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

Департамент цифровых, робототехнических систем и электроники

«Основы языка Python»
Отчет по лабораторной работе № 2
по дисциплине «Программирование на Python»
Вариант 5

Выполнил студент группы ИВТ-б-о-24-1
Грабарь Артемий Павлович
«__» октября 2025г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2025

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

Ссылка на GitHub: https://github.com/Arhi258/Laba_2.git

Задание:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
2. Выполнить клонирование данного репозитория.
3. Организовать репозиторий в соответствии с моделью ветвления git-flow.
4. Создать проект PyCharm в папке репозитория.
5. Решить следующие задачи с помощью языка программирования Python3 и IDE PyCharm:
6. Написать программу, которая будет запрашивать у пользователя: его имя, возраст, место жительства и вывести эти три строки.
7. Написать программу, которая предлагала бы пользователю решить пример 4*100-54. Потом выводила бы на экран правильный ответ и ответ пользователя.
8. Запросить у пользователя 4 числа. Отдельно сложить первые два и отдельно вторые два. Разделить первую сумму на вторую. Вывести результат на экран так, чтобы ответ содержал две цифры после запятой.
9. Написать программу для решения индивидуального задания: “Даны длины сторон прямоугольного параллелепипеда. Найти его объем и площадь боковой поверхности”.
10. Написать программу для решения задания повышенной сложности: “ Даны целые числа h, m, s (0 <h ≤ 23, 0 <m ≤ 59, 0 <s ≤ 59), указывающие момент времени: «h часов, т минут, s секунд». Определить угол (в градусах) между положением часовой стрелки в начале суток и в указанный момент времени”.

11. Выполнить коммит всех файлов в репозиторий git в ветку для разработки.

12. Добавить отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксировать изменения.

13. Выполнить слияние ветки для разработки с веткой main/master.

14. Отправить сделанные изменения на сервер GitHub.

Ход работы:

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

2. Было выполнено клонирование данного репозитория (рис 1).

```
arhi2@DESKTOP-25N997A MINGW64 ~
$ cd Z:\Репозиторий

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий
$ git clone https://github.com/Arhi258/Laba_2.git
Cloning into 'Laba_2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (5/5), done.
```

Рисунок 1. Клонирование репозитория

3. Была создана ветка develop и feature (рис 2).

```
arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$ git status
On branch develop
nothing to commit, working tree clean

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$ git checkout -b feature
Switched to a new branch 'feature'

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (feature)
$ git status
On branch feature
nothing to commit, working tree clean

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (feature)
$
```

Рисунок 2. Создание веток

4. Была написана программа, которая будет запрашивать у пользователя: его имя, возраст, место жительства и вывести эти три строки. (рис 3). Программа была сохранена в ветку feature.

```
name = input("What is your name? ")
age = input("How old are you? ")
home = input("Where are you live? ")

print(f"This is {name}")
print(f"It is {age}")
print(f"(S)he live in {home}")
```

Рисунок 3. Программа для задания 6

5. Была написана программа, которая предлагала бы пользователю решить пример 4*100-54. Потом выводила бы на экран правильный ответ и ответ пользователя (рис 4). Программа была сохранена в ветку feature.

```
result = input("Solve the example: 4*100-54=")

print(f"Correct answer: {4*100-54}, your answer {result}")
```

Рисунок 4. Программа для задания 7

6. Была написана программа которая запрашивает у пользователя 4 числа. Отдельно складывает первые два и отдельно вторые два. Делит первую сумму на вторую. Выводит результат на экран так, чтобы ответ содержал две цифры после запятой (рис 5). Программа была сохранена в ветку feature.

```
number_1 = int(input("Введите первое число: "))
number_2 = int(input("Введите второе число: "))
number_3 = int(input("Введите третье число: "))
number_4 = int(input("Введите четвертое число: "))

sum_1_and_2 = number_1 + number_2
sum_3_and_4 = number_3 + number_4
result = sum_1_and_2 / sum_3_and_4

print(f"{result:.2f}")
```

