

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ-
СКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №2.1

Условные операторы и циклы в языке Python

Выполнил студент группы

ИВТ-б-о-21-1

Богадунов В.И. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

(подпись)

Ставрополь 2022

Тема: Условные операторы и циклы в языке Python.
№1. Создать новый репозиторий и выполнить задания указанные в методических указаниях.

ItsMyLife1337

/

LB.2.1

Great repository names are short and memorable. Need inspiration? How about [literate-octo-train?](#)

Description (optional)

Для лабораторной работы 2.1

☒

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

This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

Рисунок 1 – Создание нового репозитория

ItsMyLife1337 Update .gitignore Latest commit bef70b3 now History

1 contributor

268 lines (212 sloc) 5.26 KB Raw Blame

```
1
2 # Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
3 # Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm
4
5 ### PyCharm ###
6 # Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio, WebStorm and Rider
7 # Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839
8
9 # User-specific stuff
10 .idea/**/workspace.xml
11 .idea/**/tasks.xml
12 .idea/**/usage.statistics.xml
13 .idea/**/dictionaries
14 .idea/**/shelf
15
16 # AWS User-specific
17 .idea/**/aws.xml
```

Рисунок 2 – Изменённый файл .gitignore для работы с pycharm

```
Microsoft Windows [Version 10.0.19044.1645]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Admin> cd/d c:\users\Admin\desktop\git

c:\Users\Admin\Desktop\git>git clone https://github.com/ItsMyLife1337/Forprograms2.1
Cloning into 'Forprograms2.1'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 4.40 KiB | 750.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.

c:\Users\Admin\Desktop\git>
```

Рисунок 3 – Клонировал репозиторий на свой компьютер

```
C:\Users\Admin> cd/d c:\users\Admin\desktop\git

c:\Users\Admin\Desktop\git> cd/d c:\users\Admin\desktop\git\forprograms2.1

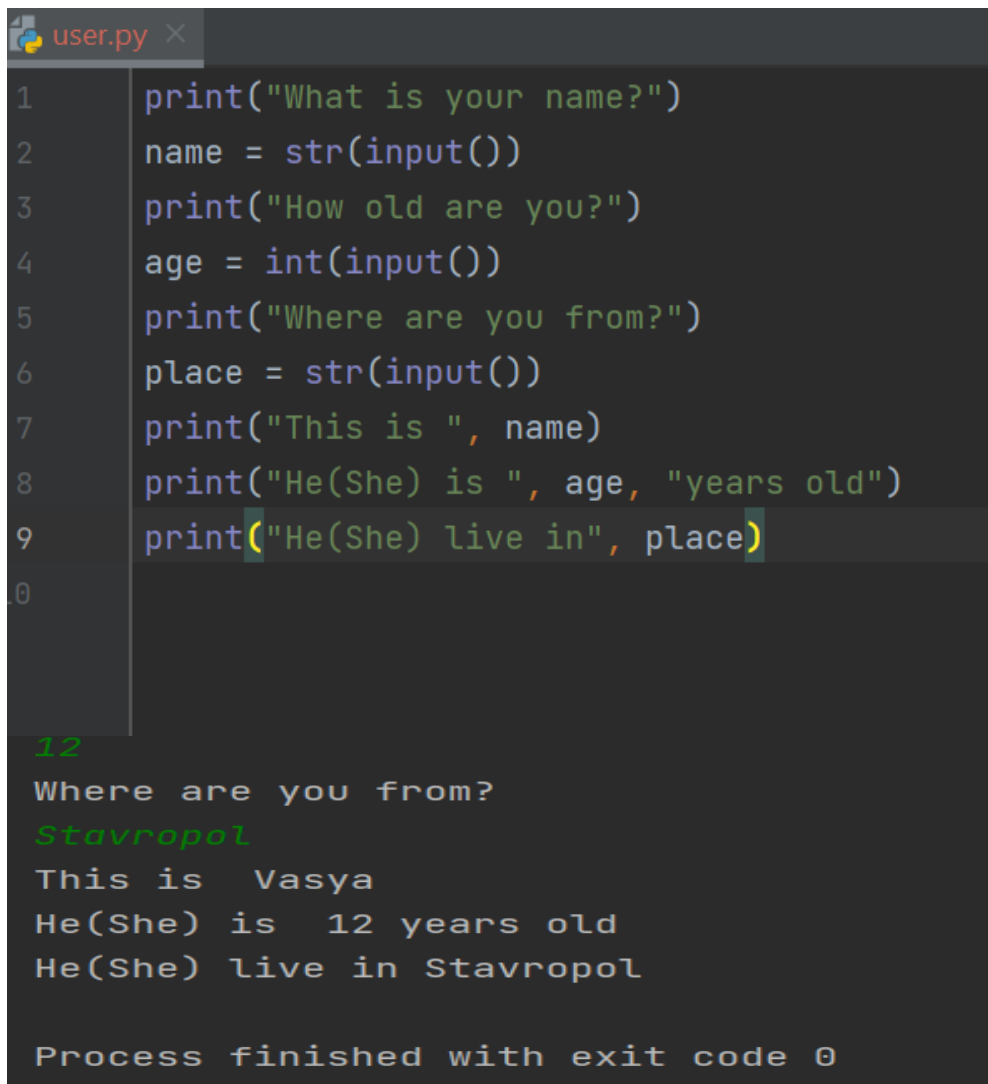
c:\Users\Admin\Desktop\git\Forprograms2.1>git flow init

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/Admin/Desktop/git/Forprograms2.1/.git/hooks]

c:\Users\Admin\Desktop\git\Forprograms2.1>
```

