Министерство образования Республики Беларусь

Белорусский Национальный Технический Университет

Факультет информационных технологий и робототехники

Кафедра «Программное обеспечение информационных систем и технологий»

**Отчёт**

***по Лабораторной работе №24***

по дисциплине ***«Языки программирования»***

Исполнитель: Колосов Артём Александрович

Группа: 10701219

Преподаватель: Лапанович Ирина Олеговна

2020

Цель работы:

Изучить и закрепить навыки работы с файлами и с высокоуровневыми типами

в Python, а также научиться грамотно обрабатывать потенциальные ошибки и

исключительные ситуацию, которые могут возникнуть в процессе работы

программы.

Основное задание:

Дан текстовый файл in.txt, в который записан список студентов групп потока

первого курса с результатами зимней экзаменационной сессии. Необходимо

написать приложение, которое подсчитывала средний балл для каждого из

студентов заданной группы и средний балл всей группы, а также находила в

соответствующей группе студентов с максимальным (минимальным) средним

баллом. Дополнительно нужно выявить группу с максимальным

(минимальным) средним баллом и успевающих (неуспевающих) студентов на

потоке. Все результирующие данные необходимо записать в файл out.txt.

Исходный файл можно создать вручную или воспользоваться тем, что

прилагается к лабораторной работы для тестирования конечного результата.

Информация в результирующем файле должна быть представлена в

читабельном (человеческом) виде.

**Индивидуальное задание:**

Задание 1.

В тексте, вводимом с клавииатуры, подсчитать количество символов, лежащих в диапазоне от 'g' до 'o'.

Задание 2.

Дана строка текста, в которой слова разделены пробелами и запятыми. Необходимо:

       а) определить количество слов, ограниченных с двух сторон пробелами;

       б) определить, сколько раз повторяется каждая буква (без учета регистра);

       в) вывести в алфавитном порядке все словосочетания, отделенные запятыми.

Строку инициализировать в коде программы:

So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and stupid, whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

Задание 3.

Написать программу, которая считывает русский текст из  файла и записывает в новый файл тот же текст, заменив все буквы на прописные.

Текстовый файл создать в блокноте.

Необходимо выполнить задания из раздела «Работа с файлами», которые

даны на лабораторной работе по теме «Строки и файлы» по параллельной

специализированной дисциплине «Основы алгоритмизации и

программирования».

**Дополнительное задание:**

Необходимо в общем задании реализовать дополнительно следующий

функционал:

- обеспечить вывода данных о студентах в алфавитном порядке по

Фамилии и имени;

- определить студентов, которые завалили сессию, т.е. хотя бы под одно

дисциплине не получили удовлетворительной оценки (для них средний

балл не высчитывается);

- добавить дисциплины, по которым велись экзамены и выявить лучших

студентов по каждой из дисциплин.

Требования к выполнению:

1) Необходимо спроектировать блок-схемы алгоритмов решений соответствую-

щих заданий согласно своему варианту (варианты назначаются преподавате-

лем или выбираются самостоятельно студентом, но ни один вариант не дол-

жен повторяться у студентов из одной группы) и на базе данных алгоритмов

разработать интерактивные консольные приложения с использованием архи-

тектурного шаблона проектирования ***MVC***.

2) Для каждого алгоритма решения задания оценить (вычислить) алгоритмиче-

скую сложность по необходимому времени и затрачиваемой памяти исполь-

зуя определение ***Big O***.

3) Для масштабируемости разрабатываемого программного решения размер

последовательности (списка) и его элементы должны задаваться пользовате-

лем во время выполнения программы или с помощью генератора псевдослу-

чайных чисел.

4) Для автоматизации заполнения различными значениями искомого контей-

нера рекомендуется использовать соответствующие функции генерирования

псевдослучайных чисел модуля ***random***.

5) Рекомендуется избегать использования глобальных переменных при написа-

нии основной логики приложения.

6) Если логически не подразумевается или в заданиях иного не указано, то вход-

ными и выходными данными являются вещественные числа (числа с плаваю-

щей запятой).

7) Все программы должны быть разбиты на отдельные функции. Среди данных

функций рекомендуется добавлять стартовую функцию ***main***, с которой лучше

производить запуск программы.

8) При выполнении заданий необходимо по максимуму пытаться разрабатывать

универсальный, масштабируемый, легко поддерживаемый и читаемый код.