Рисунок 5. Программа для задания 8

7. Была написана программа для решения индивидуального задания: “Даны длины сторон прямоугольного параллелепипеда. Найти его объем и

площадь боковой поверхности” (рис 6). Программа была сохранена в ветку feature.

```
side_length_a = int(input("Введите сторону a = "))
side_length_b = int(input("Введите сторону b = "))
side_length_c = int(input("Введите сторону c = "))

volume = side_length_a * side_length_b * side_length_c
lateral_surface = 2 * side_length_c * (side_length_a + side_length_b)

print(f"Объем прямоугольного параллелепипеда равен: {volume}")
print(f"Площадь боковой поверхности прямоугольного параллелепипеда равна: {lateral_surface}")
```

Рисунок 6. Программа для задания 9

8. Написать программу для решения задания повышенной сложности: “ Даны целые числа h, m, s (0 <h ≤ 23, 0 <m ≤ 59, 0 <s ≤ 59), указывающие момент времени: «h часов, т минут, s секунд». Определить угол (в градусах) между положением часовой стрелки в начале суток и в указанный момент времени” (рис 7). Программа была сохранена в ветку feature.

```
h = int(input("Введите часы: "))
m = int(input("Введите минуты: "))
s = int(input("Введите секунды: "))

angle_h = h * 30
angle_m = m * 0.51
angle_s = s * 0.0086
common_angle = angle_m + angle_s + angle_h

print(f"Угол между положением часовой стрелкой в начале суток и в указанный момент: {common_angle}")
```

Рисунок 7. Программа для задания 10

9. Был выполнен коммит всех файлов и слияние веток feature и develop (рис 8).

```
arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (feature)
$ git checkout develop
Switched to branch 'develop'

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$ git merge --no-ff feature
Merge made by the 'ort' strategy.
 .idea/.gitignore  |  3 +++
 .idea/vcs.xml     |  4 ++++
 arithmetic.py     |  3 +++
 difficult_task.py | 10 ++++++++++
 individual.py     |  9 +++++++++
 numbers.py        | 10 ++++++++++
 user.py           |  7 +++++++
 7 files changed, 46 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/vcs.xml
 create mode 100644 arithmetic.py
 create mode 100644 difficult_task.py
 create mode 100644 individual.py
 create mode 100644 numbers.py
 create mode 100644 user.py

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$ git branch -d feature
Deleted branch feature (was 8a61973).

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$
```

Рисунок 8. Слияние веток

10. Было выполнено слияние ветки разработки с веткой main (рис 9).

```
arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

arhi2@DESKTOP-25N997A MINGW64 /z/Репозиторий/Laba_2 (main)
$ git merge --no-ff develop
Merge made by the 'ort' strategy.
 .idea/.gitignore  |  3 +++
 .idea/vcs.xml     |  4 ++++
 arithmetic.py     |  3 +++
 difficult_task.py | 10 ++++++++++
 individual.py     |  9 +++++++++
 numbers.py        | 10 ++++++++++
 user.py           |  7 +++++++
 7 files changed, 46 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/vcs.xml
 create mode 100644 arithmetic.py
 create mode 100644 difficult_task.py
 create mode 100644 individual.py
 create mode 100644 numbers.py
 create mode 100644 user.py
```

Рисунок 9. Слияние веток

11. Был добавлен отчет по лабораторной работе в формате PDF в папку doc репозитория. Изменения были зафиксированы.

Контрольные вопросы:

1. Основные этапы установки Python на Windows: скачать установщик, запустить его, выбрать способ установки, завершить установку (для проверки установки можно открыть командную строку и ввести python –version). Этапы установки на Linux: чаще всего Python установлен по умолчанию (проверка с помощью команды python3 --version), если он не установлен, нужно для каждой версии Linux ввести определенную команду (для Debian: sudo apt install python3, для Febora: sudo dnf install python3, для Arch Linux: sudo pacman –s python), с помощью команды python3 –version проверить установку.

