

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Институт цифрового развития
Кафедра инфокоммуникаций**

**ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2.8
дисциплины
«Программирование на Python»**

Выполнила студентка группы
ИВТ-б-о-21-1
Яковлева Е.А. « » _____ 20__ г.
Подпись студента _____
Работа защищена
« » _____ 20__ г.

Проверил доцент
Кафедры инфокоммуникаций,
старший преподаватель
Воронкин Р.А.

(подпись)

Ставрополь, 2022 г.

Тема: Работа с функциями в языке Python

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

Порядок выполнения работы:

Создала общедоступный репозиторий на GitHub, в котором использованы лицензия MIT и язык программирования Python.

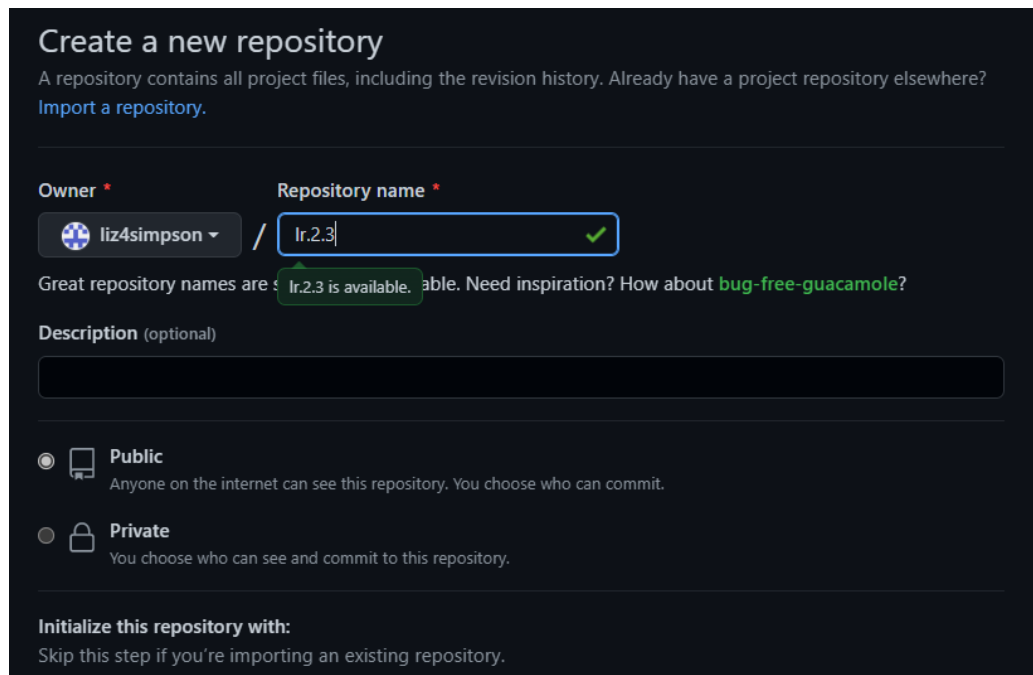
The image shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation of what a repository is. Below this, there are two main input fields: 'Owner' and 'Repository name'. The 'Owner' field is set to 'liz4simpson' with a dropdown arrow. The 'Repository name' field contains 'lr.2.3' and has a green checkmark next to it. Below these fields, there is a message: 'Great repository names are: lr.2.3 is available. able. Need inspiration? How about bug-free-guacamole?'. There is also a 'Description (optional)' text area. At the bottom, there are two radio buttons for 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is described as 'You choose who can see and commit to this repository.' At the very bottom, there is a section 'Initialize this repository with:' and a note 'Skip this step if you're importing an existing repository.'

