# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

**ФЕДЕРАЦИИ**

# Федеральное государственное автономное образовательное учреждение высшего образования

**«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

# Кафедра инфокоммуникаций Институт цифрового развития

**ОТЧЁТ**

# по лабораторной работе №11

Дисциплина: «Основы программной инженерии»

Тема: «Работа с функциями в языке Python»

Выполнил: студент 2 курса группы Пиж-б-о-21-1

Рязанцев Матвей Денисович

Ставрополь 2022

Цель работы: приобретение навыков по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

**Выполнение работы**

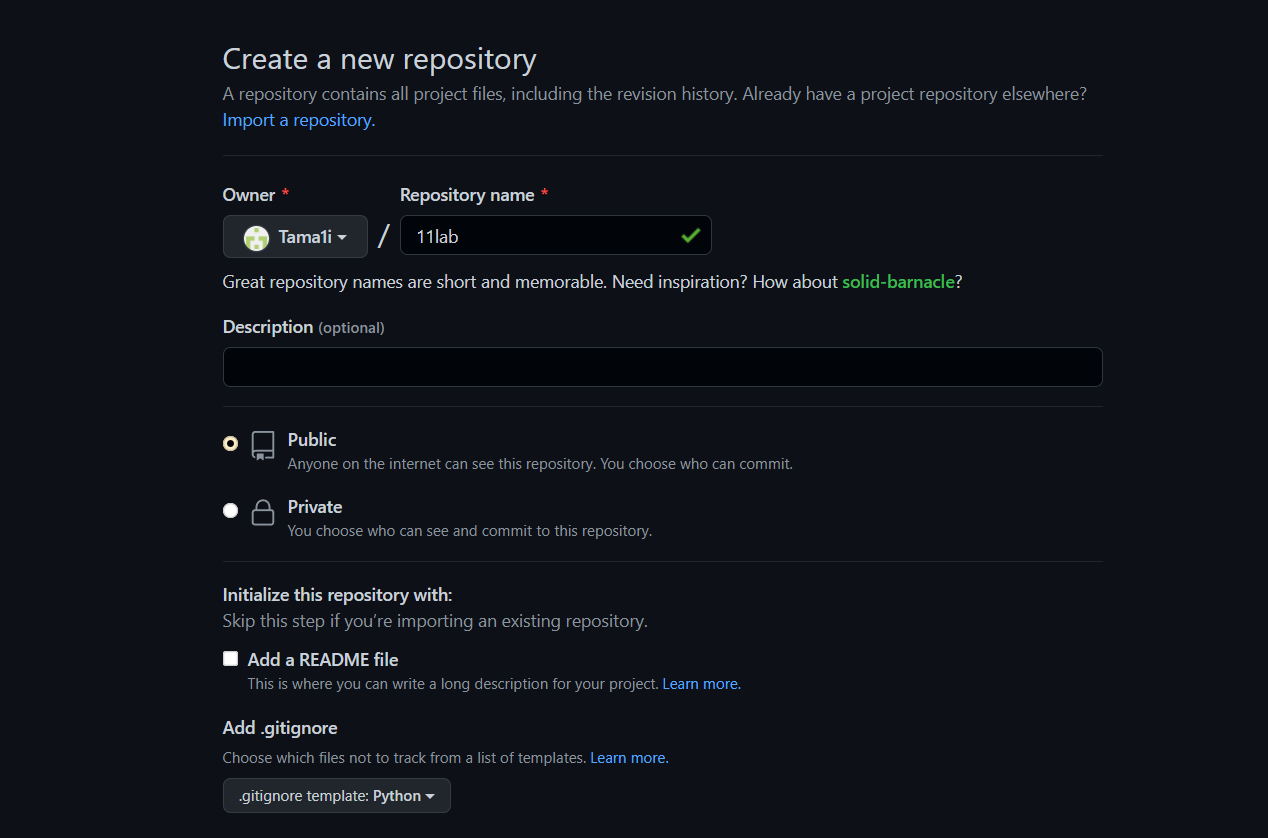


Рисунок 1 -создание репозитория

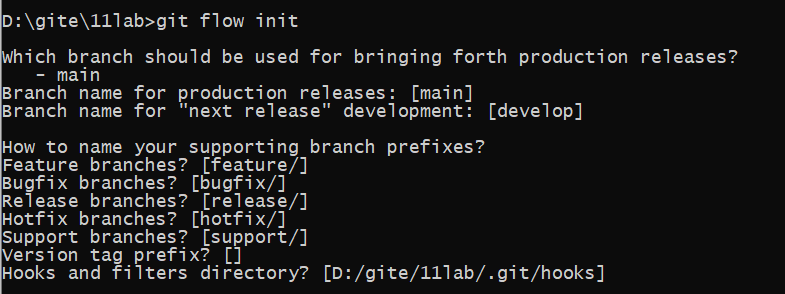


Рисунок 2 - Организация репозитория по модели ветвления git flow

Код общего задания:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

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

