

**МИНИСТЕРСТВО 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 №3

Дисциплина: «Программирование на Python»

Тема: «Работа с функциями в языке Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

Уланбекова Айканыш Уланбековна

Ставрополь 2022

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

Порядок выполнения работы:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

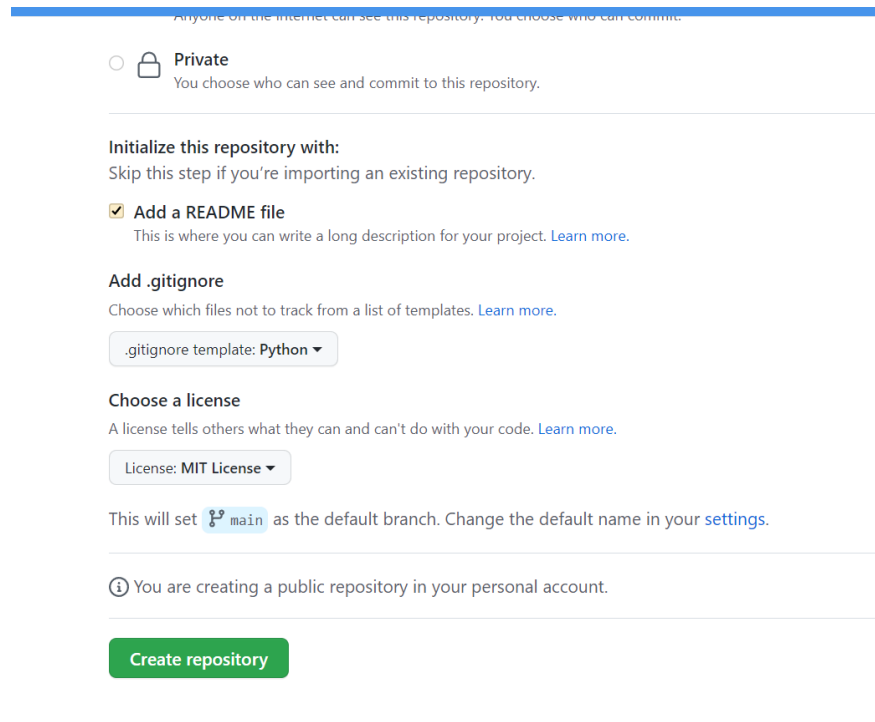


Рисунок 1. Создание репозитория

2. Выполните клонирование созданного репозитория.

```
C:\Users\User>cd C:\Users\User\Desktop\2 курс Python\lab 11
C:\Users\User\Desktop\2 курс Python\lab 11>git clone https://github.com/aikanyshkaukanbekova/lab11.git
Cloning into 'lab11'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 4.50 KiB | 383.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
C:\Users\User\Desktop\2 курс Python\lab 11>
```

Рисунок 2. Клонирование репозитория

3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.

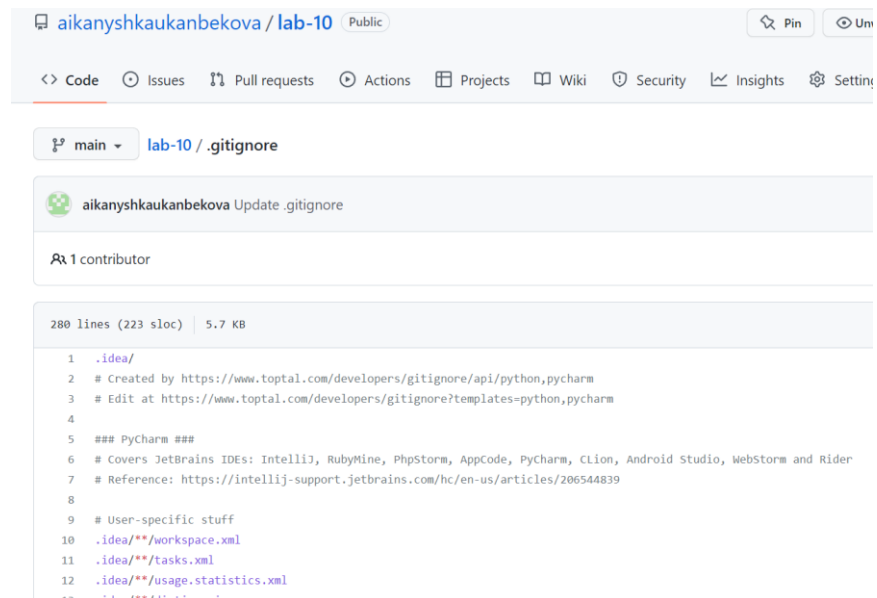


Рисунок 3. Дополнение файла .gitignore

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```

C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/User/Desktop/2 кypc Python/lab 10/lab-10/.git/hooks]
C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>

```

Рисунок 4. Организован модель ветвления git flow

5. Проработайте пример лабораторной работы. Создайте для него отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