Рисунок 1. Создание репозитория

Выполнила клонирование созданного репозитория.

```
C:\Users\Elizaveta>cd /d C:\Users\Elizaveta\Desktop\git
C:\Users\Elizaveta\Desktop\git>git clone https://github.com/liz4simpson/lr2.8.git
Cloning into 'lr2.8'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 13 (delta 3), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (13/13), 5.78 KiB | 394.00 KiB/s, done.
Resolving deltas: 100% (3/3), done.
C:\Users\Elizaveta\Desktop\git>
```

Рисунок 2. Клонирование репозитория

Организовала свой репозиторий в соответствие с моделью ветвления git-flow.

```

C:\Users\Elizaveta\Desktop\git\lr2.3>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/Elizaveta/Desktop/git/lr2.3/.git/hooks]

```

Рисунок 3. Организация репозитория согласно модели ветвления get-flow

2. Создала проект PyCharm в папке репозитория, проработала примеры из лабораторной работы.

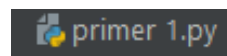


Рисунок 4. Проработанные примеры

```

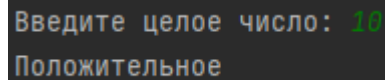
>>> add
Фамилия и инициалы: Коваленко А.З
Должность: младший слесарь
Год поступления: 2011
>>> list
+-----+-----+-----+-----+
| № |           Ф.И.О.           |      Должность      |   Год   |
+-----+-----+-----+-----+
|  1 | Коваленко А.З              | младший слесарь     |  2011   |
+-----+-----+-----+-----+
>>> select 1
+-----+-----+-----+-----+
| № |           Ф.И.О.           |      Должность      |   Год   |
+-----+-----+-----+-----+
|  1 | Коваленко А.З              | младший слесарь     |  2011   |
+-----+-----+-----+-----+

```

Рисунок 5. Результат выполнения программы

3. Задание №1. Решить следующую 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". Понятно, что вызов `test()` должен следовать после определения функций. Однако имеет ли значение порядок

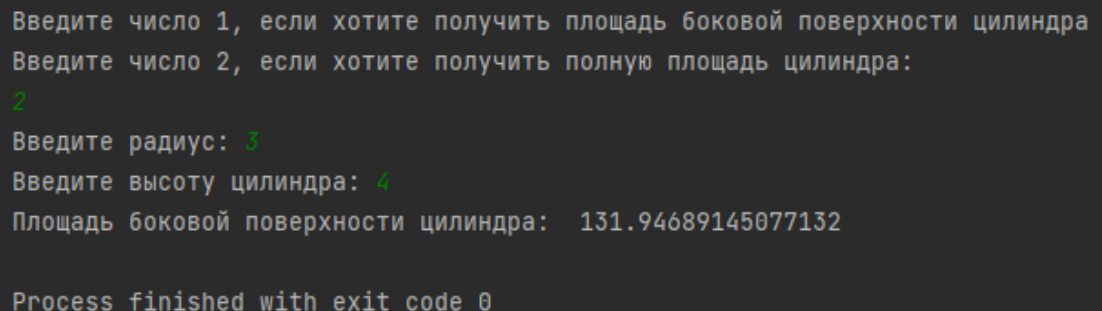
определения самих функций? То есть должны ли определения `positive()` и `negative()` предшествовать `test()` или могут следовать после него? Проверьте вашу гипотезу, поменяв объявления функций местами. Попробуйте объяснить результат.



```
Введите целое число: 10
Положительное
```

Рисунок 6. Результат выполнения программы

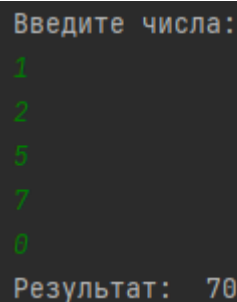
4. Задание №2. Решите следующую задачу: в основной ветке программы вызывается функция `cylinder()`, которая вычисляет площадь цилиндра. В теле `cylinder()` определена функция `circle()`, вычисляющая площадь круга по формуле $S = \pi r^2$. В теле `cylinder()` у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле $S_{\text{бок}} = 2\pi r h$, или полную площадь цилиндра. В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции `circle()`.



```
Введите число 1, если хотите получить площадь боковой поверхности цилиндра
Введите число 2, если хотите получить полную площадь цилиндра:
2
Введите радиус: 3
Введите высоту цилиндра: 4
Площадь боковой поверхности цилиндра: 131.94689145077132
Process finished with exit code 0
```

