Московский государственный технический университет
им. Н.Э. Баумана (национальный исследовательский университет)

ИНФОРМАТИКА. ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ PYTHON

Учебно-методическое пособие

Москва
Издательство МГТУ им. Н.Э. Баумана
2024

УДК 004.43
ББК 32.973

Факультет Международных образовательных программ
Кафедра «Русский язык как иностранный»
Рекомендовано Научно-методическим советом
МГТУ им. Н. Э. Баумана в качестве учебно-методического пособия
Автор:
Р.И. Жамалетдинов
Рецензент: _____

Информатика. Программирование на языке Python. : учебно-методическое пособие / Р.И. Жамалетдинов — Москва : Издательство МГТУ им. Н.Э. Баумана, 2024. — 40 с.

ISBN……

Учебно-методическое пособие по дисциплине «Информатика» предназначено для иностранных слушателей, обучающихся в российских вузах на подготовительном отделении. Целью пособия является проверка знаний, умений и навыков, полученных ранее. Предназначено для самостоятельной работы слушателей. Данное учебно-методическое пособие базируется на программе «Информатика».

УДК 004.43
ББК 32.973

Предисловие

Данное учебно-методическое пособие носит учебно-контролирующий характер. Предназначено для иностранных слушателей, обучающихся в российских вузах на подготовительном отделении. Пособие подготовлено в соответствии с Рабочей программой дисциплины «Информатика». В учебно-методическом пособии даны самостоятельные работы для проверки знаний иностранных слушателей.

Цель пособия состоит в том, чтобы систематизировать уже имеющиеся у учащихся знания по информатике, обобщить умения и навыки в учебной и научной сферах. **Задачей** пособия является помощь в усвоении материала по информатике и его закрепление. Мы знаем, что важным условием приобретения прочных знаний является сознательное, систематическое усвоения программного материала, поэтому огромное значение для иностранцев при изучении информатики является самостоятельная работа над учебным материалом.

За время обучения иностранные слушатели выполняют самостоятельно задания по информатике, которые являются отчётом слушателя о проделанной работе, состоящей из изучения материала, изложенного преподавателем на занятии и выполнения письменных заданий.

Для продуктивного **выполнения** самостоятельных работ **рекомендуется** разбить работу на несколько этапов.

- Прочитать тему конкретной самостоятельной работы.
- Выполнять самостоятельную работу следует по отдельным тематическим блокам. Взяв один блок, слушатель должен вспомнить всю информацию, которую он получил об этой теме на уроках информатики.
- Выполняя конкретное задание, слушатель может использовать учебник, справочные материалы. Умение соотносить теорию с практикой – одно из важнейших при закреплении материала в изучении информатики.
- Слушателям рекомендуется выполнять задания последовательно, не «перепрыгивая» с темы на тему.

• Нужно обдуманно подходить к выполнению самостоятельной работы. В случае вопроса преподавателя слушатель должен смочь объяснить, чем он руководствовался в том или ином случае. Даже ошибочное объяснение лучше, чем бездумное выполнение.

• По окончании необходимо ещё раз просмотреть работу целиком во избежание неточностей и опечаток. И лишь потом сдавать её на проверку. Самостоятельная работа – это закрепление пройденного материала, поэтому лишь чёткое и скрупулёзное её выполнение даст необходимый результат.

Существуют следующие **требования к выполнению** самостоятельной работы.

Задания выполняются на компьютере. Самостоятельные работы, выполненные небрежно или неполностью, возвращаются для доработки. Слушатели, не выполнившие самостоятельные, на зачёт не допускаются.

Критерий оценивания:

• 100% - 60% – «зачтено»

• менее 60% – «не зачтено»

Условия и требования защиты.

Самостоятельные работы выполняются на компьютере и сдаются преподавателю в определённый срок. После проверки проводится разбор самостоятельных работ (Работа над ошибками) в группе слушателей. Слушатели имеют возможность ознакомиться с замечаниями преподавателя и проанализировать допущенные в работе ошибки. После этого каждый слушатель должен (самостоятельно или с помощью преподавателя) объяснить допущенные им ошибки и найти правильное решение. Только в таком случае (в случае подробного разбора самостоятельной работы и анализа ошибок) самостоятельные работы считаются полностью выполненными.

В конце пособия имеется Терминологический словарь.

Данное пособие используется для самостоятельной работы слушателей. Материалы прошли необходимую апробацию в иностранной аудитории.

Самостоятельная работа 1. Знакомство с Python

Задание 1.

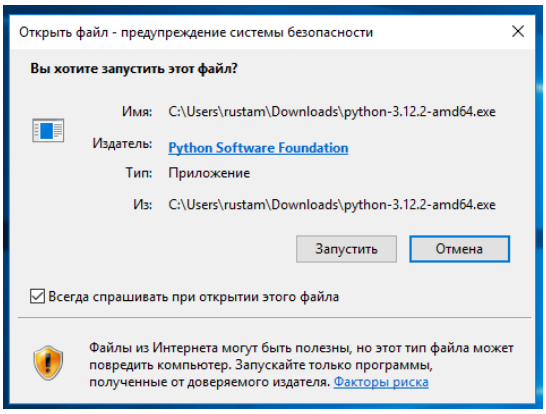
Установить интерпретатор на OS Windows.

Инструкция к выполнению.

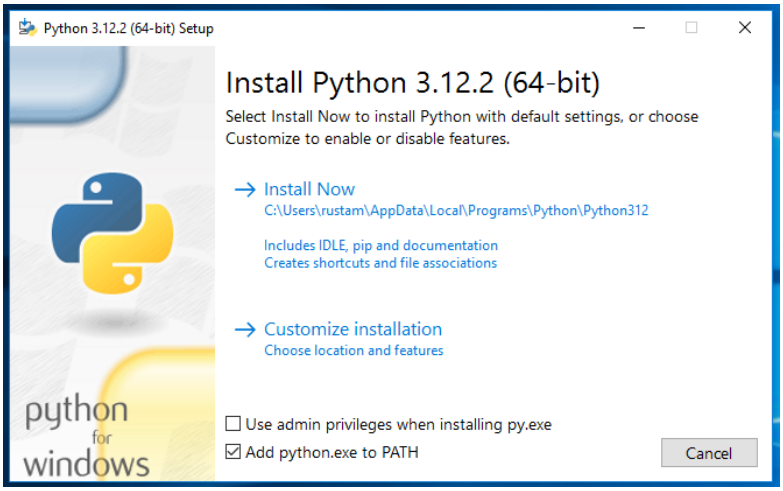
1. Заходим на сайт https://www.python.org/
2. Перейдите на вкладку СКАЧАТЬ (download), система автоматически определит установленную на ваш компьютер операционную систему, в противном случае выбираем Windows.

3. После того как скачали загрузочный файл, двойным кликом левой клавиши мыши запускаем процесс установки.

4. Появляется окно, предупреждения системы безопасности. Нажимаем на кнопку «Запустить».