```

#!/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_worker(staff):
    """
    Отобразить список работников
    """
    # Проверить что список работников не пуст
    if staff:
        # Заголовок таблицы
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках
        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
    """
    # Получить текущую дату
    today = date.today()
    # Сформировать список работников
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)

    # Возвратить список выбранных работников
    return result

```

```

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

    # Организовать бесконечный цикл запроса команд
    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_worker(workers)

        elif command.startswith('select '):
            parts = command.split(' ', maxsplit=1)
            period = int(parts[1])
            selected = select_workers(workers, period)
            display_worker(selected)

        elif command == 'help':
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            print("help - вывести список команд;")
            print("exit - завершить работу с программой.")

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

if __name__ == '__main__':
    main()

```

Рисунок 5. Пример лаб работы

6. Решите задачу: основная ветка программы, не считая заголовков функций, состоит из двух строки кода. Это вызов функции `test()` и инструкции `if __name__ == '__main__':`. В ней запрашивается на ввод целое число. Если оно положительное, то вызывается функция `positive()`, тело которой содержит команду вывода на экран слова "Положительное". Если число отрицательное, то вызывается функция `negative()`, ее тело содержит выражение вывода на экран слова "Отрицательное".

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

def te():
    number = int(input("Введите число: "))
    if number > 0:
        positive()
    else:
        negative()

def positive():
    print("Положительное")

def negative():
    print("Отрицательное")

if __name__ == '__main__':
    te()
```

Рисунок 6. Выполненное задание 1

7. Решите задачу: в основной ветке программы вызывается функция `cylinder()`, которая вычисляет площадь цилиндра. В теле `cylinder()` определена функция `circle()`, вычисляющая площадь круга по формуле . В теле `cylinder()` у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле , или полную площадь цилиндра. В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции `circle()`.

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

import math
import sys

def cylinder():
    """
    Вычисление площади полной или боковой поверхности цилиндра
    """
    while True:
        vs = int(input("Какую площадь вы хотите вычислить? (введите число):\n"))
        """
        1 -> Боковой поверхности\n
        2 -> Полной поверхности\n
        """
        if (vs != 1) and (vs != 2):
            print(f"Неизвестная команда {vs}", file=sys.stderr)
            break
```

```

r = int(input("Введите радиус "))
h = int(input("Введите высоту цилиндра: "))

if vs == 1:
    s = 2 * math.pi * r * h
    print("S(бок.) = ", '{:.3f}'.format(s))
    break

elif vs == 2:
    s = (2 * math.pi * r * h) + (circle(r) * 2)
    print("S(полн.) = ", '{:.3f}'.format(s))
    break

def circle(r):
    """
    Вычисление площади круга по заданному радиусу
    """
    return math.pi * (r ** 2)

if __name__ == '__main__':
    cylinder()

```

Рисунок 7. Выполненное задание 2

8. Решите следующую задачу: напишите функцию, которая считывает с клавиатуры числа и перемножает их до тех пор, пока не будет введен 0. Функция должна возвращать полученное произведение. Вызовите функцию и выведите на экран результат ее работы.

```

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

def factor():
    """
    Умножение введенных чисел, пока не будет введен 0
    """
    print("Вводите числа: ")
    res = 1
    while True:
        a = int(input(""))
        if a == 0:
            break
        else:
            res *= a
    print("Результат: ", res)

if __name__ == '__main__':
    factor()

```

Рисунок 8. Выполненное задание 3

9. Решите следующую задачу: напишите программу, в которой определены следующие четыре функции:

1. Функция `get_input()` не имеет параметров, запрашивает ввод с клавиатуры и возвращает в основную программу полученную строку.

2. Функция `test_input()` имеет один параметр. В теле она проверяет, можно ли переданное ей значение преобразовать к целому числу. Если можно, возвращает логическое `True`. Если нельзя – `False`.

3. Функция `str_to_int()` имеет один параметр. В теле преобразовывает переданное значение к целочисленному типу. Возвращает полученное число.

4. Функция `print_int()` имеет один параметр. Она выводит переданное значение на экран и ничего не возвращает.

