Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии

Департамент цифровых, робототехнических систем и электроники

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2**

**дисциплины «Программирование на Python»**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнила:  Ковжого Елизавета Андреевна  2 курс, группа ИВТ-б-о-24-1,  09.03.01 «Информатика и вычислительная техника», направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Руководитель практики:  Воронкин Р. А., доцент департамента цифровых, робототехнических систем и электроники института перспективной инженерии.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  |  | |  |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2025 г.

Тема: оcновы языка Python.

Цель: исследовать процесс установки и базовые возможности языка Python версии 3.х.

Порядок выполнения работы:

Адрес созданного репозитория: https://github.com/LissKovzogo/Python\_LAB\_2.git

1. Последовательно установили на устройство Python, Anaconda и PyCharm, после чего проверили работоспособность.

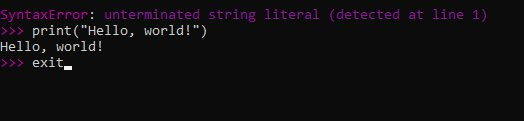


Рис.1 — Проверка работоспособности Python

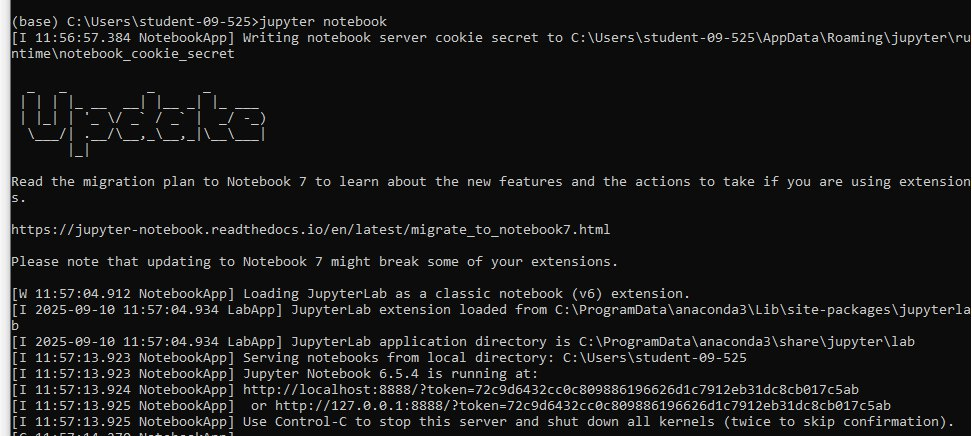


Рис. 2 — Проверка работоспособности Anaconda

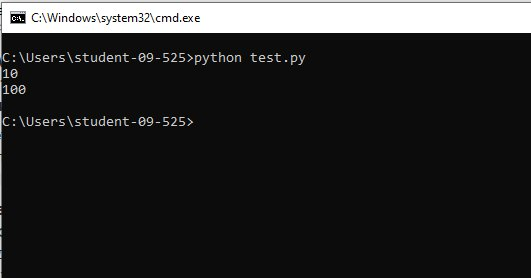


Рис. 3 —Проверка работоспособности PyCharm

2. Создали новый репозиторий и клонировали их.