Рисунок 4 – Организация репозитория согласно модели git-flow

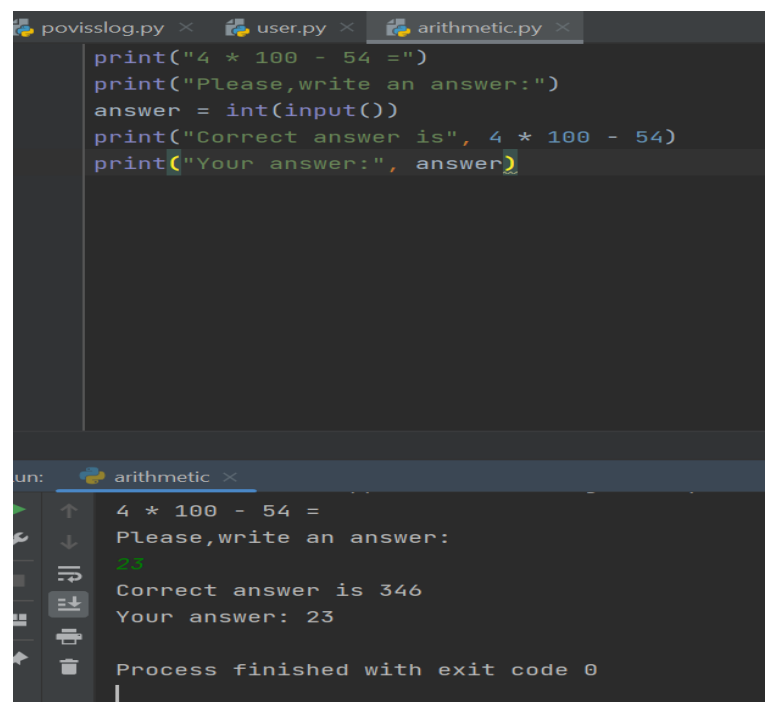


```
user.py x
1 print("What is your name?")
2 name = str(input())
3 print("How old are you?")
4 age = int(input())
5 print("Where are you from?")
6 place = str(input())
7 print("This is ", name)
8 print("He(She) is ", age, "years old")
9 print("He(She) live in", place)
0

12
Where are you from?
Stavropol
This is Vasya
He(She) is 12 years old
He(She) live in Stavropol

Process finished with exit code 0
```