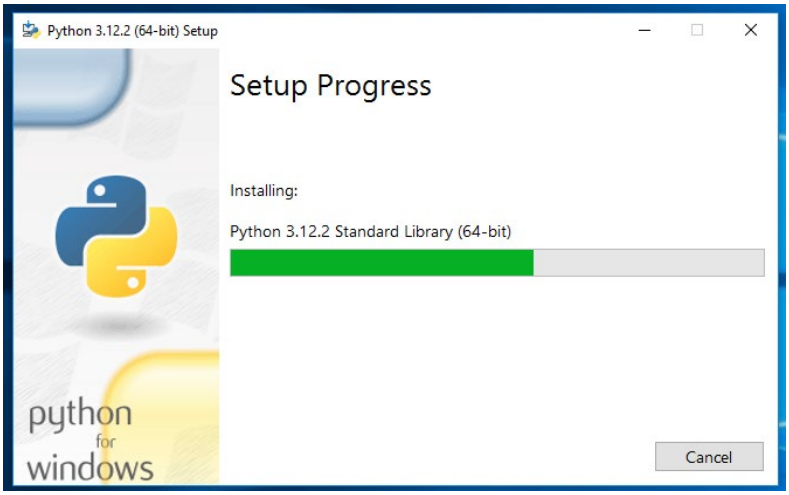


5. Окно установки интерпретатора Python. В нашем случаи будет устанавливаться версия 3.12.2.

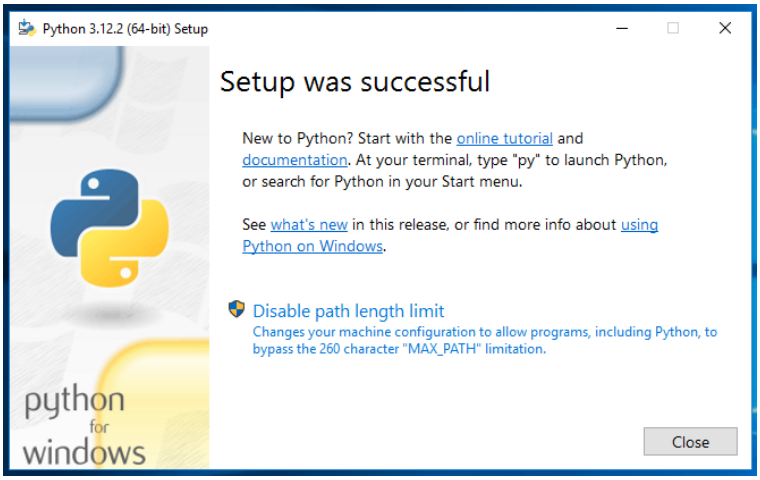


- Ставим **галочку** напротив **Add python to PATH** это необходимо для того, чтобы появилась возможность запускать интерпретатор без указания полного пути до исполняемого файла при работе в командной строке.
- Нажимаем на **Install Now**

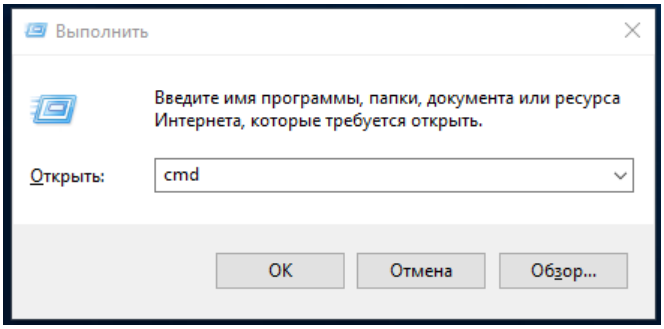
6. Окно процесса установки интерпретатора.



7. После завершения процесса установки интерпретатора появиться окно «Установка прошла успешно». Данное окно также позволяет узнать, что нового в данной версии интерпретатора, для этого необходимо кликнуть **online tutorial** (онлайн руководство) или ознакомиться с документацией, нажимаем на **documentation**. Так же окно дает возможность «отключить» ограничения длины имени в 260 символов. Этого делать мы не будет т.к. наш курс вводный. Для завершения нажимает на **Close**.



После установки интерпретатора, нам необходимо убедиться в том, что интерпретатор установлен и работает корректно. Для этого, нажимаем сочетание клавиш “**WIN + R**”, в появившемся окне пишем “**cmd**”, далее нажимаем “**OK**”



В открывшемся окне командной строки напишем **python** и нажмем клавишу **ввода** (Enter).

```
C:\Users\rustam>python
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Можем проверить работу интерпретатора, для этого напишем **2 + 2** и нажмем клавишу **ввода** (Enter).

```
C:\Users\rustam>python
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 2
4
>>>
```

Для выхода напишем **exit()** или **quit()** и нажмем клавишу **ввода** (Enter)

```
C:\Users\rustam>python
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 2
4
>>> exit()

C:\Users\rustam>
```

8. В окне командной строки пишем “**Python -V**” что бы узнать установленную версию интерпретатора, в нашем случаи установлена версия 3.12.2

```
C:\Users\rustam>python -V
Python 3.12.2
```

Задание 2.

Установить интерпретатор на OS Ubuntu:

В операционной системе Ubuntu версии 22.04.4 по умолчанию установлен интерпретатор Python. Для проверки установленной версии в терминале вводим команду **python3 -V**

```
rustam@rustam-VirtualBox:~$ python3 -V
Python 3.10.12
```

1. До обновления Python до последней версии необходимо обновить операционную систему. Для этого вводим:

```
rustam@rustam-VirtualBox:~/Desktop$ sudo apt update
```

2. Далее обновляем все устаревшие пакеты:

```
rustam@rustam-VirtualBox:~/Desktop$ sudo apt upgrade -y
```

3. Импортируем PPA Python 3.12 в Ubuntu. Для пользователей Ubuntu лучший способ установить Python 3.12 — использовать PPA Launchpad. Вводим следующую команду, чтобы импортировать стабильный PPA для Python 3.12

```
rustam@rustam-VirtualBox:~/Desktop$ sudo add-apt-repository ppa:deadsnakes/ppa -y
```

4. После импорта PPA необходимо обновить индекс APT, чтобы он отражал вновь добавленный репозиторий.

```
rustam@rustam-VirtualBox:~/Desktop$ sudo apt update
```

5. Поскольку PPA Python 3.12 теперь является частью репозиториев вашей системы, вы можете приступить к установке Python 3.12. Выполните следующую команду для установки Python:

```
rustam@rustam-VirtualBox:~/Desktop$ sudo apt install python3.12 -y
```

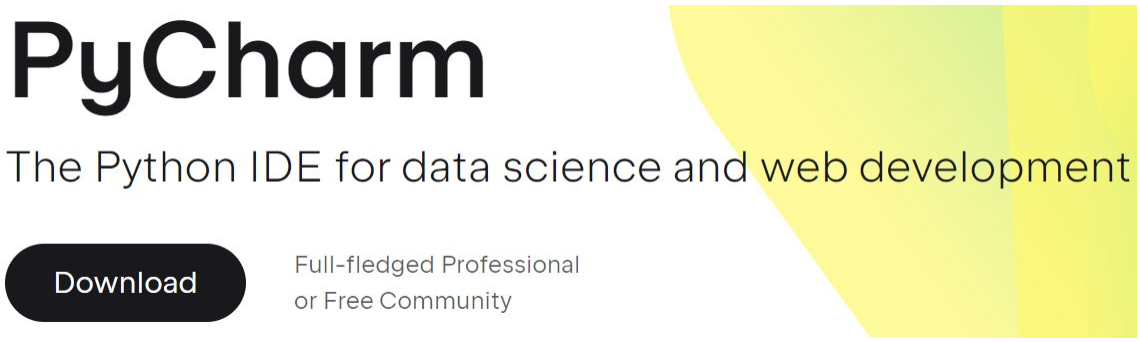
6. Чтобы подтвердить успешную установку и проверить версию сборки Python, используйте команду:

```
rustam@rustam-VirtualBox:~/Desktop$ python3.12 -V
Python 3.12.2+
```

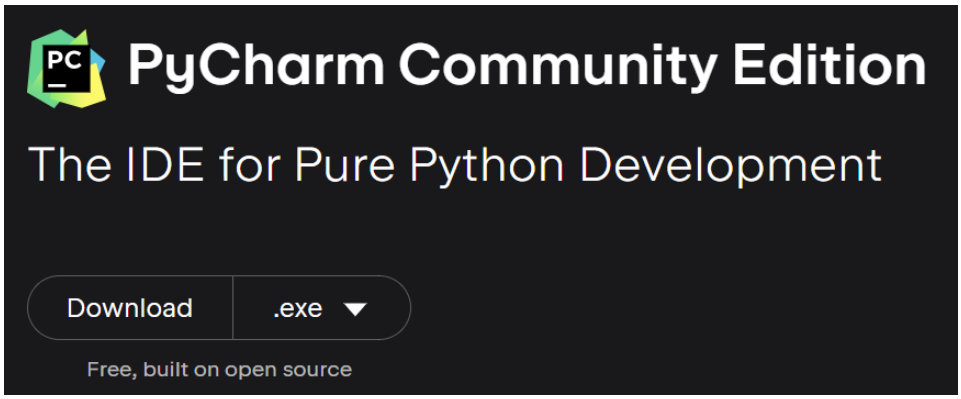
Задание 3.

Установить интегрированную среду разработки на ОС Windows.

Перейдем на сайт разработчика по адресу https://www.jetbrains.com/pycharm/

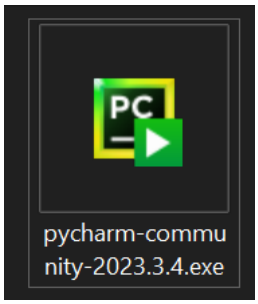


Нажмем на иконку Download для перехода на страницу скачивания необходимой версии.

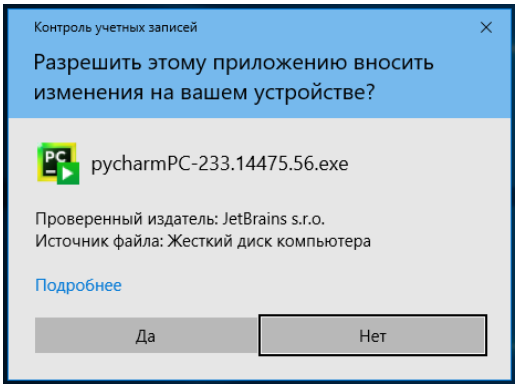


Для наших занятий будет достаточно версии Community Edition. Кликаем на иконку Download.

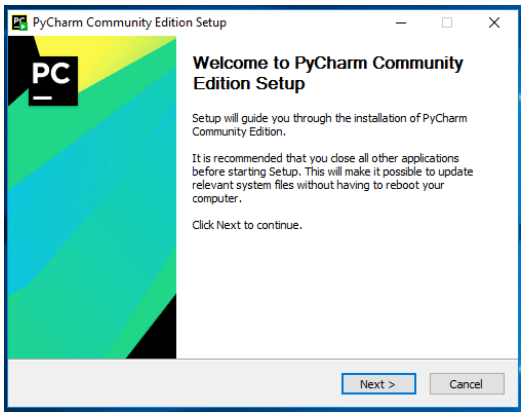
После окончания скачивания файла pycharm-community-2023.3.4.exe



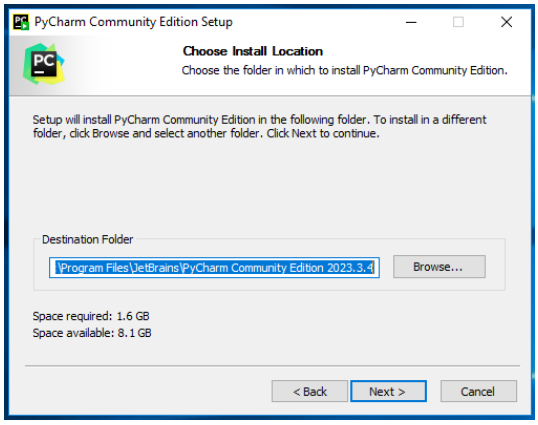
Двойным кликом по файлу инициируйте процесс установки. После, разрешаем данному приложению внести изменения на наше устройство.



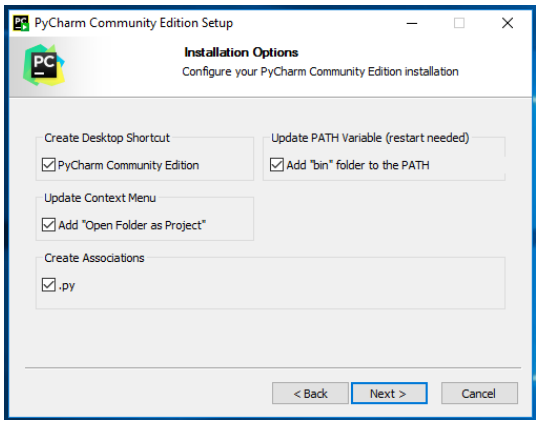
Окно приветствия, нажимаем **Next>**



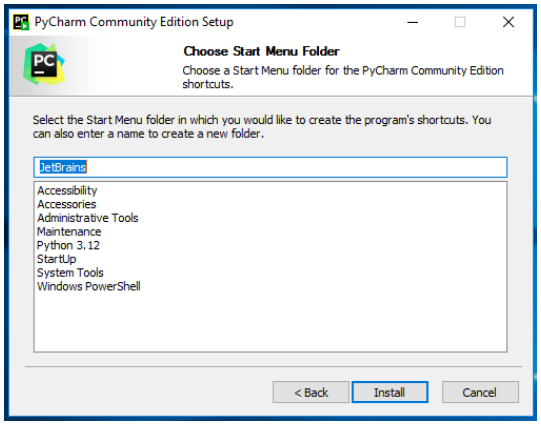
Выбираем папку, в которую будет установлен PyCharm. При необходимости изменить папку нажимаем на **Browse…** В нашем случаи нажимаем **Next>**



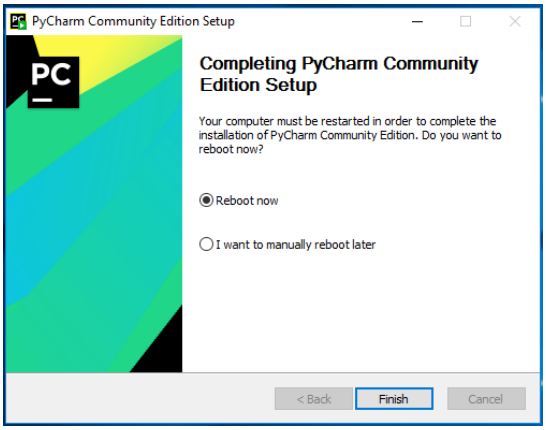
Окно вариантов установки.



Выбираем папку меню «ПУСК». Нажимаем на кнопку **Install**



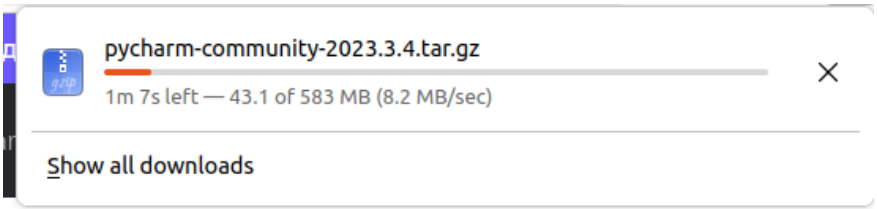
После успешной установки появляется окно Завершения, для вступления изменений в силу, перезагрузим компьютер. Выбираем **Reboot now** и нажимаем на кнопку **Finish**



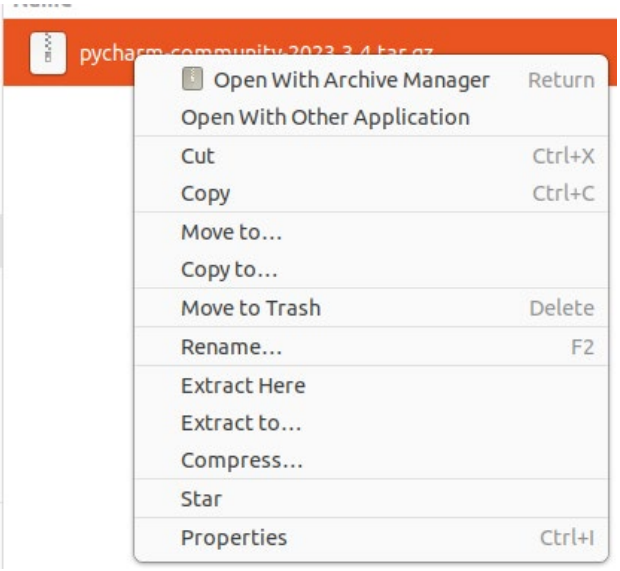
Задание 4.

Установить интегрированную среду разработки на ОС Ubuntu.

Скачиваем архив с сайта https://www.jetbrains.com/



Кликаем по архиву правой клавишей мыши, вызывая контекстное меню, выбираем «извлечь в» (Extract to) указываем необходимую директорию. В нашем случаи распаковываем в директорию «Program», которую мы создали заранее.



Для запуска приложения PyCharm в начале откроем терминал и введем следующую строчку:

```
@rustam-VirtualBox:~$ cd Program/pycharm-community-2023.3.4/bin/
```

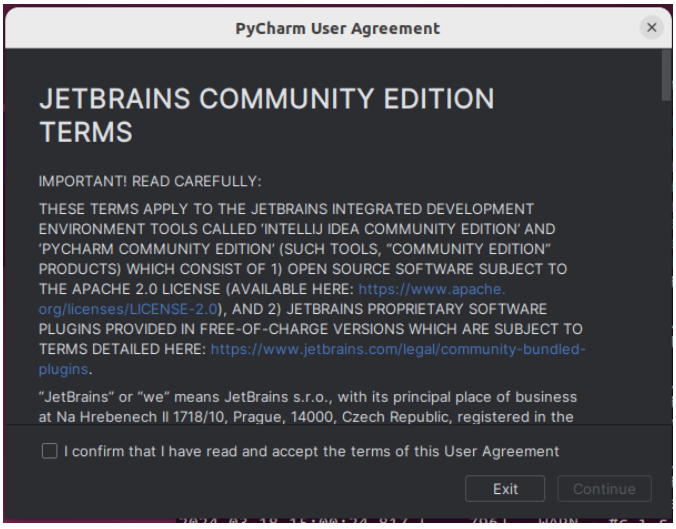
Затем при помощи команды «**ls**» просмотрим содержание директории и убеждаемся в наличие bash скрипта (pycharm.sh)

```
rustam@rustam-VirtualBox:~/Program/pycharm-community-2023.3.4/bin$ ls
brokenPlugins.db  jetbrains_client64.vmoptions  pycharm.png
format.sh         jetbrains_client.sh           pycharm.sh
fsnotifier        libdbm.so                     pycharm.svg
idea.properties   ltedit.sh                     repair
inspect.sh        pycharm64.vmoptions           restarter
```

Запустим скрипт.

```
rustam-VirtualBox:~/Program/pycharm-community-2023.3.4/bin$ ./pycharm.sh
```

После того как мы нажмем «Enter», запустится PyCharm. Первое что предложит программа – это согласиться с пользовательским соглашением. Дальнейшие действия идентичны действиям в ОС Windows, которые мы рассмотрим ниже.



Задание 5.

Создать ярлык для IDE PyCharm в OS Ubuntu

Каждый раз для запуска программы PyCharm нам необходимо будет открывать терминал и прописывать пусть до скрипта и запускать его, это не совсем удобно. Для устранения данного неудобства, создадим «ярлык» PyCharm.

1. Перейдем в домашнюю директорию, просмотрим содержание данной директории и создадим файл с названием «pycharm.desktop».

✓
```
rustam@rustam-VirtualBox:/usr/share/applications$ cd ~
rustam@rustam-VirtualBox:~$ ls
Desktop    Downloads  Pictures  Public  Templates
Documents  Music      Program   snap    Videos
```

✓
```
@rustam-VirtualBox:~$ touch pycharm.desktop
```

✓
```
rustam@rustam-VirtualBox:~$ ls
Desktop    Downloads  Pictures  Public          snap       Videos
Documents  Music      Program   pycharm.desktop  Templates
```

2. Откроем созданный нами файл pycharm.desktop в редакторе nano.

✓
```
@rustam-VirtualBox:/usr/share/applications$ nano pycharm.desktop
```

3. Перепишите и сохраните текст сочетаниями клавиш: Ctrl+O, Enter, Ctrl+X

```
GNU nano 6.2                    pycharm.desktop *
[Desktop Entry]
Version=1.0
Name=PyCharm
Exec=Program/pycharm-community-2023.3.4/bin/./pycharm.sh
Icon=Program/pycharm-community-2023.3.4/bin/pycharm.png
Terminal=false
Type=Application
Categories=Development;IDE;
```

4. Перемещаем наш файл в нужную нам директорию:

```
sudo mv pycharm.desktop /usr/share/applications/
```

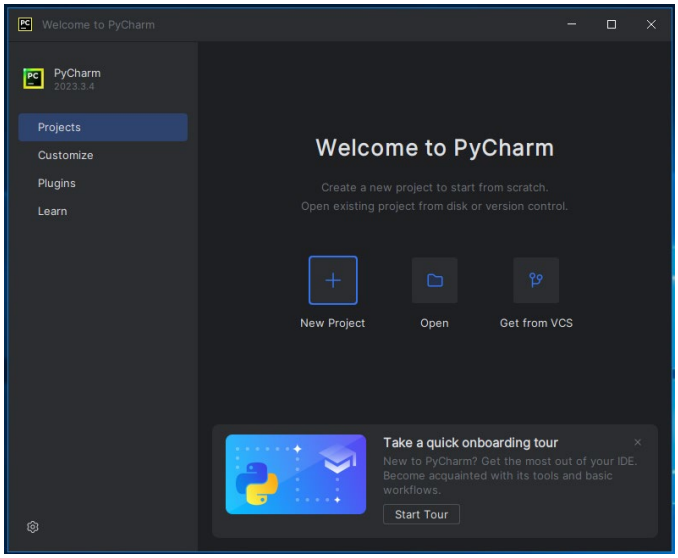
5. После чего в панели приложений должен появиться наш ярлык, облегающий запуск программы PyCharm.

Задание 6.

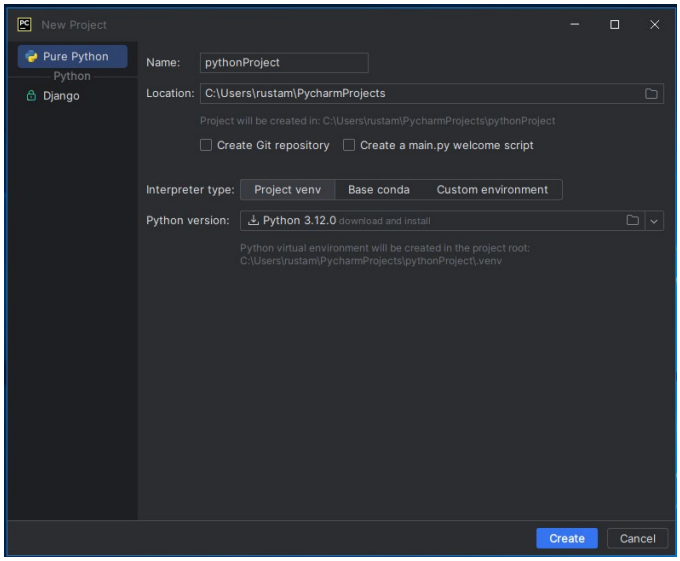
Выполнить первый запуск IDE PyCharm на ОС Windows.

После того как мы согласились с лицензионным соглашением, мы соглашаемся с отправкой анонимных сообщений по статистики работы программы.

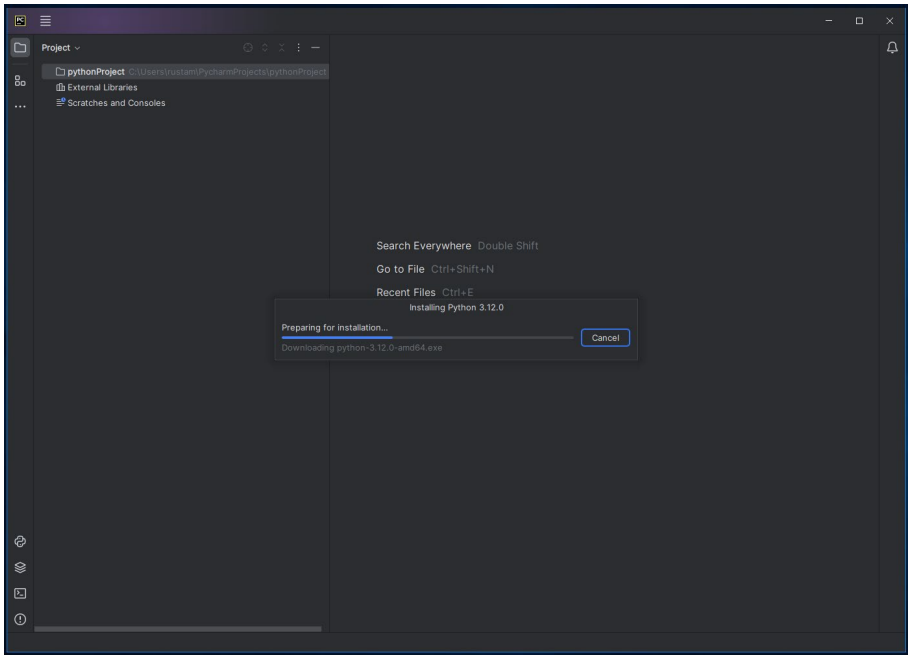
Создаем новый проект, нажимаем на «**New Project**»



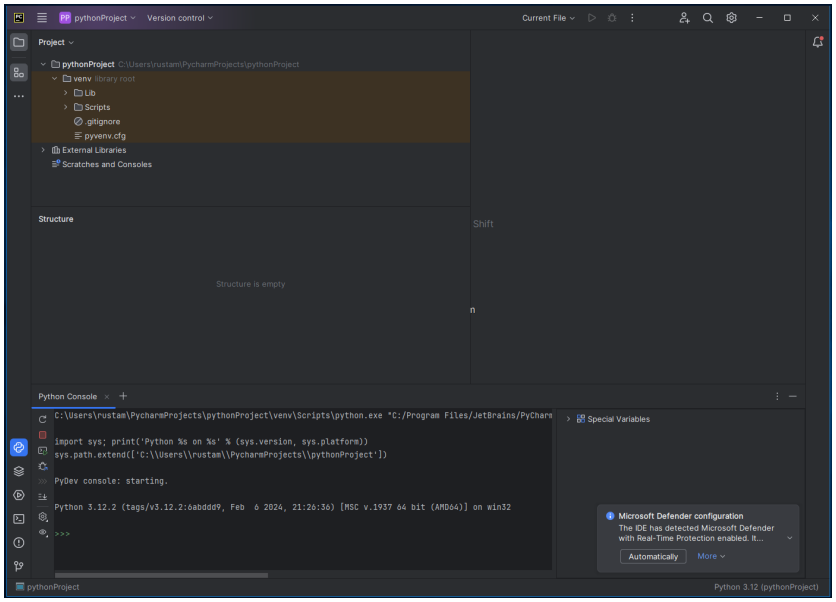
В окне «New project» мы можем присвоить имя проекту, изменить расположение проекта и версию Python. После внесенных изменений нажимаем «Create».



Окно «Подготовки к установке».

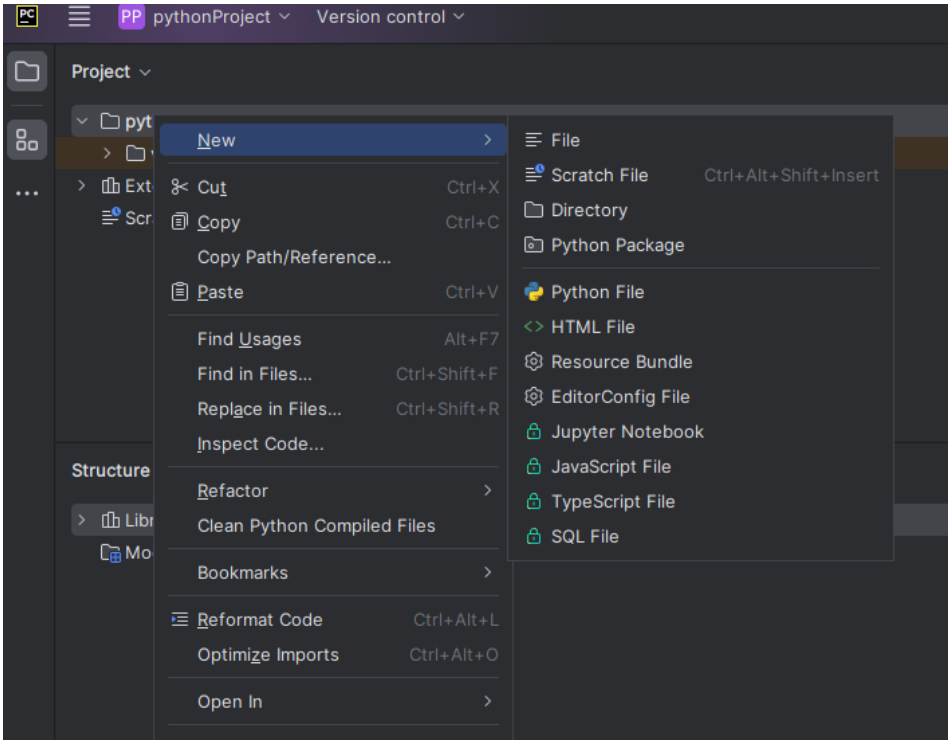


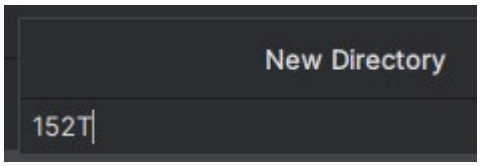
Окно нашего проекта «pythonProject»



Создать папку внутри проекта.

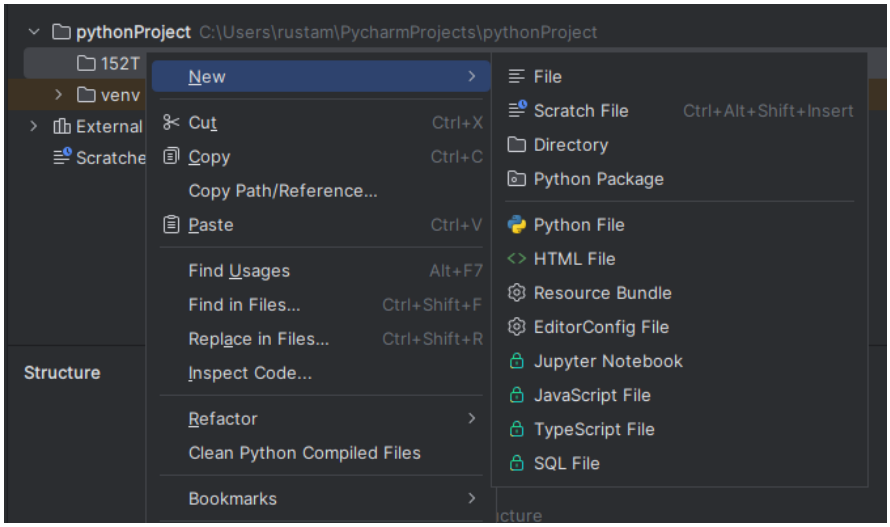
- Выбрать «pythonProject»;
- Нажать правую клавишу мыши;
- В открывшемся меню: New → Directory → в появившемся окне написать номер своей группы → нажать «Enter»

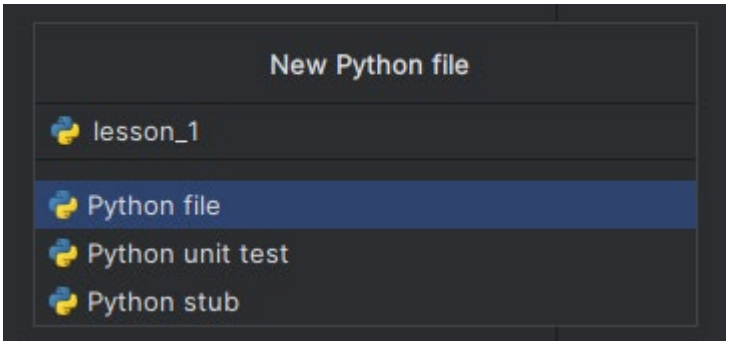




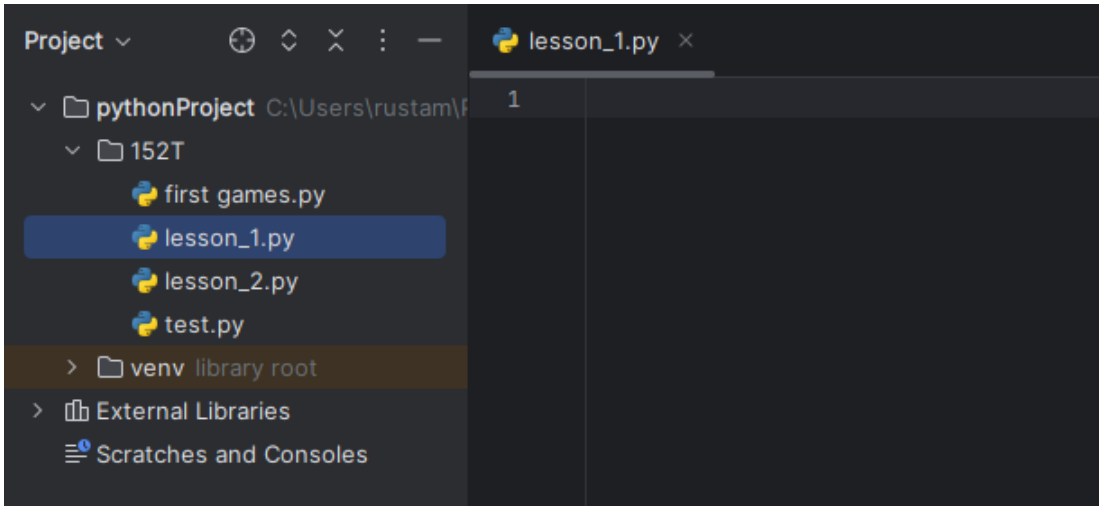
Создать внутри вашей папки «**152T**» python файл:

- Выбрать папку «**152T**»;
- Нажать правую клавишу мыши;
- В открывшемся меню: **New → Python File →** в окне написать **lesson_1 →** нажать «**Enter**»





После нажатия клавиши «Enter» в правой части экрана появится вкладка lesson_1, в которой пишем код.



Задание 7.

Выполнить первый запуск IDE PyCharm на ОС Ubuntu

Выполнять аналогично **Заданию 6.** (смотрите выше)

Самостоятельная работа 2. Ввод-вывод данных.

Задание 1.

Напишите программу, которая выводит на экран текст: Я учусь в МГТУ им. Н.Э. Баумана!

Задание 2.

Напишите программу, которая выводит на экран число Эйлера и число золотого сечения (каждое число на отдельной строке).

Задание 3.

Напишите программу, которая считывает Ваше имя и выводит на экран текст: Здравствуй …!

Задание 4.

Напишите программу, которая выводит на экран текст: Мне ^_^нравиться^_^учиться^_^в^_^МГТУ.

Задание 5.

Допишите программу из Задания 3., таким образом, чтоб восклицательный знак (!) выступал параметром окончания строки.

Задание 6.

Напишите программу вывода на экран четырёх последовательно идущих чисел, каждое на отдельной строке. Первое число вводите Вы, остальные числа вычисляет программа.

Задание 7.

Напишите программу, которая выводит на экран сумму трёх введенных чисел (каждое на отдельной строке).

Задание 8.

Напишите программу, которая находит дискриминант квадратного уравнения:

$5x^2 – 9x – 2 = 0$.

Самостоятельная работа 3. Условный оператор.

Задание 1.

Напишите программу, которая сравнивает введенный пароль и его подтверждение, пароль должен состоять только из букв. Если они совпадают, то программа выводит: «Пароль принят», иначе: «Пароль не принят».

Задание 2.

Напишите программу, которая определяет, является введённое число четным или нечетным.

Задание 3.

Напишите программу, которая определяет наименьшее из веденых двух чисел.

Задание 4.

Напишите программу, которая принимает три положительных числа и определяет, существует ли невырожденный треугольник с такими сторонами. Программа должна вывести «Да» или «Нет» в соответствии с условием задачи.

Задание 5.

Напишите программу, которая определяет, является ли год с данным номером високосным. Если год является високосным, то выведите «Да», иначе выведите «Нет». Год является високосным, если его номер кратен 4, но не кратен 100, или если он кратен 400.

Задание 6.

Напишите программу, которая рассчитает стоимость покупки с учетом скидки в 35%, которая предоставляется покупателю, если сумма покупки превышает 2000 рублей. Сумму покупки ввести с клавиатуры, а результаты округлить до сотых (копейки). Вывести на экран итоговую стоимость и размер предоставленной скидки.