Рисунок 7. Результат выполнения программы

5. Задание №3. 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.



```
Введите числа:
1
2
5
7
0
Результат: 70
```

Рисунок 8. Результат выполнения программы

6. Задание №4. Решите следующую задачу: напишите программу, в которой определены следующие четыре функции:

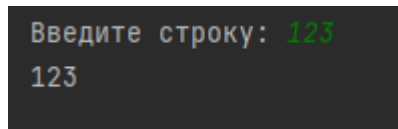
1. Функция `get_input()` не имеет параметров, запрашивает ввод с клавиатуры и возвращает в основную программу полученную строку.

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

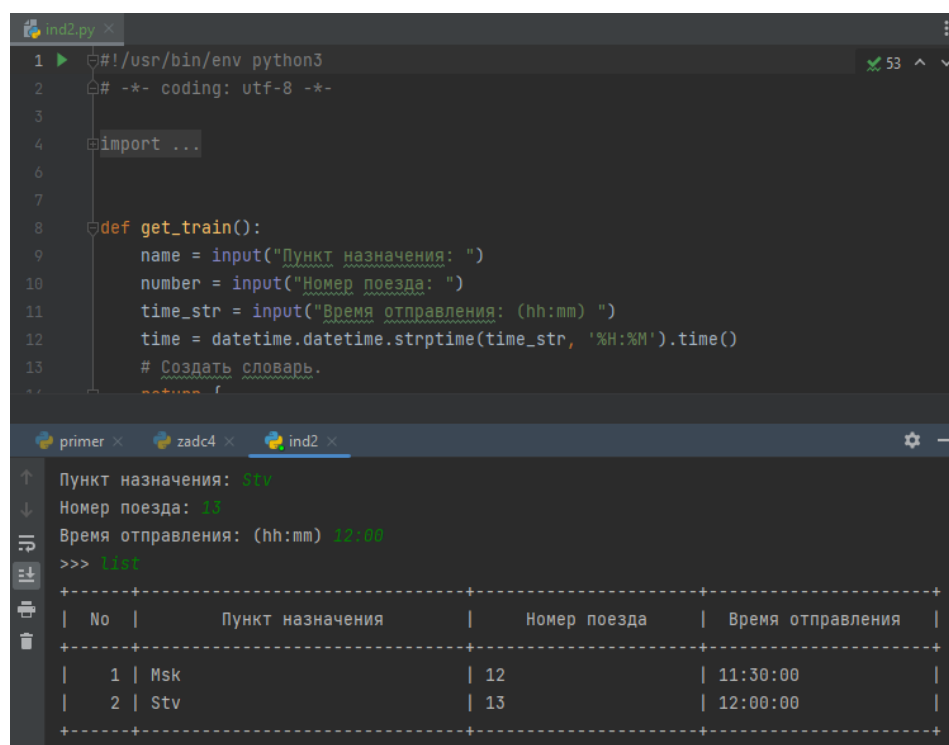


```
Введите строку: 123
123
```

Рисунок 9. Результат выполнения программы

Индивидуальное задание

7. Решить индивидуальное задание лабораторной работы 2.6, оформив каждую команду в виде отдельной функции.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import ...
5
6
7
8 def get_train():
9     name = input("Пункт назначения: ")
10    number = input("Номер поезда: ")
11    time_str = input("Время отправления: (hh:mm) ")
12    time = datetime.datetime.strptime(time_str, '%H:%M').time()
13    # Создать словарь.
14    return f
```

primer x zadc4 x ind2 x

Пункт назначения: Stv
Номер поезда: 13
Время отправления: (hh:mm) 12:00
>>> list

No	Пункт назначения	Номер поезда	Время отправления
1	Msk	12	11:30:00
2	Stv	13	12:00:00

Рисунок 10. Результат выполнения индивидуального задания

4. Сделала коммит, выполнила слияние с веткой main, и запустила изменения в удаленный репозиторий.

