

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

Институт цифрового развития  
Кафедра инфокоммуникаций

**«Исследование возможностей Git для работы с локальными  
репозиториями»**

**ОТЧЕТ  
по лабораторной работе №2\_1  
дисциплины  
«Основы программной инженерии»**

Выполнил:

Разворотников Денис Сергеевич  
2 курс, группа ПИЖ-б-о-21-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

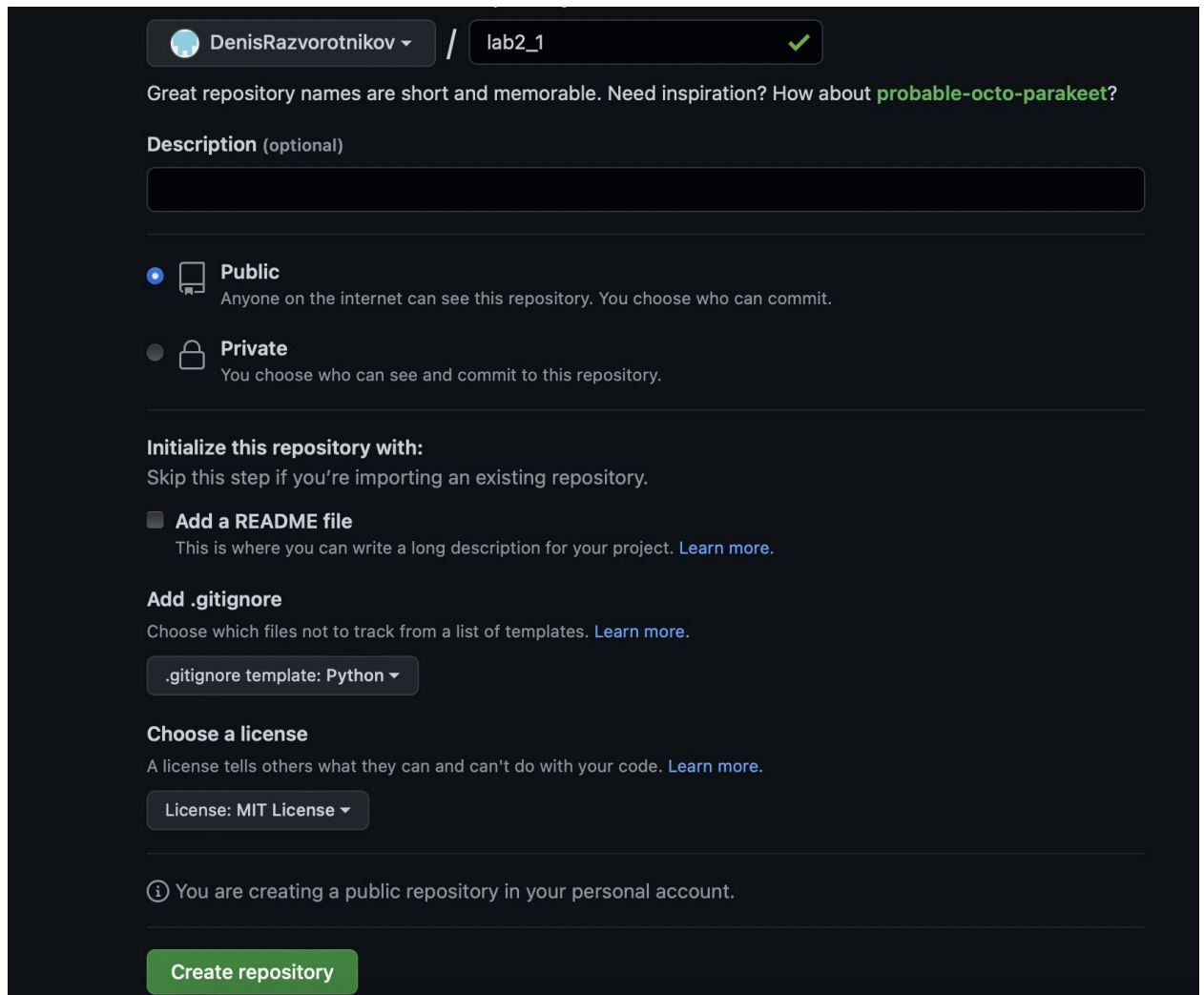
Проверил:

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

**Цель работы:** исследование процесса установки и базовых возможностей языка Python версии 3.x.



DenisRazvorotnikov / lab2\_1 ✓

Great repository names are short and memorable. Need inspiration? How about **probable-octo-parakeet**?