Задание 7.

Напишите программу, которая принимает три положительных числа, определяет вид треугольника (равносторонний, равнобедренный, разносторонний) и выводит сообщение о виде треугольника, длины сторон которого равны введенным числам.

Задание 8.

Напишите программу, которая считывает с клавиатуры два целых числа и строку. Если эта строка является обозначением одной из четырёх математических операций (+, -, *, /), то выведите результат применения этой операции к введённым ранее числам, в противном случае выведите «Неверная операция». Если пользователь захочет поделить на ноль, выведите текст «На ноль делить нельзя!»

Самостоятельная работа 4. Типы данных.

Задание 1.

На вход программе подается число n – количество собачьих лет. Напишите программу, которая вычисляет возраст собаки в человеческих годах по следующему алгоритму: в течение первых двух лет собачий год равен 10.5 человеческим годам, после этого каждый год собаки равен 4 человеческим годам.

Задание 2.

Напишите программу, которая определяет, какой температуре по шкале Цельсия соответствует указанное значение по шкале Фаренгейта. На вход программе подаётся вещественное число градусов по шкале Фаренгейта. Программа должна вывести число градусов по шкале Цельсия.

Задание 3.

Напишите программу, которая считывает с клавиатуры название футбольной команды и выводит фразу: «Футбольная команда [введённая строка] имеет длину [длина введённой строки] символов». На вход программе подаётся строка – название футбольной команды.

Задание 4.

Напишите программу, которая считывает одну строку, после чего выводит "ДА", если во введенной строке есть подстрока "суббота" или "воскресенье", и "НЕТ" в противном случае.

Задание 5.

Будем считать e-mail адрес корректным, если в нем есть символ собачки (@) и точки (.). Напишите программу, проверяющую корректность e-mail адреса.

Программа должна вывести строку «ДА», если e-mail адрес является корректным и «НЕТ» в ином случае.

Задание 6.

Напишите программу, вычисляющую значение тригонометрического выражения **sin x+cos x+tg^2 x** (функция **radians()** переводит угол из градусов в угол в радианах).

Задание 7.

Даны три вещественных числа a, b, c. Напишите программу, которая находит вещественные корни квадратного уравнения: $ax^2 + bx + c = 0$

Задание 8.

Напишите программу, которая решает квадратное уравнение: $5x^2 – 9x – 2 = 0$.

Самостоятельная работа 5. Циклы for и while.

Задание 1.

Напишите программу, которая считывает одну строку текста (Ваше имя) и выводит 10 строк, пронумерованных от 0 до 9, каждая с указанной строкой текста.

Задание 2.

На вход программе подается натуральное число n. Напишите программу, которая для каждого из чисел от 0 до n (включительно) выводит фразу: Квадрат числа [**число**] равен [**число**]

Задание 3.

Дано натуральное число n. Напишите программу, которая выводит таблицу умножения на n (от 1 до 10 включительно).

Задание 4.

Напишите программу, которая считывает последовательность из 10 целых чисел и определяет, является ли каждое из них четным или нет.

Задание 5.

Существуют монеты с номиналами 1, 5, 10, 25. Напишите программу, которая определяет, какое минимальное количество чеканных монет нужно заплатить за товар. На вход программе подается одно натуральное число - цена за товар.

Задание 6.

На обработку поступает последовательность из 10 целых чисел (каждое на отдельной строке). Нужно написать программу, которая выводит на экран количество неотрицательных чисел последовательности и их произведение. Если неотрицательных чисел нет, требуется вывести на экран «НЕТ». Программист торопился и написал программу неправильно.

Найдите все ошибки в этой программе (их ровно 4). Известно, что каждая ошибка затрагивает только одну строку и может быть исправлена без изменения других строк. Примечание. При необходимости вы можете добавить необходимые строки кода.