2. Отличия между Anaconda и стандартным дистрибутивом Python с официального сайта заключаются в основном в составе, назначении и удобстве для определенных задач. Стандартный Python (с официального сайта): минималистичный дистрибутив (содержит только интерпретатор Python и стандартную библиотеку), установка базовых инструментов, гибкость и легкость. Anaconda: полный дистрибутив (ориентирован на научные вычисления, анализ данных и машинное обучение), включает Python и более 150 заранее установленных пакетов, включает собственный менеджер пакетов и окружений.

3. Чтобы проверить работоспособность пакета Anaconda нужно выполнить следующие шаги: открыть терминал (Linux/Mac) или Anaconda prompt (Windows), ввести команду проверки версии conda --version и python --version, запустить интерпретатор Python и проверить его работу с помощью любой команды, проверить работу менеджера пакетов conda (обновить список пакетов командой: conda list, выведутся установленные пакеты).

4. Способ задания используемого интерпретатора в IDE PyCharm: перейти в меню File, settings на этом экране будет текущий интерпретатор, можно выбрать другой интерпретатор. Чтобы добавить интерпретатор нужно открыть окно Add interpreter.

5. Варианты запуска программы с помощью IDE PyCharm: Нажать по нужному файлу правой кнопкой мыши, запустить. Открыть нужный файл в редакторе, нажать зеленую кнопку Play.

6. В интерактивном режиме код вводится и выводится строка за строкой. В пакетном режиме Python читает, интерпретирует и выполняет весь файл целиком.

7. Язык программирования Python называется языком с динамической типизацией потому, что типы данных переменных в нем определяются и проверяются во время выполнения программы, а не заранее в процессе компиляции.

8. Основные типы данных в Python: числовые, логические, последовательности, изменяемые и неизменяемые множества, ассоциативные коллекции, специальные типы.

9. Любые данные в Python представляют собой объекты в памяти, они создаются при явном объявлении литералов, вызове функций и конструкторов, вычислениях. Каждый объект в Python имеет: счетчик ссылок, указатель на структуру типа, данные объекта. В Python переменная – это имя указывающее на объект (переменные не хранят значение, а ссылку на объект). Присваивание не копирует объект, а только создает новую ссылку.

10. В Python список ключевых слов можно получить с помощью стандартного модуля keyword. Сначала импортируем keyword, затем пишем keywords_list = keyword.kwlist, и выводим полученную переменную print(keywords_list).

11. Функция id() возвращает уникальный идентификатор объекта, который соответствует его месту в памяти во время выполнения программы. Функция type() возвращает тип переданного объекта.

12. Объект изменяемого типа можно изменять без создания нового файла. Объект неизменяемого типа после создания нельзя изменить, любая операция которая меняет значение, создает новый объект.

13. Операция деления выполняет деление с плавающей точкой, операция целочисленного деления выполняет деление с отбрасыванием остатка.

14. Средства для работы с комплексными числами: литералы комплексных чисел, функция complex (), атрибуты (real, imag, conjugate()), функция abs(), математические функции из модуля cmath.

15. Модуль math в Python – это стандартная библиотека для выполнения математических вычислений. Он работает с числами (int, float) и предоставляет набор функций, которые расширяют возможности обычных арифметических операций. Основные функции: константы, арифметика и округления, степени и корни, логарифмы, тригонометрия, работа с углами, специальные функции. Модуль cmath очень похож на модуль math, но он всегда возвращает комплексное число.

16. Параметры sep и end в функции print() управляют форматированием вывода. Параметр sep определяет строку, которая будет вставлена между выводимыми значениями. Параметр end определят строку, которая будет добавлена в конце вывода.

17. Метод format() позволяет подставлять значения в строку по шаблону. Другие способы форматирования: f-строки, %-форматирование, конкантенация строк, метод string.Template.

18. Ввод с консоли выполняется с помощью команды input(), чтобы получить целочисленное или вещественное значение нужно обернуть команду input () в int() или float().

Вывод: В результате работы были исследованы процесс установки и базовые возможности языка Python версии 3.x.