9) Также рекомендуется придерживаться ***Single Responsibility Principle****,* ***SRP***

(принципа единственной ответственности) – постарайтесь вынести основную

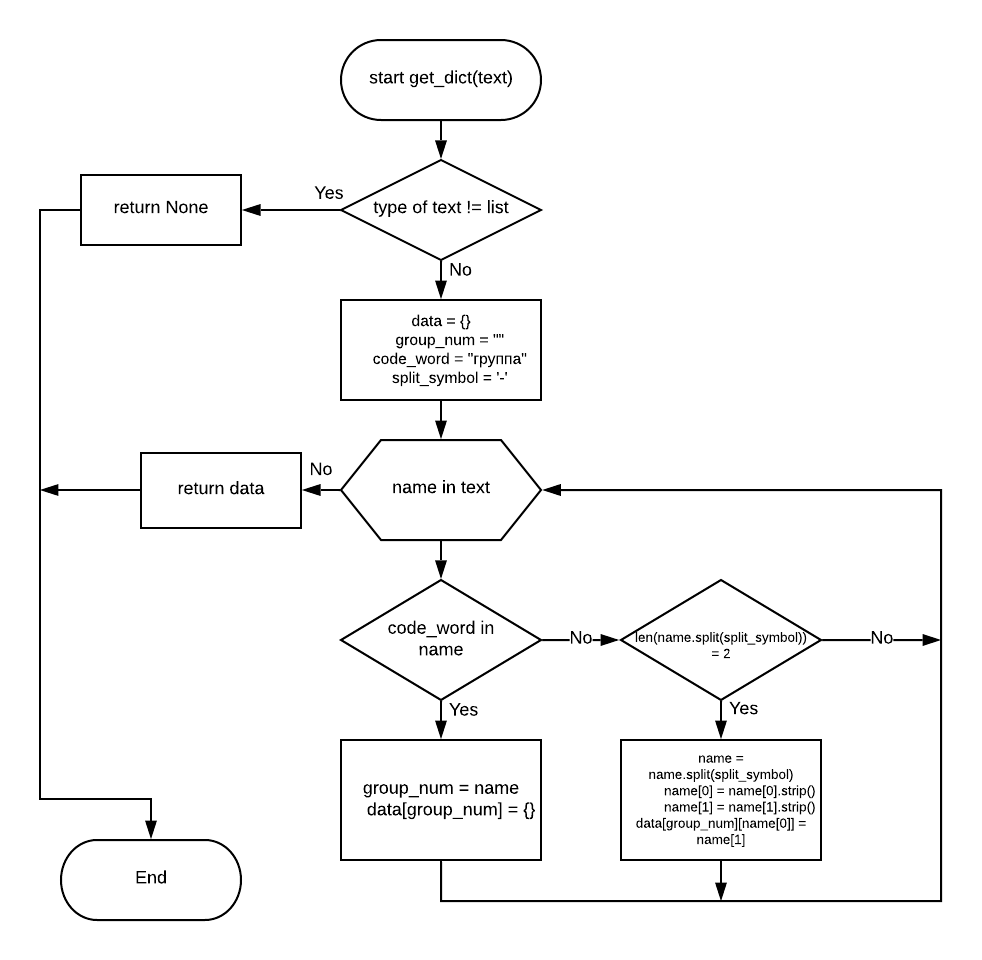
бизнес-логику задания в отдельную функцию (т.е. архитектура приложения

должна минимум состоять из нескольких функций).

