МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Департамент цифровых, робототехнических систем и электроники

«Работа с функциями в языке Python»
Отчет по лабораторной работе № 6
по дисциплине «Программирование на Python»
Вариант 9

Выполнил студент группы ИВТ-б-о-24-1
Каиров Вадим Сосланович
«__» ноября 2025г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2025

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

Ход работы:

Ссылка на GitHub: https://github.com/ozetin/labochka6

Создаём удалённый репозиторий в Github и клонируем его к себе на компьютер.

Пишем в нём все программы и примеры из лабораторной работы и делаем коммиты.

Сначала были изучены приведенные примеры.

Пример 1.

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

import sys

def median(*args):
    if args:
        values = [float(arg) for arg in args]
        values.sort()

        n = len(values)
        idx= n // 2
        if n % 2:
            return values[idx]
        else:
            return (values[idx - 1] + values[idx]) / 2

    else:
        return None

if __name__ == "__main__":
    print(median())
    print(median(3, 7, 1, 6, 9))
    print(median(1, 5, 8, 4, 3, 9))
```

Пример 2.

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-
```

```
import sys
from datetime import date

def get_worker():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))

    #Создать словарь
    return {
        "name": name,
        "post": post,
        "year": year,
    }
def display_workers(staff):
    """
    Отобразить список работников.
    """
    #Проверить, что список работников не пуст.
    if staff:
        line="+-{}-+-{}-+-{}-+-{}-+".format(
        '-' * 4,
        '-' * 30,
        '-' * 20,
        '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)

        for idx, worker in enumerate(staff, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
```

```
                worker.get("name", ''),
                worker.get("post", ''),
                worker.get("year", 0)
            )
        )
        print(line)

    else:
        print("Список работников пуст.")

def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    today = date.today()

    result = []
    for employee in staff:
        if today.year - employee.get("year", employee.get("year")) >= period:
            result.append(employee)

    return result

def main():
    """
    Главная функция программы
    """
    workers = []

    while True:
        command = input(">>> ").lower()

        if command == "exit":
            break

        elif command == "add":
            worker = get_worker()
            workers.append(worker)

            if len(workers) > 1:
                workers.sort(key=lambda item: item.get("name", ''))

        elif command == "list":
```

```
            display_workers(workers)

        elif command.startswith("select"):
            parts = command.split(" ", maxsplit=1)
            period = int(parts[1])

            selected = select_workers(workers, period)
            display_workers(selected)

        elif command == 'help':
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - внести список работников;")
            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)
            return 1

    return 0

if __name__ == "__main__":
    sys.exit(main())
```

Далее были выполнены все индивидуальные задания.

Задание 1. Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам маршрутов; вывод на экран информации о маршруте, номер которого введен с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
```

```
marshruti = []
def add(start, end, number):
    global marshruti
    imeetsya = 1

    for i in marshruti:
        if i['number'] == number:
            imeetsya = 0
            print("Маршрут с таким номером уже есть")
            break
    if imeetsya:
        marshrut = {
            'start': start,
            'end': end,
            'number': number
        }

        marshruti.append(marshrut)
        marshruti = sorted(marshruti, key=lambda k: k['number'])

def vivod(number):
    schet = 0
    global marshruti

    for i in marshruti:
        if i['number'] == number:
            print(i)
            schet += 1

    if schet == 0:
        print("Таких маршрутов не найдено")

def list():
    global marshruti
    if marshruti:
        for i in marshruti:
            print(f"Номер: {i['number']}\n"
                  f"Начало: {i['start']}\n"
                  f"Конец: {i['end']}\n")
    else:
        print("Маршрутов нет")


while True:
    chto = input("Что сделать?\n "
```

```
                "vivod - вывести маршрут по номеру\n "
                "add - добавить маршрут\n"
                "list - вывести список маршрутов\n")

        if chto == "vivod":
            vivod(int(input("Введите номер маршрута: ")))

        elif chto == "list":
            list()

        elif chto == "add":
            start = input("Введите начальный пункт маршрута: ")
            end = input("Введите конечный пункт маршрута: ")
            number = int(input("Введите номер маршрута: "))
            add(start, end, number)
```