# Определим универсальное множество

u = set("abcdefghijklmnopqrstuvwxyz")

a = {"b", "c", "h", "o"}

b = {"d", "f", "g", "o", "v", "y"}

c = {"d", "e", "j", "k"}

d = {"a", "b", "f", "g"}

x = (a.intersection(b)).union(c)

print(f"x = {x}")

# Найдем дополнения множеств

bn = u.difference(b)

cn = u.difference(c)

y = (a.difference(d)).union(cn.difference(bn))

print(f"y = {y}")

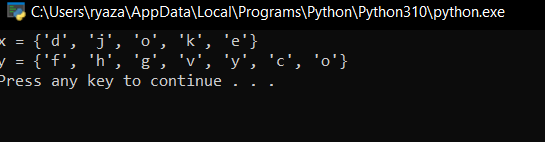


Рисунок 1 – результат выполнения программы

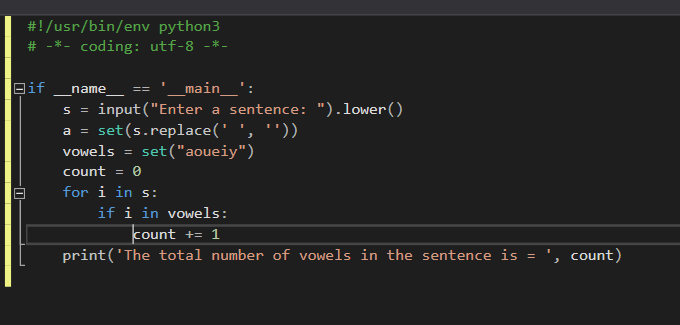


Рисунок 2 – код программы задание 1

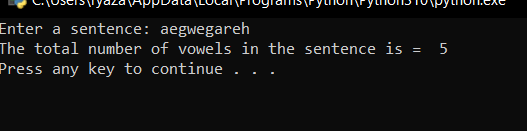


Рисунок 3 – результат работы программы

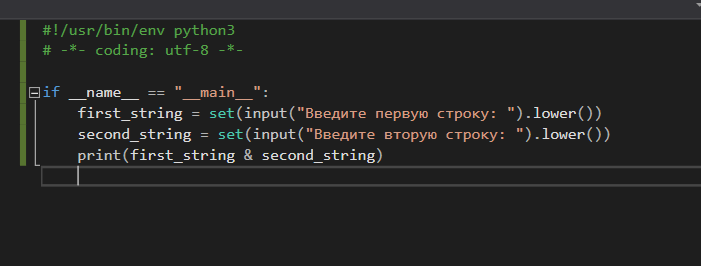


Рисунок 4 – код программы задание 2

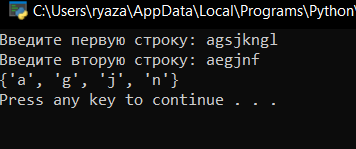


Рисунок 5 – результат работы программы

Индивидуальное задание

Код программы:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import sys

def add(pep):