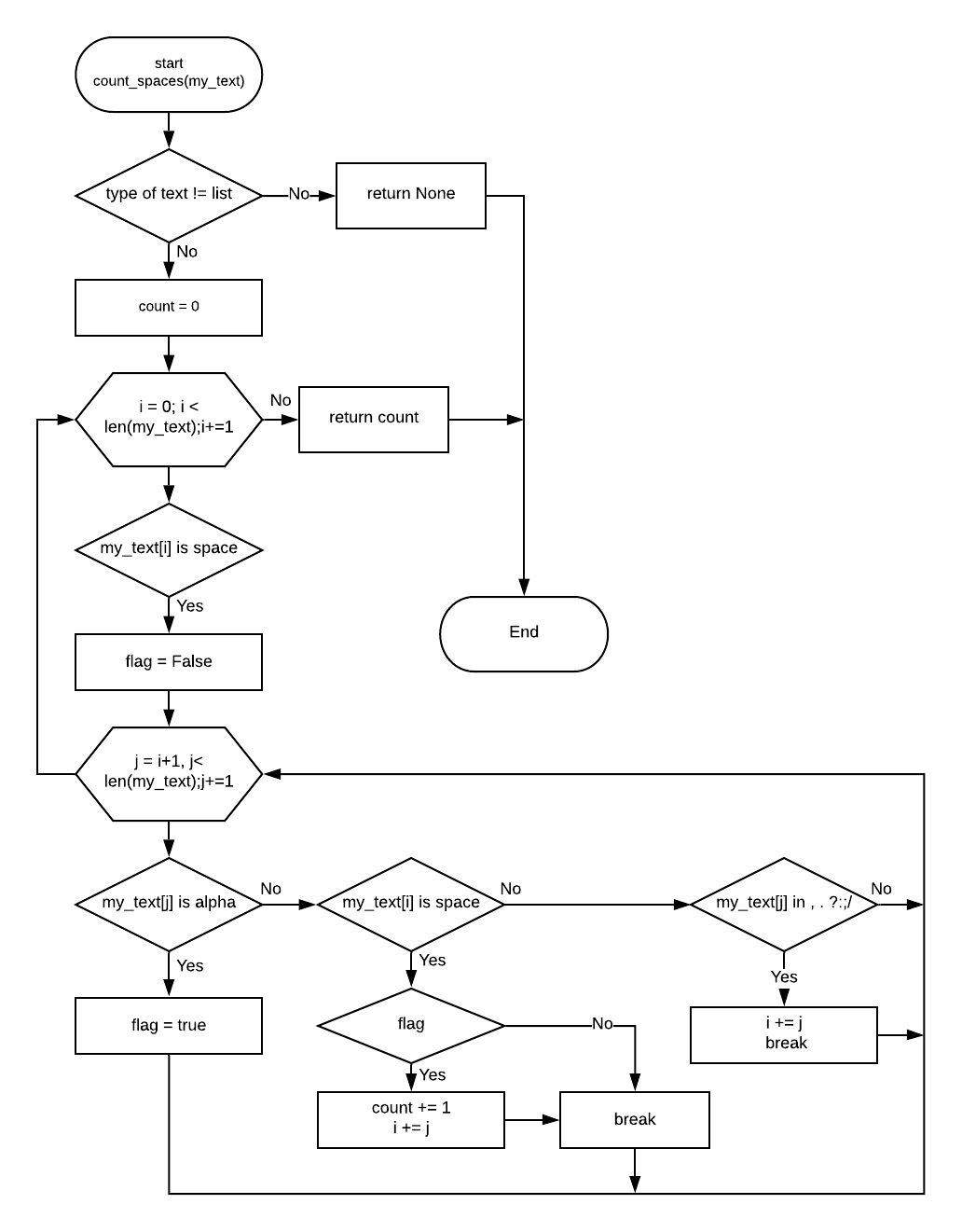
10) В соответствующих компонентах (функциях) бизнес-логики необходимо

предусмотреть «защиту от дурака».

**Блок-схемы алгоритмов:**



**Рис 24.1 Создание словаря с данными**



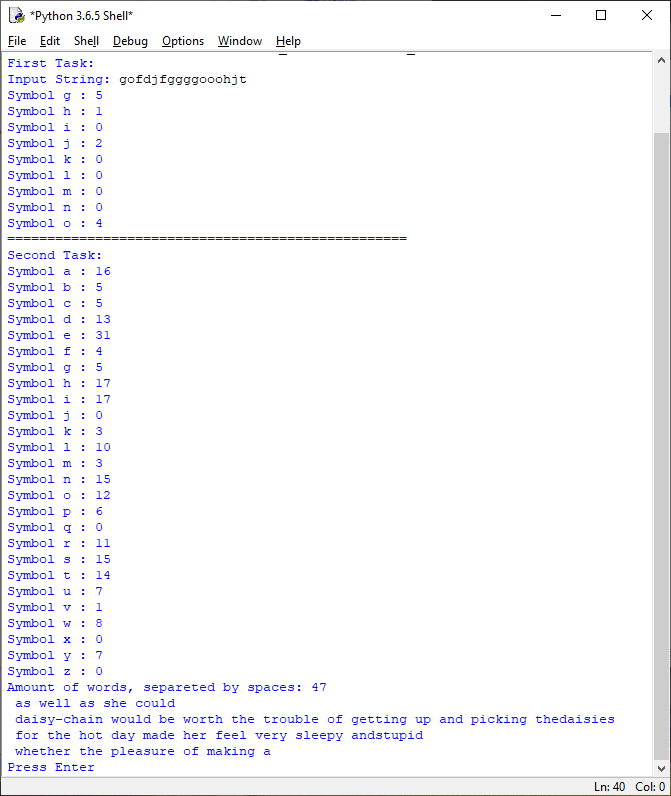
**Рис 24.2 Поиск кол-во слов, разд. пробелами**