Description (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▾

ⓘ You are creating a public repository in your personal account.

**Create repository**

Рисунок 1 – был создан общедоступный репозиторий на Github с лицензией MIT и языком программирования Python

```
denisrazvorotnikov@Air-Denis ~ % cd desktop
denisrazvorotnikov@Air-Denis desktop % git clone https://github.com/DenisRazvorotnikov/lab2_1.git
Cloning into 'lab2_1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
denisrazvorotnikov@Air-Denis desktop %
```

Рисунок 2 – Клонирование репозитория

```
[denisrazvorotnikov@Air-Denis lab2_1 % git branch ForWork
[denisrazvorotnikov@Air-Denis lab2_1 % git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
[denisrazvorotnikov@Air-Denis lab2_1 % git checkout ForWork
Switched to branch 'ForWork'
denisrazvorotnikov@Air-Denis lab2_1 %
```

Рисунок 3 – Организация репозитория в соответствии с моделью ветвления git-flow

1. Напишите программу (файл user.py), которая запрашивала бы у пользователя:

его имя (например, "What is your name?")

возраст ("How old are you?")

место жительства ("Where are you live?")

После этого выводила бы три строки:

"This is `имя`"

"It is `возраст`"

"(S)he lives in `место\_жительства`"

Вместо имя, возраст, местожительства должны быть данные, введенные пользователем.

Примечание: можно писать фразы на русском языке, но, если вы планируете стать профессиональным программистом, привыкайте к английскому.

Код программы

```
a=input ("What is your name? ")
b=input("How old are you? ")
c=input("Where are you live? ")
print ("This is ", a)
print ("It is ", b)
print ("(S)he live in ",c)
```

```
What is your name? Den
How old are you? 21
Where are you live? Stav
This is Den
It is 21
(S)he live in Stav
```

Рисунок 4 – Результат работы программы

2. Напишите программу (файл arithmetic.py), которая предлагала бы пользователю решить пример  $4 * 100 - 54$ . Потом выводила бы на экран правильный ответ и ответ пользователя. Подумайте, нужно ли здесь преобразовывать строку в число.

Код программы:

```
sol = input("what is the answer?\n4*100-54 = ")
corsol = 346
print ("the correct solution is: ",corsol)
print ("your answer is: ",sol)
```

```
4*100-54 = 346
the correct solution is: 346
your answer is: 346

Process finished with exit code 0
```

Рисунок 5 – Результат работы программы

3. Запросите у пользователя четыре числа (файл numbers.py). Отдельно сложите первые два и отдельно вторые два. Разделите первую сумму на вторую. Выведите результат на экран так, чтобы ответ содержал две цифры после запятой.

Код программы:

```

print("enter four digits: ")
dg1 = int(input())
dg2 = int(input())
dg3 = int(input())
dg4 = int(input())

sum1 = dg1 + dg2
sum2 = dg3 + dg4
div = sum1 / sum2

print("we have made some calculation\nthe result is: ",format(div,".3"))

```

```

enter four digits:
3
5
2
5
we have made some calculation
the result is:  1.14

```

Рисунок 6 – Результат работы программы

4. Напишите программу (файл individual.py) для решения индивидуального задания. Вариант индивидуального задания уточните у преподавателя.

Даны длины сторон прямоугольного параллелепипеда. Найти его объем и площадь боковой поверхности.

Код программы:

```

from math import sqrt
a = float(input("Длина большего основания:"))
b = float(input("Длина меньшего основания:"))
h = float(input("Высота:"))
print("P =", a + b + 2 * sqrt(sqrt(h) + sqrt(a - b) / 4))

```

```

Длина большего основания:3
Длина меньшего основания:2
Высота:5
P = 8.153453965099088

```

Рисунок 7 – Результат выполнения программы

```
denisrazvorotnikov@Air-Denis lab2_1 % git add .
denisrazvorotnikov@Air-Denis lab2_1 % git commit -m "Add of folders"
[ForWork bad5e35] Add of folders
16 files changed, 108 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/lab2_1.iml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 PY/.idea/.gitignore
create mode 100644 PY/.idea/PY.iml
create mode 100644 PY/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 PY/.idea/misc.xml
create mode 100644 PY/.idea/modules.xml
create mode 100644 PY/.idea/vcs.xml
create mode 100644 PY/arithmetric.py
create mode 100644 PY/individual.py
create mode 100644 PY/numbers.py
create mode 100644 PY/user.py
```