Рисунок 5 – Задача 1

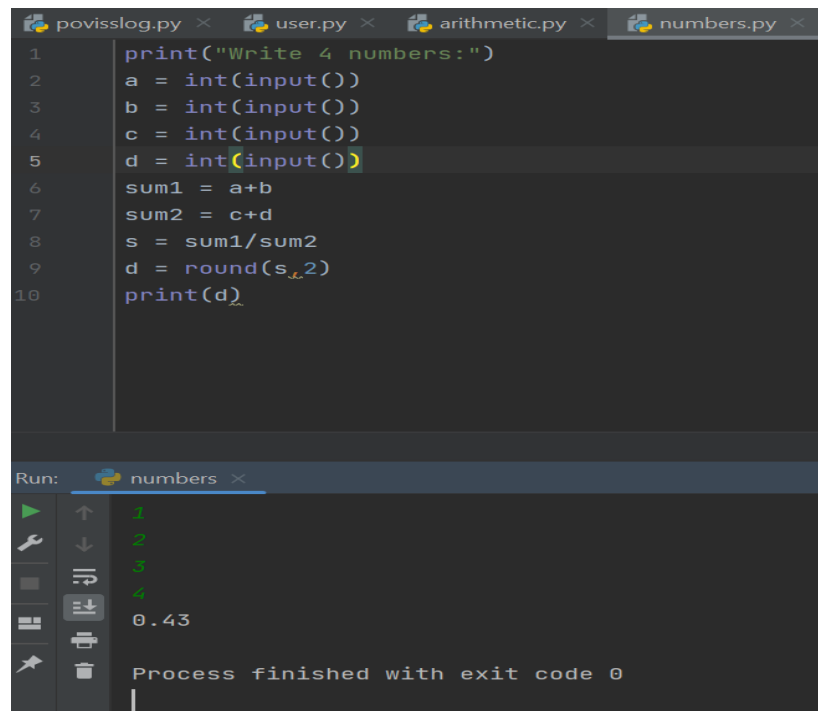


```
povisslog.py x user.py x arithmetic.py x
print("4 * 100 - 54 =")
print("Please,write an answer:")
answer = int(input())
print("Correct answer is", 4 * 100 - 54)
print("Your answer:", answer)

un: arithmetic x
4 * 100 - 54 =
Please,write an answer:
23
Correct answer is 346
Your answer: 23

Process finished with exit code 0
```

Рисунок 6 – Задача 2



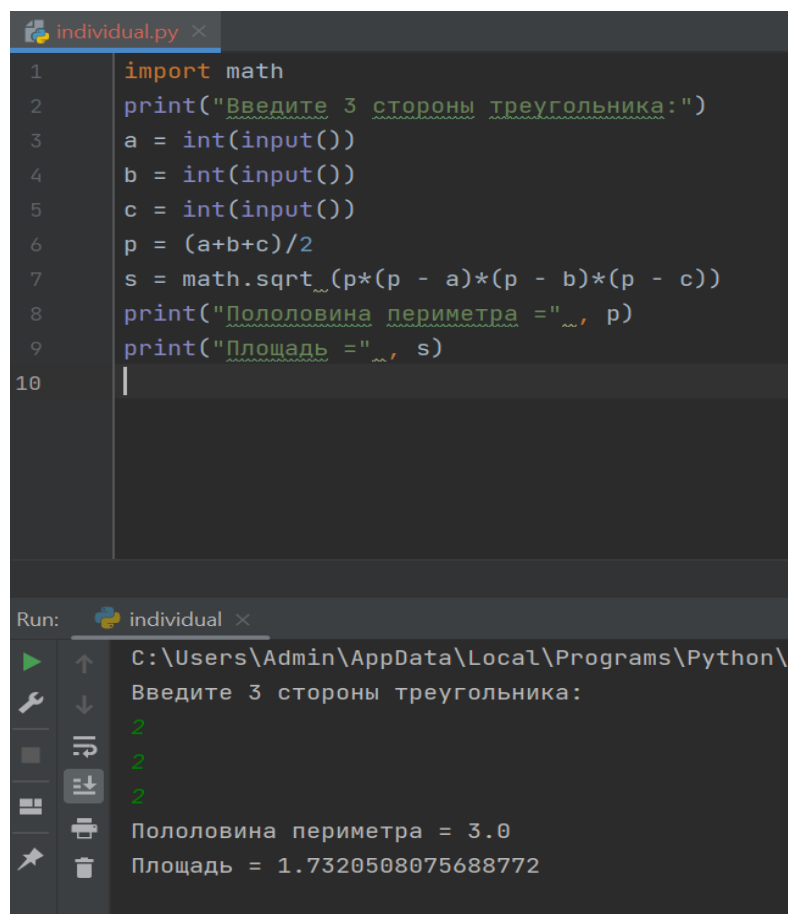
```
1 print("Write 4 numbers:")
2 a = int(input())
3 b = int(input())
4 c = int(input())
5 d = int(input())
6 sum1 = a+b
7 sum2 = c+d
8 s = sum1/sum2
9 d = round(s,2)
10 print(d)
```