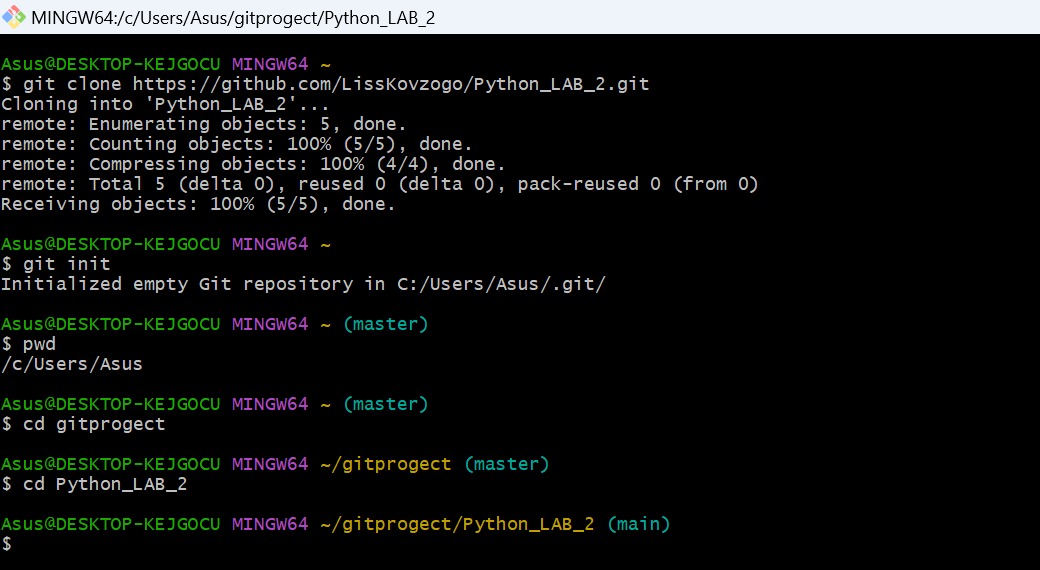


Рис. 4 — Клонирование репозитория

3. В соответствии с заданиями написали код и сохранили в файлы user.py, arithmetic.py,numbers.py,individul.py,individual2.py.