```
count = 0
p = 0
for i in range(1, 10):
    x = int(input())
    if x > 0:
        p = p * x
        count = count + 1
if count > 0:
    print(x)
    print(p)
else:
    print('НЕТ')
```

Задание 7.

На обработку поступает натуральное число. Нужно написать программу, которая выводит на экран максимальную цифру числа, кратную 3. Если в числе нет цифр, кратных 3, требуется на экран вывести «НЕТ». Найдите все ошибки в этой программе (их ровно 5). Известно, что каждая ошибка затрагивает только одну строку и может быть исправлена без изменения других строк.

Примечание 1. Число 0 делится на любое натуральное число.

Примечание 2. При необходимости вы можете добавить нужные строки кода.

```
n = int(input())
max_digit = n % 10
while n > 0:
    digit = n % 10
    if digit % 3 == 0:
        if digit < max_digit:
            digit = max_digit
    n = n % 10
if max_digit == 0:
    print('НЕТ')
else:
    print(max_digit)
```

Самостоятельная работа 6. Строковый тип данных.

Задание 1.

На вход программе подается одна строка. Напишите программу, которая выводит элементы строки с индексами 0, 2, 4, ... в столбик.

Формат входных данных. На вход программе подается одна строка.

Формат выходных данных. Программа должна вывести элементы строки с индексами 0, 2, 4, ..., каждое на отдельной строке.

Задание 2.

На вход программе подается одна строка. Напишите программу, которая выводит в столбик элементы строки в обратном порядке.

Формат входных данных. вход программе подается одна строка.

Формат выходных данных. Программа должна вывести в столбик элементы строки в обратном порядке.

Задание 3.

На вход программе подается одно слово, записанное в нижнем регистре. Напишите программу, которая определяет, является ли оно палиндромом.

Формат входных данных. На вход программе подается одно слово в нижнем регистре.

Формат выходных данных. Программа должна вывести «ДА», если слово является палиндромом, и «НЕТ» в противном случае.

Примечание. Палиндром считается слово, которое читается одинаково в обоих направлениях. Например, слово «**мадам**» является палиндромом.

Задание 4.

На вход программе подается строка текста. Напишите программу, которая разрежет ее на две равные части, переставит их местами и выведет на экран.

Формат входных данных. На вход программе подается строка текста.

Формат выходных данных. Программа должна вывести текст в соответствии с условием задачи.

Примечание. Если длина строки нечетная, то длина первой части должна быть на один символ больше.

Входные данные (input)	Выходные данные (output)
Qwerty	rtyQwe
Hello	loHel

Самостоятельная работа 7. Списки. Функции.

Задание 1.

Дополните приведенный код, используя индексатор, так чтобы он вывел 17-ый элемент списка primes.

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71]
print()
```

Задание 2.