```
C:\Users\vadim\AppData\Local\Programs\Python\Python313\python.exe
Что сделать?
 vivod - вывести маршрут по номеру
 add - добавить маршрут
add
Введите начальный пункт маршрута: Ставрополь
Введите конечный пункт маршрута: Владикавказ
Введите номер маршрута: 3
Что сделать?
 vivod - вывести маршрут по номеру
 add - добавить маршрут
vivod
Введите номер маршрута: 3
{'start': 'Ставрополь', 'end': 'Владикавказ', 'number': 3}
Что сделать?
 vivod - вывести маршрут по номеру
 add - добавить маршрут
```

Рисунок 1. Результат работы программы

Задание 2. Создать функцию evaluate(*args, **kwargs), которая принимает произвольное количество чисел и операции:

op=”add” - сложение

op=”sub” - вычитание

op=”mul” - умножение

op=”div” – деление

Если операция неизвестная – вернуть None.

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    def evaluate(*args, **kwargs):
        for kluch, znach in kwargs.items():
            if znach == "add":
                print(sum(args))

            elif znach == "sub":
                vich = args[0]
                for i in args[1:]:
                    vich -= i
                print(vich)

            elif znach == "mul":
                proizv = 1
                for i in args:
                    proizv *= i
                print(proizv)

            elif znach == "div":
                vich = args[0]
                for i in args[1:]:
                    vich /= i
                print(vich)

            else:
                print("Неизвестная команда", file=sys.stderr)

    while True:
        print("Доступные операции:\n"
              "add - сложение\n"
              "sub - вычитание\n"
              "mul - умножение\n"
              "div - деление\n")
        operation = input("Введите операцию: ")
        chisla = list(map(int, input("Введите числа через пробел: ").split()))
        evaluate(*chisla, op=operation)
```

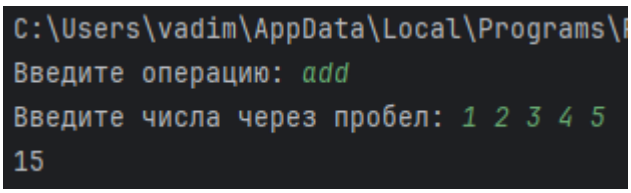

Рисунок 2. Результат работы программы

Задание 3. Реализуйте функцию hanoi(n, src, dst, aux), которая печатает шаги для перемещения n дисков с башни src на dst, используя aux как вспомогательную.

```
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    schet = 0
    def hanoi(n, src, dst, aux):
        global schet
        if n == 1:
            schet += 1
            print(f"Move {schet} from {src} to {dst}")
            return

        hanoi(n - 1, src, aux, dst)
        schet += 1
        print(f"Move {schet} from {src} to {dst}")
        hanoi(n - 1, aux, dst, src)

    n = int(input("Введите количество дисков: "))
    bashni = list(input("Введите названия 3-х башен: ").split())
    hanoi(n,bashni[0],bashni[1],bashni[2] )
```

```
Введите количество дисков: 4
Введите названия 3-х башен: pervaya vtoraya tretia
Move 1 from pervaya to tretia
Move 2 from pervaya to vtoraya
Move 3 from tretia to vtoraya
Move 4 from pervaya to tretia
Move 5 from vtoraya to pervaya
Move 6 from vtoraya to tretia
Move 7 from pervaya to tretia
Move 8 from pervaya to vtoraya
Move 9 from tretia to vtoraya
Move 10 from tretia to pervaya
Move 11 from vtoraya to pervaya
Move 12 from tretia to vtoraya
Move 13 from pervaya to tretia
Move 14 from pervaya to vtoraya
Move 15 from tretia to vtoraya
```

Рисунок 3. Результат работы программы

Контрольные вопросы:

1) Каково назначение функций в языке программирования Python?

Функции – это обособленный участок кода, который выполняет команды тела функции при вызове.

2) Каково назначение операторов def и return?

С помощью def функции определяются.

С помощью return можно указать, что будет возвращать функция. Также return завершает работу функции.

3) Каково назначение локальных и глобальных переменных при написании функций в Python?