Рисунок 8 – Коммит изменений ветки ForWork

```
denisrazvorotnikov@Air-Denis lab2_1 % git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
denisrazvorotnikov@Air-Denis lab2_1 % git merge ForWork
Updating a0926d7..bad5e35
Fast-forward
 .idea/.gitignore                | 3 +++
 .idea/inspectionProfiles/profiles_settings.xml | 6 ++++++
 .idea/lab2_1.iml                | 10 ++++++++
 .idea/misc.xml                  | 7 ++++++
 .idea/modules.xml               | 8 ++++++
 .idea/vcs.xml                   | 6 ++++++
 PY/.idea/.gitignore             | 3 +++
 PY/.idea/PY.iml                 | 10 ++++++++
 PY/.idea/inspectionProfiles/profiles_settings.xml | 6 ++++++
 PY/.idea/misc.xml               | 7 ++++++
 PY/.idea/modules.xml            | 8 ++++++
 PY/.idea/vcs.xml                | 6 ++++++
 PY/arithmetric.py               | 5 +++++
 PY/individual.py                 | 6 ++++++
 PY/numbers.py                   | 11 ++++++++
 PY/user.py                      | 6 ++++++
16 files changed, 108 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/lab2_1.iml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 PY/.idea/.gitignore
create mode 100644 PY/.idea/PY.iml
create mode 100644 PY/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 PY/.idea/misc.xml
create mode 100644 PY/.idea/modules.xml
create mode 100644 PY/.idea/vcs.xml
create mode 100644 PY/arithmetric.py
create mode 100644 PY/individual.py
create mode 100644 PY/numbers.py
create mode 100644 PY/user.py
```

Рисунок 8 – Слияние ветки ForWork с веткой main

```
[denisrazvorotnikov@Air-Denis lab2_1 % git push git@github.com:DenisRazvorotnikov]
/lab2_1.git
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 8 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (19/19), 2.51 KiB | 1.25 MiB/s, done.
Total 19 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To github.com:DenisRazvorotnikov/lab2_1.git
   a0926d7..bad5e35  main -> main
denisrazvorotnikov@Air-Denis lab2_1 %
```

Рисунок 9 – Push на удаленный сервер

Вывод: в результате лабораторной работы исследования процесса установки и базовых возможностей языка Python, был получен опыт установки таких программ как Anaconda, PyCharm, Python, также были получены знания работы с PyCharm

## Контрольные вопросы

1. Опишите основные этапы установки Python в Windows и Linux.

Для установки интерпретатора Python первое, что нужно сделать – это скачать дистрибутив. Загрузить его можно с официального сайта, перейдя по ссылке <https://www.python.org/downloads/>.

2. Запустить скачанный установочный файл.
3. Выбрать способ установки.
4. Отметить необходимые опции установки
5. Выбирать место установки

При установке для Linux, в случае ошибки необходимо либо собрать Python из исходников, либо взять из репозитория. Для установки из репозитория в Ubuntu воспользуйтесь командой «`sudo apt-get install python3`»

2. В чем отличие пакета Anaconda от пакета Python, скачиваемого с официального сайта?

Пакет Anaconda содержит версии языка Python 2 и 3, набор наиболее часто используемых библиотек и удобную среду разработки

3. Как осуществить проверку работоспособности пакета Anaconda?

В Windows это можно сделать, выбрав следующий пункт главного меню системы Пуск Anaconda3 (64-bit) Anaconda Prompt. В появившейся командной строке необходимо ввести «`jupyter notebook`», в результате чего отобразиться процесс загрузки веб-среды Jupyter Notebook, после чего запустится веб-сервер и среда разработки в браузере.

Создайте ноутбук для разработки, для этого нажмите на кнопку New (в правом углу окна) и в появившемся списке выберете Python. В результате будет создана новая страница в браузере с ноутбуком. Введите в первой ячейке команду «`print("Hello, World!")`» и нажмите Alt+Enter на клавиатуре. Ниже ячейки должна появиться соответствующая надпись.



4. Как задать используемый интерпретатор языка Python в IDE PyCharm?

Указать путь до интерпретатора в настройках IDE, для этого:

- 1) Нажмите на шестеренку в верхнем правом углу, выберите "Add..".
- 2) Далее выберите "System Interpreter";
- 3) Нажмите на 3 точки "..." справа от поля в выбор интерпретатора;
- 4) Укажите путь до интерпретатора

5. Как осуществить запуск программы с помощью IDE PyCharm?  
Shift+F10

6. В чем суть интерактивного и пакетного режимов работы Python?  
В интерактивном.