```
C:\Users\Elizaveta\Desktop\git\lr2.8>git add .
C:\Users\Elizaveta\Desktop\git\lr2.8>git commit -m "added"
[develop b1e16e2] added
7 files changed, 51 deletions(-)
delete mode 100644 .idea/.gitignore
delete mode 100644 .idea/inspectionProfiles/Project_Default.xml
delete mode 100644 .idea/inspectionProfiles/profiles_settings.xml
delete mode 100644 .idea/lr2.8.iml
delete mode 100644 .idea/misc.xml
delete mode 100644 .idea/modules.xml
delete mode 100644 .idea/vcs.xml
```

Рисунок 11. Фиксация и коммит файлов

```
C:\Users\Elizaveta\Desktop\git\lr2.8>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Elizaveta\Desktop\git\lr2.8>git merge develop
Updating 10e7331..b1e16e2
Fast-forward
 project/ind2.py   | 101 ++++++
 project/primer.py | 123 ++++++
 project/zadc1.py  | 24 ++++++
 project/zadc2.py  | 25 ++++++
 project/zadc3.py  | 18 ++++++
 project/zadc4.py  | 38 ++++++
 6 files changed, 329 insertions(+)
 create mode 100644 project/ind2.py
 create mode 100644 project/primer.py
 create mode 100644 project/zadc1.py
 create mode 100644 project/zadc2.py
 create mode 100644 project/zadc3.py
 create mode 100644 project/zadc4.py
```

Рисунок 12. Слияние ветки develop с main

```
C:\Users\Elizaveta\Desktop\git\lr2.8>git push
Enumerating objects: 21, done.
Counting objects: 100% (21/21), done.
Delta compression using up to 4 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (20/20), 5.81 KiB | 540.00 KiB/s, done.
Total 20 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/liz4simpson/lr2.8.git
 10e7331..b1e16e2  main -> main
```

Рисунок 13. Отправка изменений на удаленный репозиторий

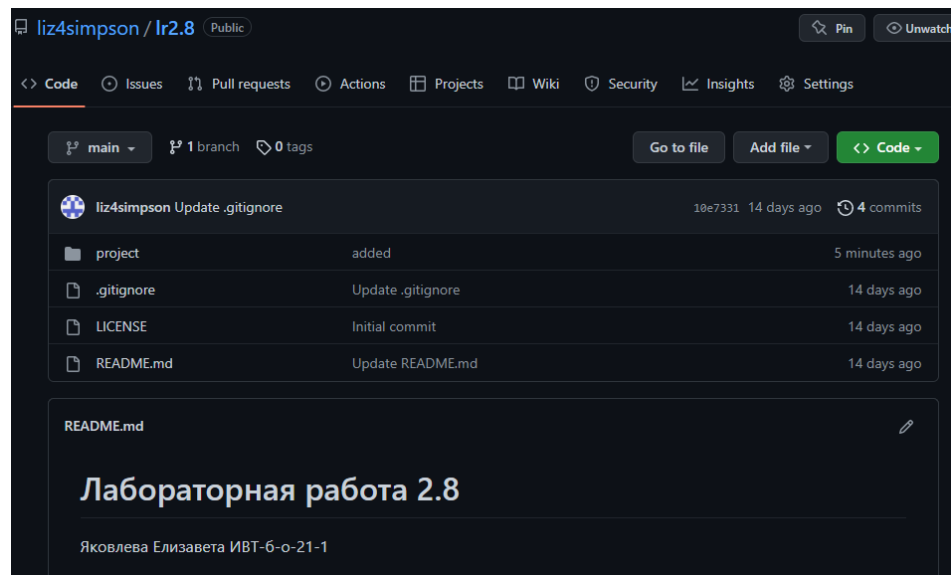


Рисунок 14. Изменения на удаленном репозитории

Вывод: в результате выполнения лабораторной работы были приобретены навыки для работы с функциями при написании программ с помощью языка программирования Python версии 3.x

Ответы на 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.