Локальные переменные видны только в локальной области видимости, которой может выступать отдельно взятая функция. Глобальные переменные видны во всей программе.

4) Как вернуть несколько значений из функции Python?

После return возвращаемые значения написать через запятую. Она вернёт объект типа tuple.

5) Какие существуют способы передачи значений в функцию?

Позиционными аргументами и именованными.

6) Как задать значение аргументов функции по умолчанию?

После параметра написать равно и значение по умолчанию.

7) Каково назначение lambda-выражений в языке Python?

Они позволяют в одну строку написать простую функцию.

8) Как осуществляется документирование кода согласно PEP257?

Документирование кода выполняется с помощью строк документации для всех публичных модулей, функций, классов и методов. Для однострочной документации строка должна быть заключена в тройные кавычки ("""), описывать действие в императивном тоне, а закрывающие кавычки могут быть на той же строке. Для многострочной документации закрывающие кавычки помещаются на отдельной строке, часто после пустой строки, а текст располагается между ними.

9) В чем особенность однострочных и многострочных форм документации?

Однострочные формы документации используются для кратких, отдельных сведений, редактируемых по отдельности, в то время как многострочные формы предназначены для текста, который нужно рассматривать как единый блок, например, для развернутых описаний, инструкций или абзацев.

Многострочные формы документации: Представлены несколькими строками или абзацами. Предназначены для более подробных объяснений, описаний, инструкций или документации. Обычно используют для полноценной документации, описания функций, руководств.

10) Какие аргументы называются позиционными в Python?

В Python позиционные аргументы — это аргументы, которые передаются в функцию в определённом порядке и соответствуют позициям параметров, объявленных в определении функции.

11) Какие аргументы называются именованными в Python?

Именованные аргументы — это аргументы, для которых явно указано имя параметра при вызове функции, что обеспечивает соответствие значения конкретному параметру вне зависимости от порядка.

12) Для чего используется оператор *?

* позволяет распаковывать объекты, внутри которых хранятся некие элементы.

13) Каково назначение конструкции *args и **kwargs?

*args – создает список позиционных аргументов на основе того, что было передано функции при вызове.

**kwargs – создаётся словарь, в котором содержатся именованные аргументы, переданные функции при её вызове.

14) Для чего нужна рекурсия?

Рекурсия — это метод решения задач, в котором функция вызывает сама себя для выполнения части задачи, обычно с более простыми входными данными. Она используется для решения задач, которые удобно разбивать на более маленькие, подобные исходным.

15) Что называется базой рекурсии?

Базой рекурсии называется условие завершения рекурсивных вызовов, при котором рекурсия прекращается.

16) Самостоятельно изучите что является стеком программы. Как используется стек программы при вызове функций?

Стек программы — это область памяти, которая используется для хранения информации о текущем состоянии выполнения программы, включая данные о вызовах функций, локальных переменных, параметрах функций и адресах возврата.

Когда вызывается функция в стек помещается кадр вызова — это так называемый кадр стека или стековый кадр. В этот кадр записываются локальные переменные функции, параметры функции, адрес возврата — место в программе, куда нужно вернуться после завершения функции.

17) Как получить текущее значение максимальной глубины рекурсии в языке Python?

В языке Python для получения текущего значения максимальной глубины рекурсии используется модуль sys. Конкретно для этого предназначена функция sys.getrecursionlimit().

18) Что произойдет, если число рекурсивных вызовов превысит максимальную глубину рекурсии в языке Python?

Произойдет ошибка — RecursionError

19) Как изменить максимальную глубину рекурсии в языке Python?

```
import sys
sys.setrecursionlimit(1000)
```

20) Каково назначение декоратора lru_cache?

Декоратор lru_cache из модуля functools используется для кэширования результатов вызовов функций, чтобы повысить их эффективность.

21) Что такое хвостовая рекурсия? Как проводится оптимизация хвостовых вызовов?

Хвостовой вызов – это просто вызов рекурсивной функции, который является последней операцией и должна быть выполнена перед возвратом значения.

Вывод: в ходе данной работы были приобретены навыки по работе с функциями при написании программ с помощью языка программирования Python версии 3.x.