Python можно использовать как калькулятор для различных вычислений, а если дополнительно подключить необходимые математические библиотеки, то по своим возможностям он становится практически равным таким пакетам как Matlab, Octave и т.п.

В проектном.

В этом режиме сначала записывается вся программа, а потом эта программа выполняется полностью.

7. Почему язык программирования Python называется языком динамической типизации?

В нем проверка типа происходит во время выполнения, а не компиляции

8. Какие существуют основные типы в языке программирования Python?

1. None
2. Логические переменные
3. Числа

4. Списки
5. Строки
6. Бинарные списки
7. Множества
8. Словари

9. Как создаются объекты в памяти? Каково их устройство? В чем заключается процесс

Для того, чтобы объявить и сразу инициализировать переменную необходимо написать её имя, потом поставить знак равенства и значение, с которым эта переменная будет создана.

При инициализации переменной, на уровне интерпретатора, создается целочисленный объект, который имеет некоторый идентификатор, значение и тип. Посредством оператора “=” создается ссылка между переменной и объектом.

10. Как получить список ключевых слов в Python?

Нужно подключить модуль `keyword` и воспользоваться командой `keyword.kwlist`.

11. Каково назначение функций `id()` и `type()`?

Функция `id()` предназначена для получения значения идентичности объекта. С помощью функции `type()` можно получить тип конкретного объекта.

12. Что такое изменяемые и неизменяемые типы в Python.

К неизменяемым (immutable) типам относятся: целые числа (`int`), числа с плавающей точкой (`float`), комплексные числа (`complex`), логические переменные (`bool`), кортежи (`tuple`), строки (`str`) и неизменяемые множества

(frozen set).

К изменяемым (mutable) типам относятся: списки (list), множества (set), словари (dict).

13. Чем отличаются операции деления и целочисленного деления?

При целочисленном делении отбрасывается дробная часть от деления чисел, при операции деления дробная часть не отбрасывается.

14. Какие имеются средства в языке Python для работы с комплексными числами?

Для создания комплексного числа можно использовать функцию `complex(a, b)`, в которую, в качестве первого аргумента, передается действительная часть, в качестве второго – мнимая. Либо записать число в виде  $a + bj$ . Комплексные числа можно складывать, вычитать, умножать, делить и возводить в степень. У комплексного числа можно извлечь действительную (`x.real`) и мнимую части (`x.imag`). Для получения комплексно сопряжённого числа необходимо использовать метод `conjugate()`.

15. Каково назначение и основные функции библиотеки (модуля) `math`? По аналогии с модулем `math` изучите самостоятельно назначение и основные функции модуля `cmath`.

`math.ceil(x)` - возвращает ближайшее целое число большее, чем  $x$ .

`math.fabs(x)` - возвращает абсолютное значение числа.

`math.factorial(x)` - вычисляет факториал  $x$ .

`math.floor(x)` - возвращает ближайшее целое число меньшее, чем  $x$ .

`math.exp(x)` - вычисляет  $e^{**}x$ .

`math.log2(x)` - логарифм по основанию 2.

`math.log10(x)` - логарифм по основанию 10.

`math.log(x[, base])` - по умолчанию вычисляет логарифм по основанию  $e$ , дополнительно можно указать основание логарифма.

`math.pow(x, y)` - вычисляет значение  $x$  в степени  $y$ .

`math.sqrt(x)` - корень квадратный от  $x$ .

`math.cos(x)` - косинус от  $x$ .

`math.sin(x)` - синус от  $x$ .

`math.tan(x)` - тангенс от  $x$ .

`math.acos(x)` - арккосинус от  $x$ .

`math.asin(x)` - арксинус от  $x$ .

`math.atan(x)` - арктангенс от  $x$ .

`math.pi` - число  $\pi$ .

`math.e` - число  $e$ .

16. Каково назначение именных параметров `sep` и `end` в функции `print()`?

`sep()` устанавливает отличный от пробела разделитель строк.

`End()` указывает, что делать, после вывода строки (по умолчанию стоит переход на новую строку)

17. Каково назначение метода `format()`? Какие еще существуют средства для форматирования строк в Python? Примечание: в дополнение к рассмотренным средствам изучите самостоятельно работу с f-строками в Python.

Форматирование может выполняться в так называемом старом стиле или с помощью строкового метода `format`. Символы `%s`, `%d`, `%f` подставляются значения переменных. Буквы `s`, `d`, `f` обозначают типы данных – строку, целое число, вещественное число.

18. Каким образом осуществить ввод с консоли значения целочисленной и вещественной переменной в языке Python?

Указать перед `input` тип данных: `int(input())`.