Run: numbers ×

1
2
3
4
0.43

Process finished with exit code 0

Рисунок 7 – Задача 3

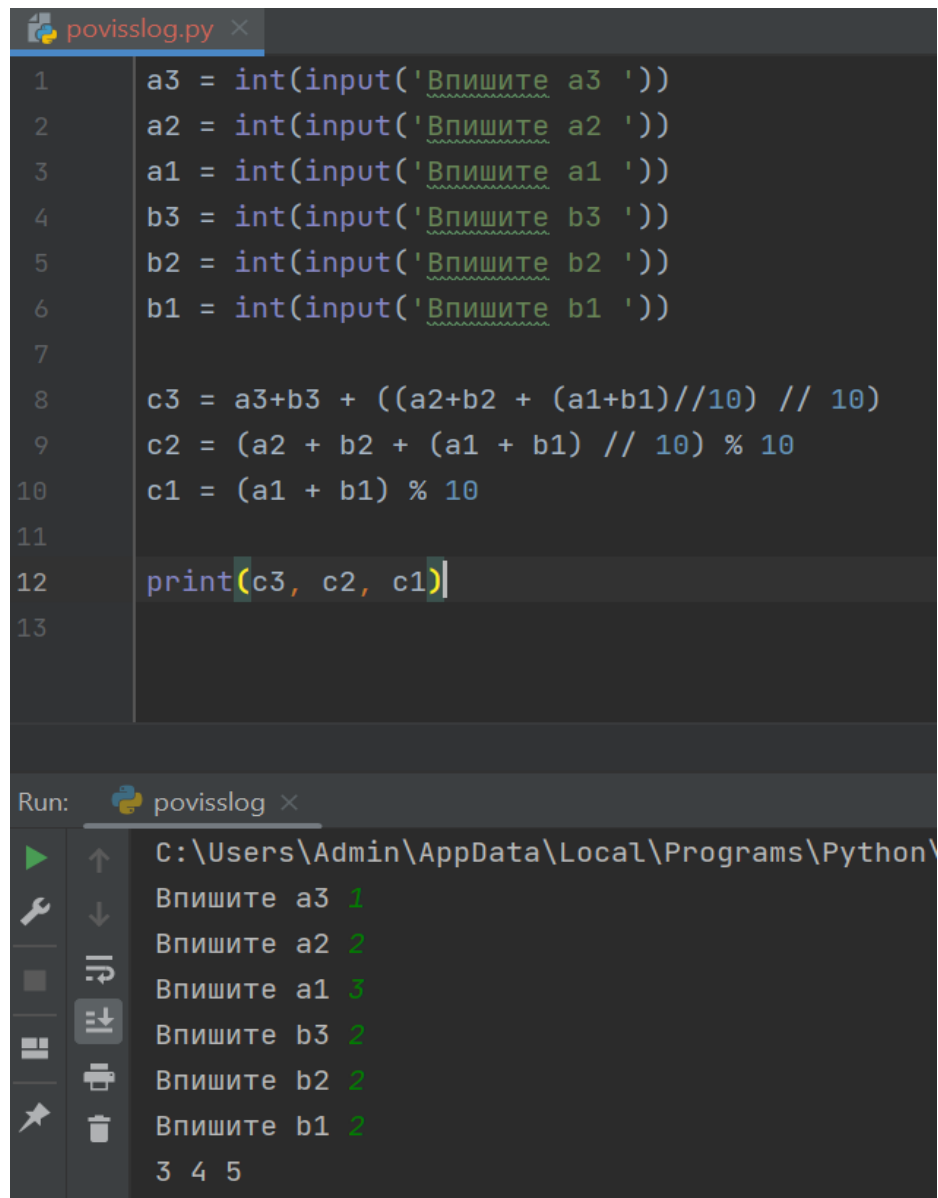


```
1 import math
2 print("Введите 3 стороны треугольника:")
3 a = int(input())
4 b = int(input())
5 c = int(input())
6 p = (a+b+c)/2
7 s = math.sqrt(p*(p - a)*(p - b)*(p - c))
8 print("Половина периметра =", p)
9 print("Площадь =", s)
10
```