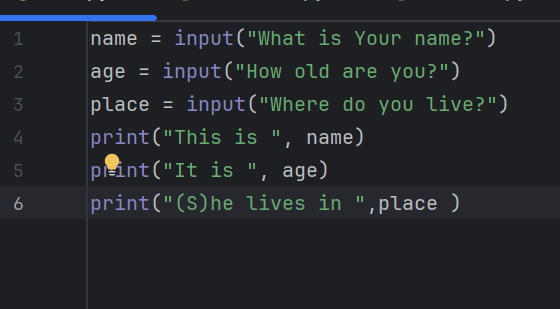


Рис. 5 — user.py

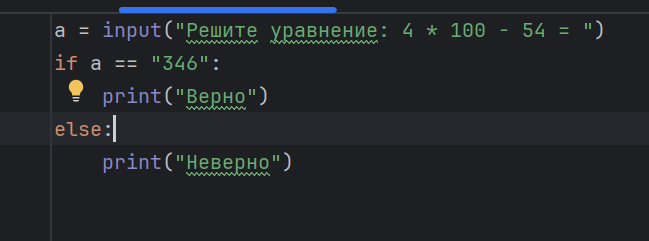


Рис. 6 — arithmetic.py

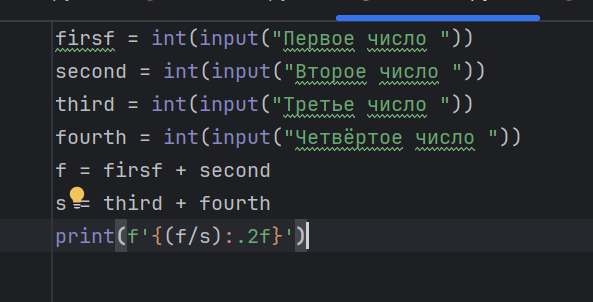


Рис. 7 — numbers.py

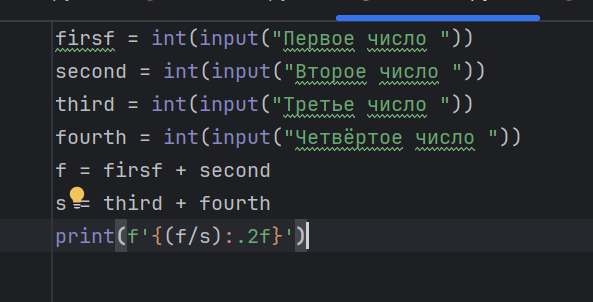


Рис. 8 — individul.py

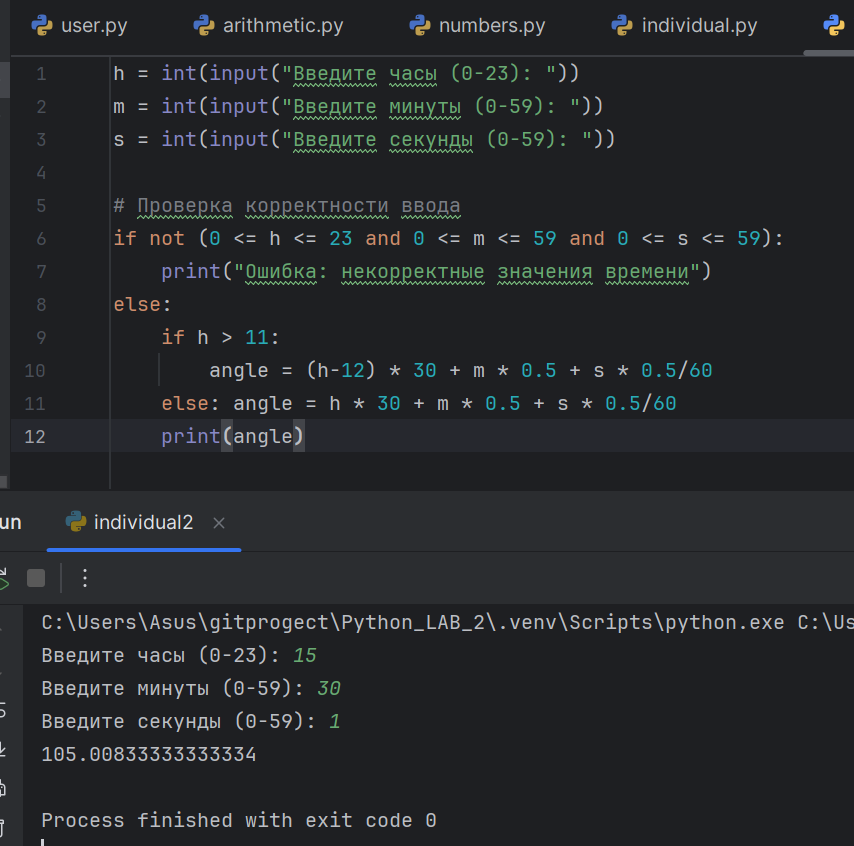


Рис. 9 — individual2.py

4. Произвели коммит всех созданных файлов и выполнили слияние веток.