Результаты выполнения основного задания:



**Рис 24.3 Входной и выходной файл основного задания**

Результаты выполнения индивидуального задания:



**Рис 24.4 Результаты выполнения индивидуального задания**

Ответы на контрольные вопросы:

Простейший файловый ввод-вывод в языке Python

1. Что такое файл в классическом его определении?

*Файл (англ. file) — именованная область данных на носителе информации.*

2. Какие существуют в мире программирования типы файлов? Опишите их

характеристики.

*Исполняемые(com, exe), Текстовые (txt,doc), Графические (bmp, jpg, png),*

*Звуковые (wav, mp3), Бинарные (.dat, .bin), Архивы и тд*

3. Что представляет собой в файловой системе директория (каталог, папка)?

*Поименованная совокупность байтов, на носителе информации, содержащая название подкаталогов и файлов*

4. Что такое дамп памяти?

*Дамп памяти — содержимое рабочей памяти одного процесса, ядра или всей операционной системы.*

5. Каков общий алгоритм работы с файлом?

*Создать Файл*

*Разместить файл*

*Найти файл*

*Открыть файл*

*Прочитать его*

6. Какие основные преимущества и недостатки у двоичных (бинарных) файлов?

*+ Они относительно компактны, с ними удобно работать в программе*

*- Сложны для чтения человеком*

7. Каков общий алгоритм работы программы с двоичными (бинарными)

файлами?

*Создать Файл*

*Разместить файл*

*Найти файл*

*Открыть файл*

*Прочитать его*

8. Какие основные преимущества и недостатки у текстовых файлов?

*-простота*

*- экономичность*

*- это открытый стандарт*

*Недостатки txt:*

*- при обмене данными между компьютерами различных платформ возможны конфликты кодировок*

*- не поддерживаются стили текста и графика*

9. Каков общий алгоритм работы программы с текстовыми файлами?

*описать переменную с путём к файлу*

*открыть файл с помощью функции open.*

*вывести информацию в файл.*

*обязательно закрыть файл.*

10. Как в языке Python получить дескриптор (файловый объект) на существующий

файл?

*Файловый дескриптор — это неотрицательное целое число. Когда создается новый поток ввода-вывода, ядро возвращает процессу, создавшему поток ввода-вывода, его файловый дескриптор.*

11. Какая существует встроенная функция в языке Python для связывания

файлового дескриптора с физическим фалом на диске?

Open()

12. Перечислите основные параметры, которая принимает встроенная в язык

*file : Строка или байтовая строка, представляющая путь (абсолютный, либо относительный для текущей рабочей директории) к файлу, который требуется открыть. 3.0 Либо целое, являющееся дескриптором файла (он будет закрыт при закрытии возвращаемого объекта ввода/вывода, при условии, что closefd установлен в True).*

*mode=r : Строка, обозначающая режим, в котором следует открыть файл.*

*buffering=-1 : Целое. Политика буферизации. 0 — отключить буферизацию (только для двоичного режима); 1 — построчная буферизация (только для текстового режима); > 1 — размер буфера в байтах. Если не указана, используются политика буферизации по умолчанию (см. ниже).*

*encoding=None : Имя кодировки для кодирования/декодирования файла. Следует указывать только для текстовых файлов. Если не указана, используется системная кодировка: для определения вызывается locale.getpreferredencoding(False). При чтении и записи двоичных файлов указывать кодировку не следует. Список поддерживаемых кодировок можно найти в модуле codecs.*