```
#!/usr/bin/env python3
# __coding: utf-8 __

import sys

def get_input():
    """
    Запрашивает ввод с клавиатуры и
    возвращает в основную программу полученную строку.
    """
    get_str = input("Введите число: ")
    return get_str

def test_input(b):
    """
    Проверяет, можно ли переданное ей значение
    преобразовать к числу. Если да - возвращает True.
    """
    if type(b) == int or type(b) == float:
        return True
    elif b.isnumeric():
        return True
    else:
        return False

def str_to_int(b):
    """
    Преобразует переданное значение к
    целочисленному типу.
    """
    c = int(b)
    return c

def print_int(c):
    """
    Выводит переданное значение на экран
    """
```



```

        """
    print(c)

if __name__ == '__main__':
    a = get_input()
    bol = test_input(a)
    if bol:
        ch = str_to_int(a)
        print_int(ch)
    else:
        print(f"Введённое значение не является числом!", file=sys.stderr)

```

Рисунок 9. Выполненное задание 4

Индивидуальное задание

```

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

import sys

def get_route():
    """
    Запросить данные о маршруте
    """
    start = input("Название начального пункта маршрута? ")
    finish = input("Название конечного пункта маршрута? ")
    number = int(input("Номер маршрута? "))

    return {
        'start': start,
        'finish': finish,
        'number': number,
    }

def display_route(routes):
    """
    Отобразить список маршрутов
    """
    if routes:
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^15} |'.format(
                "№",
                "Начальный пункт",
                "Конечный пункт",
                "Номер маршрута"
            )
        )
        print(line)

    for idx, worker in enumerate(routes, 1):

```

```

        print(
            '| {:>4} | {:<30} | {:<20} | {:>15} |'.format(
                idx,
                worker.get('start', ''),
                worker.get('finish', ''),
                worker.get('number', 0)
            )
        )
    print(line)
else:
    print("Список маршрутов пуст.")

def select_route(routes, period):
    """
    Выбрать маршрут
    """
    result = []
    for employee in routes:
        if employee.get('number') == period:
            result.append(employee)

    return result

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

    while True:
        command = input(">>> ").lower()
        if command == 'exit':
            break

        elif command == 'add':
            route = get_route()
            routes.append(route)
            if len(routes) > 1:
                routes.sort(key=lambda item: item.get('number', ''))

        elif command == 'list':
            display_route(routes)

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

            selected = select_route(routes, period)
            display_route(selected)

        elif command == 'help':
            print("Список команд:\n")
            print("add - добавить маршрут;")
            print("list - вывести список маршрутов;")
            print("select <номер маршрута> - запросить данные о маршруте;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)

```

```
if __name__ == '__main__':  
    main()
```

Рисунки 9. Выполненное индивидуальное задание

8. Сделала коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```
C:\Users\User\Desktop\2 курс Python\lab 11\lab11>git add .  
C:\Users\User\Desktop\2 курс Python\lab 11\lab11>git commit -m "new"  
[main 682a77d] new  
6 files changed, 371 insertions(+)  
create mode 100644 indiv.py  
create mode 100644 primer.py  
create mode 100644 zadanie 1.py  
create mode 100644 zadanie 2.py  
create mode 100644 zadanie 3.py  
create mode 100644 zadanie 4.py  
C:\Users\User\Desktop\2 курс Python\lab 11\lab11>git push  
Enumerating objects: 9, done.  
Counting objects: 100% (9/9), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (8/8), done.  
Writing objects: 100% (8/8), 4.36 KiB | 893.00 KiB/s, done.  
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/aikanyshkaukanbekova/lab11.git  
e0ccd12..682a77d main -> main  
C:\Users\User\Desktop\2 курс Python\lab 11\lab11>
```

Рисунок 10. Сохранения

Контрольные вопросы:

1. Каково назначение функций в языке программирования Python?

Главной задачей функций в Python, как и в других языках программирования, является сокращение объёма кода и его структуризация. В функции, как правило, выносятся те части кода, которые выполняются в программе многократно.

2. Каково назначение операторов def и return?

Оператор def необходим для определения функции. После него идёт название самой функции, передаваемые в функцию параметры и само тело функции. Оператор return служит для возвращения результата выполнения функции в основную программу, где эта функция была вызвана.

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

Локальные переменные существуют только внутри функции. В другой части программы как-либо вызывать или изменить их невозможно.

Глобальные напротив – существуют во всей программе.

4. Как вернуть несколько значений из функции Python?

После оператора `return` необходимо записать все возвращаемые переменные через запятую, а при вызове функции нужно задать необходимое количество переменных. Куда будут возвращены параметры.

5. Какие существуют способы передачи значений в функцию?

По ссылке и по значению.

6. Как задать значение аргументов функции по умолчанию?

Нужно в скобках передаваемых параметров присвоить им значение.

7. Каково назначение `lambda`-выражений в языке Python?

`Lambda`-выражения – это небольшие функции, которые вызываются в программе один раз.

8. Как осуществляется документирование кода согласно PEP257?

Если пояснение функции содержит одну строку, то достаточно двух кавычек с каждой стороны строки. Пример: `"""Пояснение"""`. Если это многострочное пояснение, то необходимо три кавычки с каждой стороны. Пояснение находится в теле функции, сразу после её объявления.