# Запросить данные о работнике.

name = input("name faname? ")

num = int(input("number? "))

br = int(input("burftday? "))

# Создать словарь.

chel = {

'name': name,

'num': num,

'br': br,

}

# Добавить словарь в список.

pep.append(chel)

# Отсортировать список в случае необходимости.

if len(pep) > 1:

pep.sort(key=lambda item: item.get('br',''))

return pep

def li(pep):

line = '+-{}-+-{}-+-{}-+-{}-+'.format(

'-' \* 4,

'-' \* 30,

'-' \* 20,

'-' \* 8

)

print(line)

print(

'| {:^4} | {:^30} | {:^20} | {:^8} |'.format(

"№",

"F.I.O.",

"NUMBER",

"BRDAY"

)

)

print(line)

for idx, chel in enumerate(pep, 1):

print(

'| {:>4} | {:<30} | {:<20} | {:>8} |'.format(

idx,

chel.get('name', ''),

chel.get('num', ''),

chel.get('br', 0)

)

)

print(line)

def sel(pep):

# Получить требуемый стаж.

zapros = int(input("zapros po numeru "))

# Инициализировать счетчик.

count = 0

# Проверить сведения работников из списка.

for chel in pep:

if chel.get('num') == zapros:

count += 1

print(

'{:>4}: {}'.format(count, chel.get('name', ''))

)

# Если счетчик равен 0, то работники не найдены.

if count == 0:

print("cheela s takim nomerom net")

from datetime import date

if \_\_name\_\_ == '\_\_main\_\_':

# Список работников.

pep = []

# Организовать бесконечный цикл запроса команд.

while True:

# Запросить команду из терминала.

command = input(">>> ").lower()

# Выполнить действие в соответствие с командой.

if command == 'exit':

break

elif command == 'add':

pep = add(pep)

elif command == 'list':

li(pep)

elif command == 'select':

sel(pep)

elif command == 'help':

# Вывести справку о работе с программой.

print("Список команд:\n")

print("add - add chel;")

print("list - show list of pep;")

print("select <стаж> - запросить работников со стажем;")

print("help - отобразить справку;")

print("exit - завершить работу с программой.")

else:

print(f"Неизвестная команда {command}", file=sys.stderr)