*errors=None : Строка, обозначающая режим, в котором будут обрабатываться ошибки кодирования/декодирования. Следует использовать только для текстовых файлов. В модуле codecs предопределено несколько режимов, пользовательские могут быть зарегистрированы при помощи codecs.register\_error().*

*newline=None : Строка, определяющая режим работы универсальных переводов строк. Следует использовать только для текстовых файлов. Варианты: None, пустая строка, \n, \r, и \r\n. None — при чтении потока будут использованы универсальные переводы строк (\n, \r и \r\n будут преобразованы в \n); при записи — \n преобразуются в разделитель, используемый в системе (os.linesep). Пустая строка — при чтении также используются универсальные переводы строк, однако символы не будут преобразованы в \n; при записи преобразование не происходит. Другие значения — при чтении разбивка на строки осуществляется при помощи значения, преобразование не производится; при записи \n преобразуются в указанное значение.*

*closefd=True : Флаг необходимости закрытия файлового дескриптора. Используется только, если в file указан дескриптор, иначе возбуждается исключение. Если False, то дескриптор будет оставлен открытым даже после закрытия файла.*

*opener=None : Пользовательский объект, поддерживающий вызов, который следует использовать для открытия файла. Этот объект получая на входе file и flags, должен возвращать открытый дескриптор файла (возврат os.open и None при этом функционально идентичны).*

13. Как в языке Python создать файл и получить на него ассоциативную ссылку?

*File = open(path, mode)*

14. Какие в языке Python используются режимы работы (r, w, a, +, t, b, …) с файлом

при получении файлового дескриптора?

*r Открывает файл только для чтения. Указатель стоит в начале файла.*

*rb Открывает файл для чтения в двоичном формате. Указатель стоит в начале файла.*

*r+ Открывает файл для чтения и записи. Указатель стоит в начале файла.*

*rb+ Открывает файл для чтения и записи в двоичном формате. Указатель стоит в начале файла.*

*w Открывает файл только для записи. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует.*

*wb Открывает файл для записи в двоичном формате. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует.*

*w+ Открывает файл для чтения и записи. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует.*

*wb+ Открывает файл для чтения и записи в двоичном формате. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует.*

*a Открывает файл для добавления информации в файл. Указатель стоит в конце файла. Создает файл с именем имя\_файла, если такового не существует.*

*ab Открывает файл для добавления в двоичном формате. Указатель стоит в конце файла. Создает файл с именем имя\_файла, если такового не существует.*

*a+ Открывает файл для добавления и чтения. Указатель стоит в конце файла. Создает файл с именем имя\_файла, если такового не существует.*

*ab+ Открывает файл для добавления и чтения в двоичном формате. Указатель стоит в конце файла. Создает файл с именем имя\_файла, если такового не существует.*

15. Каков по умолчанию режим доступа?

“r”

16. Для чего нужен буферизированный ввод-вывод?

*Манипуляции с буфером происходят быстрее, чем манипуляции с непосредственным источником данных (например, с жестким диском или сетевым ресурсом).*

*буферизованный поток – это поток, имеющий буфер.*

17. Является ли вывод функции print( ) буферизированный? Как это проверить?

*Да, является. Функция использует стандартный поток ввода вывода, который*

*Является буферезированным.*

18. Как можно получить информацию о файле, на который указывает файловый

дескриптор (объект) и какие именно характеристики можно прочитать?

(closed, mode, name, encoding, buffer, newlines, …)

19. Какие есть основные методы у файлового объекта для чтения данных и что

каждый из них делает? (read( ), readline( ), readlines( ))

read() – Прочитать весь файл

readline() – прочитать одну строку

readlines() – прочитать все строки

20. Какие есть основные методы у файлового объекта для записи данных и что

каждый из них делает? (write( ), writelines( ))

write() – записать одну строку

writelines() – записать несколько строк

21. Что именно записывают и читают методы write( ), writelines( )?

*Write() – данные*

*Writelines() - последовательность*

22. Что методы write( ), writelines( ) возвращают?

*Write() – кол-во записанных байт*

*Writelines() - ничего*

23. Какие есть ещё методы, которые можно вызвать у файлового объекта и зачем

они нужны? (tell( ), seek( ), …)

*tell() – возврощает кол-во байт в положении, где стоит курсов*

*seek() – переставить курсор*

*close() – закрыть файл*

*flush() – сбросить данные из буфера в файл*

*truncate() – обрезает файл до пред. Кол-ва символов*

24. С помощью какого метода файлового объекта можно завершить работу с

файлом?