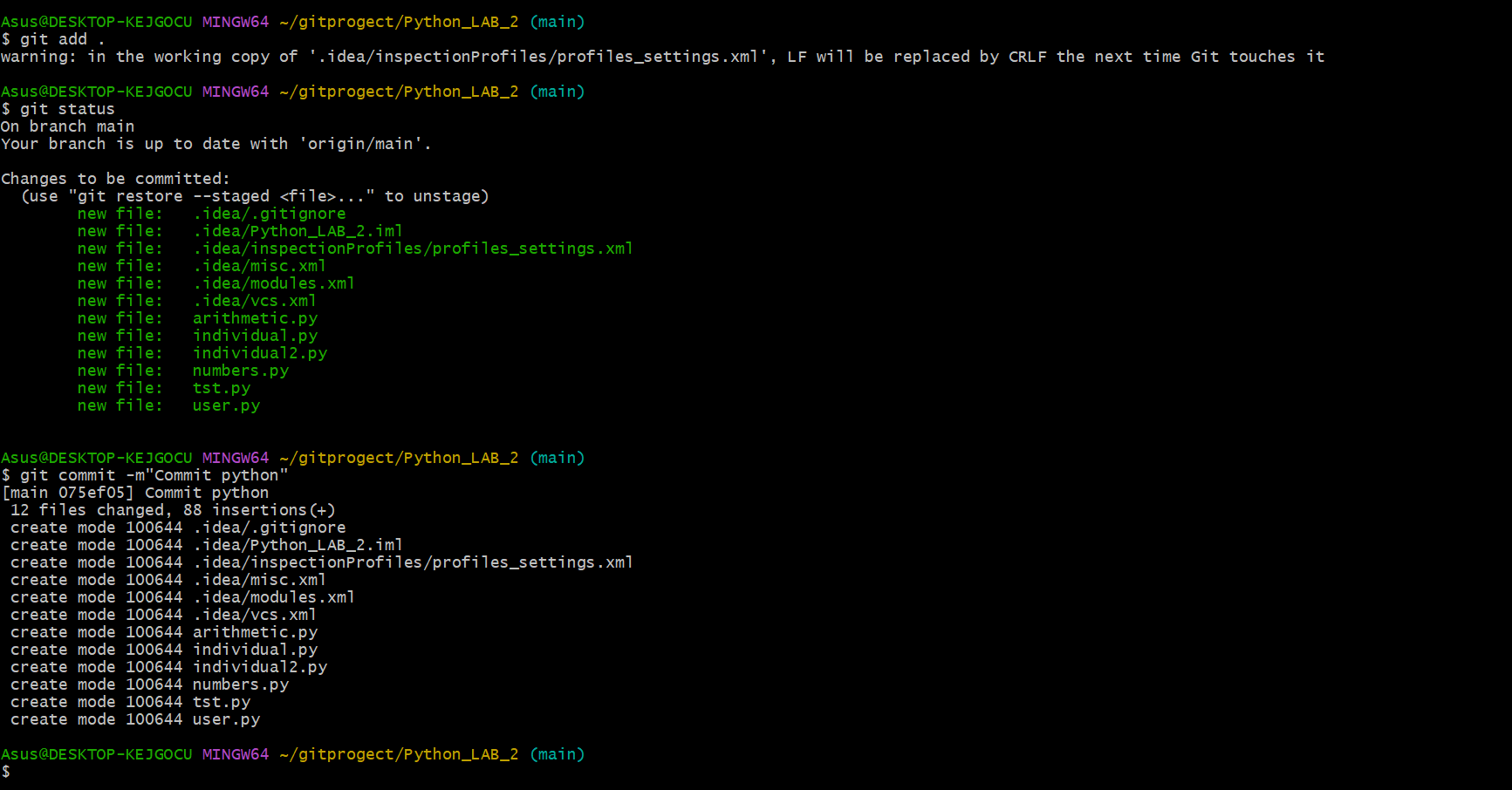


Рис. 10 — Сохранение обновлений репозитория

5. Отправили изменения в удалённый репозиторий.

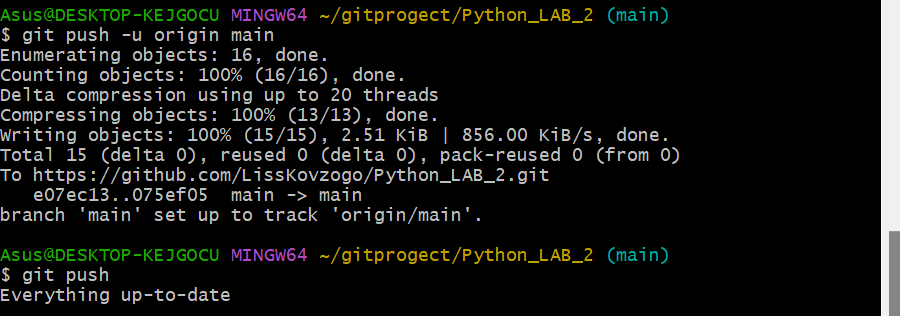


Рис. 11 — Отправление на сервер GitHub

Контрольные вопросы:

1. Основные этапы установки Python

Windows:

* Скачать установщик с python.org
* Запустить установщик, отметить "Add Python to PATH"
* Выбрать опцию "Install Now" или "Customize installation"
* Завершить установку
* Проверить в командной строке: python --version

2. Отличие Anaconda от официального Python

Anaconda:

* Включает Python + 1500 популярных научных пакетов
* Собственный менеджер пакетов conda
* Упрощенное управление виртуальными окружениями
* Готовое окружение для data science

Официальный Python:

* Только интерпретатор Python
* Менеджер пакетов pip
* Требует ручной установки дополнительных библиотек

3. Проверка работоспособности Anaconda

Пуск → Anaconda3 (64-bit) → Prompt

jupyter notebook

New → Python

После чего ввести простейшую программу. Нажать Alt+Enter.

4. Настройка интерпретатора в PyCharm

File → Settings → Project → Python Interpreter

Settings → Add → Выбрать нужный интерпретатор

5. Запуск программы в PyCharm

Правой кнопкой по файлу → Run 'filename'

6. Режимы работы Python:

Интерактивный и пакетный режимы работы с Python — это два подхода к запуску программ на интерпретируемом языке Python. Суть режимов — интерпретация строк кода, вводимых с клавиатуры, и выполнение файлов с исходным кодом.