Run: individual ×

C:\Users\Admin\AppData\Local\Programs\Python\
Введите 3 стороны треугольника:
2
2
2
Половина периметра = 3.0
Площадь = 1.7320508075688772

Рисунок 8 – Индивидуальное задание В – 3



```
povisslog.py x
1 a3 = int(input('Впишите a3 '))
2 a2 = int(input('Впишите a2 '))
3 a1 = int(input('Впишите a1 '))
4 b3 = int(input('Впишите b3 '))
5 b2 = int(input('Впишите b2 '))
6 b1 = int(input('Впишите b1 '))
7
8 c3 = a3+b3 + ((a2+b2 + (a1+b1)//10) // 10)
9 c2 = (a2 + b2 + (a1 + b1) // 10) % 10
10 c1 = (a1 + b1) % 10
11
12 print(c3, c2, c1)
13

Run: povisslog x
C:\Users\Admin\AppData\Local\Programs\Python\
Впишите a3 1
Впишите a2 2
Впишите a1 3
Впишите b3 2
Впишите b2 2
Впишите b1 2
3 4 5
```

Рисунок 9 – Задача повышенной сложности В - 3

№2. Сделать коммит изменений в ветку разработки, выполнить её слияние с веткой main и отправить сделанные изменения на удалённый репозиторий.

```

c:\Users\Admin\Desktop\git\Forprograms2.1>git branch
* develop
  main

c:\Users\Admin\Desktop\git\Forprograms2.1>git add .
warning: LF will be replaced by CRLF in .idea/inspectionProfiles/profiles_settings.xml.
The file will have its original line endings in your working directory

c:\Users\Admin\Desktop\git\Forprograms2.1>git commit -m "Added programs"
[develop fad380b] Added programs
11 files changed, 80 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/Forprograms2.1.iml
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 arithmetic.py
create mode 100644 individual.py
create mode 100644 numbers.py
create mode 100644 povisslog.py
create mode 100644 user.py

c:\Users\Admin\Desktop\git\Forprograms2.1>