*Close()*

25. Какое есть средство или структура в языке Python, которое предоставляет

предопределённые действия по очистке (закрытие файла также относится к

данному действию)?

*Конструкция with…as*

26. Как можно читать заново файл не закрывая и открывая его ещё раз?

*Нужно переставить курсор на начало файла file.seek(0)*

27. Если Python-программа работает с текстовыми файлами под управлением

операционной системы Windows, то какая кодировка используется по

умолчанию? Как это определить?

*Chcp1251. Файл будет прочтен некорректно, если открыть его с неправильной кодировкой*

28. Каким образом записывается в файл символ новой строки ('\n') в

операционной системе Windows? А в других операционных систем?

29. Какими дополнительными средствами стандартного модуля os языка Python

можно воспользоваться чтобы, к примеру, переименовать файл или удалить

его?

*Os.remove(), os.rename()*

30. Что предоставляет программисту стандартный модуль os языка Python для

работы с директориями (основные функции mkdir( ), chdir( ), getcwd( ),

*rmdir( ))?*

*Mkdir() – создаёт новый каталог*

*Rmdir() – Удаляет указанный каталог*

*Chdir() – делает указанный каталог текущим*

*Getcwd() – возвращает путь текущей папки*

*Listdir() – возвращает все файлы в текущей папке*

Обработка исключения в языке Python

1. Какие разновидности ошибок могут возникнуть в программах, разработанных

на языке Python (Syntax Errors, Logical Error и Exceptions)?

*Могут возникнуть все разновидности ошибок*

2. Что такое ошибка синтаксиса (ошибка интерпретации) в языке Python?

*Когда код не соответствует синтаксису языка*

3. Что такое логическая ошибка синтаксиса (ошибка алгоритма) в языке Python?

*Это вид ошибки, когда код выполняет не то, что ожидается*

4. Что такое исключение (ошибка выполнения) в языке Python?

*Ошибка, возникающая во время выполнения программы*

5. Какие ошибки зачастую очень сложно выявлять?

*Логические*

6. Каков жизненный цикл исключения в Python-программе?

*Только на время его перехвата или возникновения*

7. Каков основной механизм отлавливания и обработки исключений в языке

Python?

*Конструкция try…catch…else…finally*

8. Какой класс является базовым классом для всех типов исключений, которые

существуют в языке Python?

*BaseException*

9. Какой класс в языке Python является базовым классом для всех типов

пользовательских (обычных) исключений, которые могут возникнуть по вине

программы?

*Exception*

10. Какова главная структура перехвата и обработки исключения в языке Python?

*Try…except*

11. Как работает структура try-except?

*Try проверяет потенциально опасный код, а catch отлавливает определённое исключение и производит некоторые дейтсвия.*

12. Если необходимо обработать несколько исключений и для каждого свой

обработчик, как это сделать и сколько существует вариантов такой обработки

в языке Python?

*Можно указать кортеж с несколькими исключениями*

13. Есть ли разница, в какой последовательности перехватывать и обрабатывать

исключения?

*Да, есть.*

14. Если нужно обработать не все существующие исключения, а только

конкретные, и причём обработать одним способом, как это сделать в языке

Python?

Создать несколько except, в которых содержатся разные типы исключений.

15. Как или с помощью какой структуры можно удостовериться, что при

выполнения заданного участка кода исключительной ситуации не произошло?

(try-except-else)

*else*

16. Если необходимо написать код, который должен выполняться всегда и не

важно, выкинулось ли исключение в заданном коде или нет, при это не

хотелось бы повторений кода. Как это сделать? (try-except-finally)

*finally*

17. Когда обычно используется структура try-except-finally?

*Если мы используем файл и вне зависимости от выполнения потенциально опасного кода на его надо закрыть*

18. Всегда ли выполняется блок finally?

*Да*

19. Что можно сказать про поведение блока finally, если в блоках до него

встречаются операторы разрыва: break, continue и return?

*Он все равно будет выполнен*

20. Можно ли размещать другие инструкции между блоками try-except-elsefinally?

*Нет*

21. Что такое аргумент объекта-исключения и как его получить?