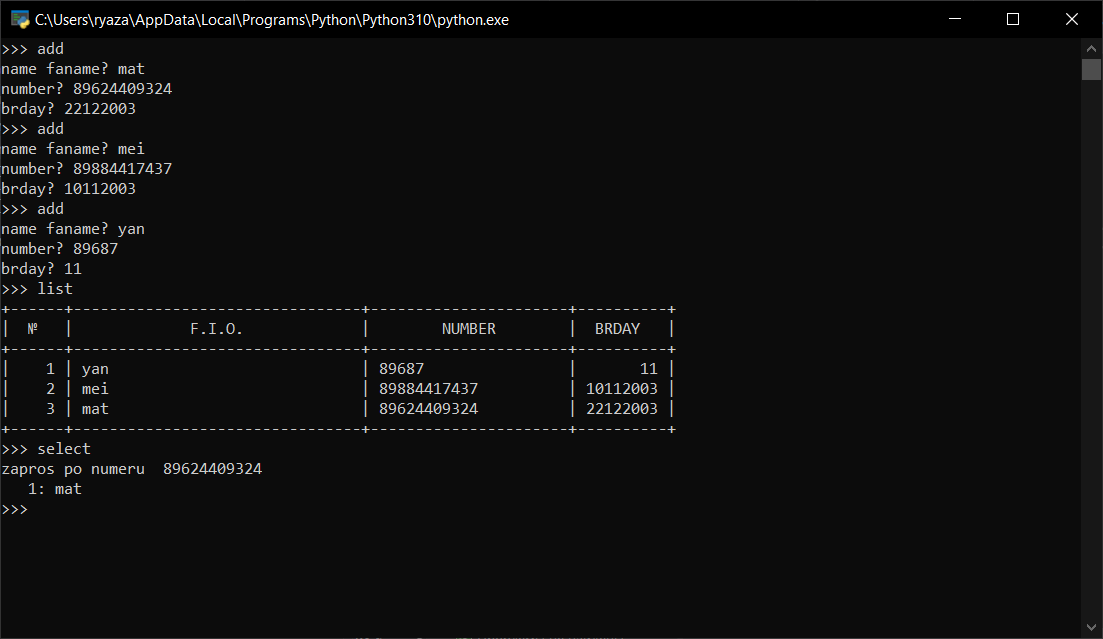


Рисунок 6 – результат работы программы идз

**Контрольные вопросы**

1. Каково назначение функций в языке программирования Python?

Функция – это средство (способ) группирования фрагментов программного кода таким образом, что этот программный код может вызваться многократно с помощью использования имени функции. Использование функций в программах на Python даёт следующие взаимосвязанные преимущества: избежание повторения одинаковых фрагментов кода в разных частях программы; уменьшение избыточности исходного кода программы. Как следствие, уменьшение логических ошибок программирования; улучшенное восприятие исходного кода программы в случаях, где вместо блоков многочисленных инструкций (операторов) вызываются имена готовых протестированных функций. Это, в свою очередь, также уменьшает количество ошибок; упрощение внесения изменений в повторяемых блоках кода, организованных в виде функций. Достаточно внести изменения только в тело функции, тогда во всех вызовах данной функции эти изменения будут учтены; с помощью функций удобно разбивать сложную систему на более простые части. Значит, функции – удобный способ структурирования программы; уменьшение трудозатрат на программирование, а, значит, повышение производительности работы программиста.

1. Каково назначение операторов def и return?

Оператор def, выполняемый внутри определения функции, определяет локальную функцию, которая может быть возвращена или передана. Свободные переменные, используемые во вложенной функции, могут обращаться к локальным переменным функции, содержащей def.

Оператор return [выражение] возвращает результат из функции. Оператор return без аргументов аналогичен return None

1. Каково назначение локальных и глобальных переменных при написании функций в Python?

Области видимости определяют, в какой части программы мы можем работать с той или иной переменной, а от каких переменная «скрыта».

Так глобальные переменные доступны в любой точке программы, а локальные переменные, только в функциях, где они объявлены.

4. Как вернуть несколько значений из функции Python?

С помощью оператора return. Чтобы вернуть несколько значений, нужно написать их через запятую.

5. Какие существуют способы передачи значений в функцию?

Существует два способа передачи параметров в функцию: по значению и по адресу. При передаче по значению на месте формальных параметров записываются имена фактических параметров. При вычислении функции в стек заносятся копии значений фактических параметров, и операторы функции работают с этими копиями.

6. Как задать значение аргументов функции по умолчанию?

В Python аргументам функции можно присваивать значения по умолчанию, используя оператор присваивания «=».

7. Каково назначение lambda-выражений в языке Python?

Лямбда-выражения на Python - конструкторы простых безымянных однострочных функций. Могут быть использованы везде, где требуется.

8. Как осуществляется документирование кода согласно PEP257?

PEP 257 описывает соглашения, связанные со строками документации python, рассказывает о том, как нужно документировать python код. Цель этого PEP - стандартизировать структуру строк документации: что они должны в себя включать, и как это написать (не касаясь вопроса синтаксиса строк документации). Этот PEP описывает соглашения, а не правила или синтаксис.

9. В чем особенность однострочных и многострочных форм строк документации?

Одиночные строки документации предназначены для действительно очевидных случаев.

Многострочные строки документации состоят из однострочной строки документации с последующей пустой строкой, а затем более подробным описанием. Первая строка может быть использована автоматическими средствами индексации, поэтому важно, чтобы она находилась на одной строке и была отделена от остальной документации пустой строкой. Первая строка может быть на той же строке, где и открывающие кавычки, или на следующей строке. Вся документация должна иметь такой же отступ, как кавычки на первой строке.

) Список – a = {1, 2, 0, 1, 3, 2} b = list(a)