Интерактивный режим: пользователь пишет в консоли команду, и как только нажимает клавишу Enter, команда выполняется. Такой режим ещё называют циклом «чтения — исполнения — печати» (Read-eval-print loop, REPL).

Пакетный режим: пользователь пишет код в файле и сохраняет его с нужным расширением. Интерпретатор получает файл, построчно считывает написанный там код и выполняет его.

Особенности:

7. Динамическая типизация

Python определяет типы переменных во время выполнения, а не компиляции.

8. Основные типы данных

Числовые: int, float, complex, bool;последовательности: str, list, tuple, range; логические: bool.

9. Создание объектов и переменных

Создание объекта:

x = 5

10. Список ключевых слов

import keyword

…..

print(keyword.kwlist)

11. Функции id() и type()

id() - возвращает уникальный идентификатор объекта

type() - возвращает тип объекта

12. Изменяемые и неизменяемые типы в python

Изменяемые типы — это объекты, содержимое которых можно менять после создания. Изменения происходят без создания нового объекта, все ссылки отражают его текущее состояние.

Примеры:

* списки (list) — упорядоченные коллекции объектов, доступных по индексу;
* словари (dict) — ассоциативные массивы для хранения данных в формате «ключ-значение»;
* множества (set) — коллекции уникальных элементов.

Неизменяемые типы — это объекты, которые нельзя изменить после создания. Любая операция, предполагающая изменение, приводит к созданию нового объекта.

Примеры:

* числа (int, float, complex) — используются для работы с числами различных форматов;
* строки (str) — текстовые данные, которые нельзя изменить после создания;
* кортежи (tuple) — упорядоченные коллекции.

13. Операции деления

/ - обычное деление (возвращает float)

// - целочисленное деление (возвращает int)

% - остаток от деления

14. Комплексные числа в Python:

z = 3 + 4j

print(z.real) *# 3.0*

print(z.imag) *# 4.0*

x = complex(2,3) *# (2+3j)*

15. Модули math и cmath

Модуль math в Python — это встроенный модуль для работы с числами и математическими операциями. Он входит в стандартную библиотеку Python, поэтому его не нужно устанавливать отдельно.  Модуль предоставляет функции для арифметических, тригонометрических, логарифмических и других операций, а также константы.

**Модуль cmath в Python** предоставляет доступ к математическим функциям для работы с комплексными числами (числами, которые состоят из действительной и мнимой части). Этот модуль расширяет функциональность стандартного модуля math, позволяя выполнять математические операции не только с вещественными, но и с комплексными числами.

16. Параметры sep и end в print()

Sep — параметр функции print() в Python, который задает, какой символ или строку ставить между элементами в выводе. По умолчанию sep = « » (пробел).

Параметр end в функции print() в Python (от слова end — «конец») определяет, какой символ будет добавлен в конец строки после вывода текста. По умолчанию print() завершает вывод новой строкой (\n), но можно изменить это поведение.

print(1, 2, 3, sep='-', end='!') *# 1-2-3!*

17. Форматирование строк

Метод format() в Python — инструмент для форматирования строк, который позволяет вставлять значения переменных, значения выражений или даже результаты выполнения функций внутрь строки.

formatted\_string = "Фамилия: {1}, Имя: {0}".format(first\_name, last\_name)

f-строки позволяют встраивать переменные прямо в строку. Чтобы задать f-строку, необходимо поставить букву f перед открывающей кавычкой строки.

print(f"Меня зовут {name}, и мне {age} лет.")

%-форматирование: старый стиль форматирования, работает по принципу подстановки. Указывается, какие значения нужно подставить и в каком виде, с помощью символа % и специальных символов-плейсхолдеров.

print("Меня зовут %s, и мне %d лет." % (name, age))

Конкатенация: "Hello " + name

18. Ввод числовых переменных осуществляется с помощью метода input с указанием требуемого типа.

h = int(input("Введите число"))

h = float(input("Введите число"))

Вывод: исследовали процесс установки и базовые возможности языка Python версии 3.х.