*Это строковое описание ошибки. Получить его можно через except*

*(except Exception as msg)*

22. Как вывести информацию, которая была записана в объект-исключения при

его непосредственном создании?

*except Exception as msg:*

*print(msg)*

23. Перечислите наиболее распространённые ошибки/исключения в языке

Python и по каким причинам они возникают? (BaseException, Exception,

AssertionError, EOFError, AttributeError, NameError, RuntimeError,

SyntaxError, TypeError, ValueError, ArithmeticError, …)

*SyntaxError(Ошибка синтаксиса), ValueError (переданный параметр не соответствует ожидаемому значению), IndexError(неверный индекс), TypeError (тип объекта не соответствует ожидаемому), NameError (Обращение к переменной до её обращения)*

24. Как генерировать исключения вручную в коде с использованием синтаксиса

языка Python?

*При помощи команды raise [Исключение]*

25. Что в качестве аргументов может использовать оператор raise?

*Класс ошибки и её описание*

26. Каким способом и как можно повторно выкинуть отловленное исключение?

*Except Exception as err:*

*Raise err*

27. Как написать универсальный обработчик всех исключений, которые могут

выскочить в программе, написанной на языке Python? Стоит ли этим

пользоваться всегда?

*Try:*

*###*

*Except BaseException:*

*###*

*Нет, не стоит.*

28. Зачем нужна инструкция assert? Как она работает?

*Она нужна для тестирования программы. Её передаётся условие, которое проверяется*

*И в случае неверности условия вызывается AssertationError*

29. Какой тип исключения генерирует инструкция assert?

*AssertationError*

30. Сколько и какие аргументы принимает инструкция assert?

*Условие и сообщение об ошибке*

Что я освоил в процессе выполнения лабораторной работы (выводы):

Мной были получены знания о том, что такое файлы, и как с ними работать в языке программирования Python. Также были получены знания об правильном использовании конструкции try…except…else…finally в программах и приложениях.

*ПРИЛОЖЕНИЕ A*

Листинг исходных кодов программ

Файл task\_main.py

# Программа считывает данные из файла с успеваемостью студентов

# обрабатывает данные и создает новый файл, который содержит

# результаты по каждому студенту

#

# Version: 1.0

# Group: 10701219

# Author: Колосов Артём Александрович

# Date: 16.4.2020

def output\_to\_user(msg, s=' ', e='\n'):

print(msg, sep=s, end=e)

def middle(marks):

if not isinstance(marks, str):

return None

result = 0

try:

marks = list(map(int, marks.split()))

result = sum(marks) / len(marks)

except ZeroDivisionError as msg:

output\_to\_user(msg)

except ValueError as msg:

output\_to\_user(msg)

except TypeError as msg:

output\_to\_user(msg)

return result

def add\_main\_data(my\_file, data):

if not isinstance(data, dict):

return None

for key in data.keys():

my\_file.write("{0} {1}\n".format(key, get\_marks\_of\_group(data, key)))

for next\_key in data[key]:

my\_file.write("{0} - {1}\n".format(next\_key, middle(data[key][next\_key])))

my\_file.write("\n")

def get\_marks\_of\_group(data, data\_key):

if not isinstance(data, dict):

return None

group\_middle = 0

num = 0

result = 0

for next\_data\_key in data[data\_key]:

group\_middle += middle(data[data\_key][next\_data\_key])

num += 1

try:

result = round(group\_middle / num, 2)

except ZeroDivisionError as msg:

output\_to\_user(msg)

return result

def add\_best\_worse\_students(data, my\_file, message, mark\_key\_start=0, mark\_key\_end=10):

if not (isinstance(data, dict) and isinstance(message, str)):

return None

my\_file.write(message + "\n")

count = 0

for key in data.keys():

for next\_key in data[key]:

mark = middle(data[key][next\_key])

if mark\_key\_start <= mark <= mark\_key\_end:

count += 1

my\_file.write("{0}) {1} - {2}\n".format(count, next\_key, mark))

my\_file.write("\n")

def get\_dict(text):

if not isinstance(text, list):

return None

data = {}

group\_num = ""

code\_word = "группа"

split\_symbol = '-'

for name in text:

if code\_word in name.lower():

group\_num = name

data[group\_num] = {}

elif len(name.split(split\_symbol)) == 2:

name = name.split(split\_symbol)