```

Рисунок 2.1 – Коммит изменений в ветку develop

```

c:\Users\Admin\Desktop\git\Forprograms2.1>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

c:\Users\Admin\Desktop\git\Forprograms2.1>git merge develop
Updating bef70b3..fad380b
Fast-forward
 .idea/.gitignore           | 3 +++
 .idea/Forprograms2.1.iml    | 8 ++++++++
 .idea/inspectionProfiles/profiles_settings.xml | 6 ++++++
 .idea/misc.xml             | 4 ++++
 .idea/modules.xml          | 8 ++++++++
 .idea/vcs.xml              | 6 ++++++
 arithmetic.py              | 5 +++++
 individual.py              | 9 ++++++++
 numbers.py                 | 10 ++++++++
 povisslog.py               | 12 ++++++++
 user.py                    | 9 ++++++++
11 files changed, 80 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/Forprograms2.1.iml
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 arithmetic.py

```

Рисунок 2.2 – Слияние ветки develop с веткой main

```
c:\Users\Admin\Desktop\git\Forprograms2.1>git push
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 4 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (15/15), 2.30 KiB | 391.00 KiB/s, done.
Total 15 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/ItsMyLife1337/Forprograms2.1
   bef70b3..fad380b  main -> main

c:\Users\Admin\Desktop\git\Forprograms2.1>
```

Рисунок 2.3 – Отправка изменений на удалённый репозиторий

main
1 branch
0 tags
Go to file
Add file
Code

ItsMyLife1337 Added programs
 fad380b 22 minutes ago
3 commits

	.idea	Added programs	22 minutes ago
	.gitignore	Update .gitignore	5 days ago
	LICENSE	Initial commit	5 days ago
	README.md	Initial commit	5 days ago
	arithmetic.py	Added programs	22 minutes ago
	individual.py	Added programs	22 minutes ago
	numbers.py	Added programs	22 minutes ago
	povisslog.py	Added programs	22 minutes ago
	user.py	Added programs	22 minutes ago

README.md

LB.2.1

Рисунок 2.4 – Зафиксировал изменения на удалённом репозитории

Ответы на контрольные вопросы:

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

Linux:

Чаще всего интерпретатор Python уже входит в состав дистрибутива.

Windows: Оsn. этапы установки Python на Windows:

- 1) Скачать дистрибутив с официального сайта;
- 2) Запустить скачанный установочный файл;
- 3) Выбрать способ установки;
- 4) Отметить необходимые опции установки;
- 5) Выбрать место установки;
- 6) Готово.

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 и исполнения, запускаемую в браузере, а также на Anaconda удобнее запускать примеры.

3. Как осуществить проверку работоспособности пакета Anaconda?

Для выполнения проверки работоспособности Anaconda необходимо вначале запустить командный процессор с поддержкой виртуальных окружений Anaconda. В появившейся командной строке необходимо ввести `> jupyter notebook`, в результате чего отобразится процесс загрузки веб-среды Jupyter Notebook, после чего запустится веб-сервер и среда разработки в браузере. Создать ноутбук для разработки, для этого нажать на кнопку New и в появившемся списке выбрать Python. В результате будет создана новая страница в браузере с ноутбуком. Ввести в первой ячейке команду `print("Hello, World!")` и нажать Alt+Enter на компьютере. 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 с выбором интерпретатора;
- 4) Укажите путь до интерпретатора.

5. Как осуществить запуск программы с помощью IDE PyCharm?

Сочетанием клавиш Shift+F10.

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

Интерактивный. 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.

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

7. Почему язык программирования Python называется языком динамической типизации?

Т. к. в ЯП Python проверка типа происходит во время выполнения, а не компиляции.

8. Какие существуют основные типы в языке программирования Python?

Типы в ЯП 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 объявления новых переменных и работа операции присваивания?

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?

Список ключевых слов можно получить непосредственно в программе, для этого 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 (frozenset). 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 комплексносопряженного 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`. Для выполнения математических операций необходим модуль `math`. Осн. операции библиотеки `math`:

`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 π .

`math.e` - число e .

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

Через параметр `sep` можно указать отличный от пробела разделитель строк. Параметр `end` позволяет указывать, что делать, после вывода строки.

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 переменных в языке Python?

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

Вывод: исследовал процесс установки и базовые возможности языка программирования Python версии 3.