicked();

private:

Ui::MainWindow \*ui;

QMediaPlayer \*MPlayer;

QAudioOutput \*audioOutput;

QStringList playlist;

qint64 mDuration;

QJsonArray jsonArray;

QPixmap getCircularPixmap(const QPixmap &src);

void playCurrent();

void updateDurationInfo(qint64 duration);

void updatePositionInfo(qint64 position);

void saveSong(const QString &filePath);

void loadPlaylistFromFile(const QString &filePath);

};

Выше приведён класс из заголовочника, в котором пока что находится весь функционал. В дальнейшем проект будет разбит на несколько файлов, классов, в котором добавим новые возможности. В блоке «private slots» находятся обработчики событий кнопок и скроллбара. Их имена говорят, что они делают.

В блоке «private» функция playCurrent() отвечает за запуск проигрывания аудио трека. saveSong() сохраняет песни из пелйлиста на локальную машину. loadPlaylistFromFile() загружает песни в плейлист из файла. Функции updateDurationInfo(qint64 duration), updatePositionInfo(qint64 position) отвечают за отображение текущего времени песни.

## 3.3 Интерфейс приложения

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Интерфейс приложения интуитивно понятен любому пользователю, на каждой кнопке есть картинка либо текст, поясняющие её функционал.

## 3.4 Описание проекта

В приложении «Аудио плеер» задумано следующее:

- воспроизведение, пауза

- перемотка трека с помощью кнопок и скролбара

- переключение на следующий/предыдущий трек, автоматическое переключение следующей песни по завершению текущей

- ­­­кнопка выключить/включить звук

- добавление, удаление, редактирование плейлистов

- добавление песни в понравившиеся

- система аккаунтов

- ведение статистики

## 3.5 Выводы к главе 3

Основной функционал приложения реализован, но ещё есть над чем работать.

Текст выводов.

Выводы и предложения должны быть конкретными, реальными и обоснованными, вытекать из полученных результатов. Выводы пишутся тезисно (по пунктам).

# Заключение

В ходе работы на практике были получены новые навыки и зниния. В дальнейшем я завершу курс по Python и проект, а также продолжу продвигаться в желанном направлении – машинном обучении.

# Список использованных источников

1. Усов, В. Swift. Разработка приложений под iOS на основе фреймворка UIKit // В. Усов. – Москва, 2021. – 492 с. (Книга с одним автором)
2. Дробышевский, Н. П. Ревизия и аудит : учеб.-метод. пособие / Н. П. Дробышевский. – Минск : Амалфея : Мисанта, 2013. – 415 с. (Книга с одним автором)
3. Дьяченко, Л. С. Методические рекомендации по подготовке и сдаче государственного экзамена по педагогике / Л. С. Дьяченко, Н. К. Зинькова, Р. В. Загорулько. – Витебск : Витеб. гос. ун-т, 2013. – 53 с. (Книга с тремя авторами)
4. Национальный статистический комитет Республики Беларусь [Электронный ресурс]. – Режим доступа: http://belstat.gov.by/. – Дата доступа: 24.06.2016. (Сайт целиком, т.е. url заканчивается доменом первого уровня)
5. Полоник, С. С. Теоретико-методологические основы обеспечения внешнеэкономической безопасности Республики Беларусь в условиях усиления международной конкуренции / С. С. Полоник, Э. В. Хоробрых, А. А. Литвинчук // Вестн. Полоц. гос. ун-та. Сер. D, Экон. и юрид. науки. – 2013. – № 5. – С. 7–16. (Статья из журнала)
6. Жданухин, Д. Ю. Коллекторские агентства: основные черты деятельности, история появления и зарубежный опыт [Электронный ресурс] / Д. Ю. Жданухин // Юридическая Россия : Федер. правовой портал. – Режим доступа: http://law.edu.ru/doc/document.asp?docID=1233648. – Дата доступа: 22.06.2016. (Статья с сайта с указанием автора)
7. О демографической ситуации в январе – июне 2016 года [Электронный ресурс] // Национальный статистический комитет Республики Беларусь. – Режим доступа: http://www.belstat.gov.by/o-belstate\_2/novosti-i-meropriyatiya/novosti/o\_demograficheskoy\_situatsii\_v\_yanvare\_iyune\_2016\_g/. – Дата доступа: 29.07.2016. (Статья с сайта без автора)