name[0] = name[0].strip()

name[1] = name[1].strip()

data[group\_num][name[0]] = name[1]

return data

def main():

path\_in = r"in.txt"

path\_out = r"out.txt"

encode = "1251"

try:

file = open(path\_in, "r", encoding=encode)

except FileNotFoundError as msg:

output\_to\_user("Error: {0}".format(msg))

input()

exit(1)

string = file.read().split("\n")

all\_data = get\_dict(string)

new\_file = open(path\_out, "w", encoding=encode)

add\_main\_data(new\_file, all\_data)

add\_best\_worse\_students(all\_data, new\_file, "Студенты, которым надо подсобраться:", 0, 4)

add\_best\_worse\_students(all\_data, new\_file, "Лучшие студенты потока:", 9, 10)

file.close()

new\_file.close()

input("Programm ended succesfully!")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Файл task\_oaip.py

# Программа подсчитывает кол-во символов в диапазоне от 'g' до 'o'

# Подсчитывает кол-во каждого символа в тексте

# Находит предложения, разделённые с двух сторон запятыми и

# Сортирует их по алфавиту

# Считает кол-во слов, разделённых с двух сторон пробелами

# Version: 1.0

# Group: 10701219

# Author: Колосов Артём Александрович

# Date: 16.4.2020

def output\_to\_user(msg, s=' ', e='\n'):

print(msg, sep=s, end=e)

def count\_symbol\_codes(string, symb\_one='g', symb\_two='o'):

if not (isinstance(string, str) and isinstance(symb\_one, str) or isinstance(symb\_two, str)):

return None

count = 0

first\_code = ord(symb\_one)

second\_code = ord(symb\_two)

for symbol in string:

if first\_code <= ord(symbol) <= second\_code:

count += 1

return count

def count\_symbol\_alternate(string, symbol\_one='g', symbol\_two='o', key=lambda s: s):

if not (isinstance(string, str) and isinstance(symbol\_one, str) and isinstance(symbol\_two, str)):

return None

first\_code = ord(key(symbol\_one))

second\_code = ord(key(symbol\_two))

for code in range(first\_code, second\_code + 1):

output\_to\_user("Symbol {0} : {1}".format(chr(code), string.count(key(chr(code)))))

def selection\_sort(text):

if not isinstance(text, list):

return None

index = 0

for i in range(len(text) - 1):

index = i

for j in range(i + 1, len(text)):

if strcmp(text[j], text[index]) < 0:

index = j

if index != i:

text[i], text[index] = text[index], text[i]

return text

def strcmp(str1, str2):

if not (isinstance(str1, str) and isinstance(str2, str)):

return None

smallest\_len = len(str1) if len(str1) <= len(str2) else len(str2)

return\_code = 0

str1 = str1.strip()

str2 = str2.strip()

for i in range(0, smallest\_len):

if str1[i] != str2[i]:

return\_code = 1 if ord(str1[i]) > ord(str2[i]) else -1

break

return return\_code

def count\_spaces(my\_text):

if not isinstance(my\_text, str):

return None

count = 0

for i in range(len(my\_text)):

if my\_text[i].isspace():

flag = False

for j in range(i + 1, len(my\_text)):

if my\_text[j].isalpha():

flag = True

elif my\_text[j].isspace():

if flag:

count += 1

i += j

break

elif my\_text[j] in (",", ".", "?", ":", ";", "/"):

i += j

break

return count

def main():

output\_to\_user("First Task:")

count\_symbol\_alternate(input("Input String: "))

output\_to\_user("="\*50)

output\_to\_user("Second Task:")

text = """So she was considering in her own mind, as well as she could, for the hot day made her feel very sleepy and

stupid, whether the pleasure of making a , daisy-chain would be worth the trouble of getting up and picking the

daisies, when suddenly a White Rabbit with pink eyes ran close by her."""

count\_symbol\_alternate(text, key=lambda s: s.lower(), symbol\_one='a', symbol\_two='z')

text = text.replace("\n", "")

words = text.split(",")

words = words[1:len(words) - 1]

output\_to\_user("Amount of words, separeted by spaces: {0}".format(count\_spaces(text)))

words = selection\_sort(words)

for word in words:

output\_to\_user(word)

input("Press Enter")

if \_\_name\_\_ == "\_\_main\_\_":

main()