Дополните приведенный код так, чтобы он вывел "перевёрнутый" список languages (т.е. элементы будут идти в обратном порядке).

Примечание. Для вывода списка воспользуйтесь функцией print().

languages = ['Chinese', 'Spanish', 'English', 'Hindi', 'Arabic', 'Bengali', 'Portuguese', 'Russian', 'Japanese', 'Lahnda']

Задание 3.

Напишите программу, выводящую следующий список:

['a', 'bb', 'ccc', 'dddd', 'eeeee', 'ffffff', ...]

Формат выходных данных. Программа должна вывести указанный список.

Примечание. Последний элемент списка должен состоять из 26 символов z.

Задание 4.

Дополните приведенный код, так чтобы он вывел сумму квадратов элементов списка numbers.

numbers = [1, 78, 23, -65, 99, 9089, 34, -32, 0, -67, 1, 11, 111]

Задание 5.

На вход программе подается натуральное число n, а затем n различных натуральных чисел. Напишите программу, которая удаляет наименьшее и наибольшее значение из указанных чисел, а затем выводит оставшиеся числа каждое на отдельной строке, не меняя их порядок.

Формат входных данных. На вход программе подаются натуральное число n, а затем n различных натуральных чисел, каждое на отдельной строке.

Входные данные (input)	Выходные данные (output)
10 9 17 189 3 55 78 11 7 888 160	9 17 189 55 78 11 7 160
7 1 2 3 4	2 3 4 5 6

5 6 7	

Задание 6.

На вход программе подается строка текста, содержащая имя, отчество и фамилию человека. Напишите программу, которая выводит инициалы человека, разделенные точкой.

Задание 7.

На вход программе подается строка текста, содержащая 4 целых неотрицательных числа, разделенных точкой. Напишите программу, которая определяет, является ли введенная строка текста корректным ip-адресом.

Задание 8.

Напишите функцию print_fio(name, surname, patronymic), которая принимает три параметра:

- name – имя человека;
- surname – фамилия человека;
- patronymic – отчество человека;

а затем выводит на печать ФИО человека.

Примечание. Предусмотрите тот факт, что все три буквы в ФИО должны иметь верхний регистр.

Входные данные (input)	Выходные данные (output)
Александр Пушкин Сергеевич	ПАС
Николай Гоголь Васильевич	ГНВ

Терминологический словарь

= - оператор присваивания

== - условный оператор сравнения

abs() – возвращает абсолютное значение числа. Аргумент может быть простым или длинным целым числом или числом с плавающей запятой. Если аргумент является комплексным числом, возвращается его величина.

and – логическое умножение (конъюнкция, И, &, ∧), - истинна только в том случае, когда все высказывания, для которых выполняется конъюнкция, истинны.

break (прервать) – преждевременное прерывание выполнения ближайшего цикла (for, while).

continue (продолжение, продолжать) – позволяет перейти к следующей итерации цикла for или while до завершения всех команд в теле цикла.

def – инструкция, которая используется для определения функций и методов.

elif - объединение else и if (иначе, если) и используется для проверки нескольких условий. Записывается после оператора if для проверки альтернативного условия, если первое условие ложно. Блок кода под оператором elif будет выполнен только в том случае, если его условие истинно.

else - ИНАЧЕ.

```
if условие:
    блок кода
else:
    блок кода
```

end – окончание, параметр команды print(). Параметр применяется, если нет необходимости делать перевод строки или требуется указать определенный символ для окончания строки. Пример:

```
1  print('Как', 'тебя', 'зовут?', end='&&&')
2  name = input()
3  print('Меня зовут', name)
```

False – ложь (0)

for (для) - цикл, повторяющиеся определенное количество раз (for, счетные циклы, counting loops). Пример:

```
for i in range(10):
    print('Привет')
```

float() – преобразует строку или число в формат с плавающей запятой.

if – ЕСЛИ.

```
if условие:
    блок кода
```

if x != 5 – (оператор сравнения) если x не равно 5

if x < 5 – (оператор сравнения) если x меньше 5

if x <=5 – (оператор сравнения) если x меньше либо равно 5

if x == 5 – (оператор сравнения) если x равно 5 (проверяет равно ли x пяти)

if x > 5 – (оператор сравнения) если x больше 5

if x >= 5 – (оператор сравнения) если x больше либо равно 5

in – оператор, проверяет наличие одной строки в другой строке.

input() – команда для ввода (считывания) данных. По умолчанию принимает строковый тип данных. Пример:

```
1  print('Как тебя зовут?')
2  name = input()
3  print('Меня зовут', name)
```

int() – преобразует строку или число в простое целое число. Если аргумент является строкой, он должен содержать десятичное число со знаком.

len() – возвращает длину (количество элементов) объекта. Аргументом может быть последовательность (строка, кортеж или список) или отображение (словарь).

math – модуль (библиотека) предоставляющий возможность проведения вычислений с вещественными числами (числами с плавающей точкой). Для использования этих функций в начале программы необходимо подключить модуль, что делается командой import. Пример:

```
import math
x = math.sqrt(121)
print(x)
```

max() – возвращает наибольший элемент повторяющегося аргумента.

min() – возвращает наименьший элемент повторяющегося аргумента.

not – логическое отрицание (инверсия, НЕ, ¬ , $\bar{A}$). Если высказывание А истинно, то после инверсии оно станет ложным, и наоборот.

or – логическое сложение (дизъюнкция, ИЛИ, |, ∨) - истинна в том случае, когда хотя бы одно из высказываний истинно.

print() – команда предназначенная для вывода данных. В случаи вывода текста, он должен быть помещен в кавычки (одинарные или двойные) при этом не забываем, что кавычки должны быть одинаковыми в начале и в конце. Пример:

```
print('Меня зовут - Иван')
```

range (диапазон) – указывает на количество повторений цикла.

sep (separator – разделитель) параметр команды print(). \n – заменяет пробел (по умолчанию) на новую строку. Пример:

```
print('Как', 'тебя', 'зовут?', sep='\n')
name = input()
print('Меня зовут', name)
```

str() – преобразование числа в строку.

True – истина (1)

while (пока) - цикл, повторяющиеся до наступления определенного события (while, условные циклы, conditional loops). Пример:

```
num = int(input())
while num != -1:
    print('Куб вашего числа равен:', num ** 3)
    num = int(input())
```

Вложенный оператор – использование любой инструкции языка Python внутри условного оператора. Пример:

```
if условие1:
    блок кода
else:
    if условие2:
        блок кода
    else:
        if условие3:
            блок кода
    ...
```

Возведение в степень (**) – арифметическая операция, первоначально определяемая как результат многократного умножения числа на себя. Степень с основанием **a** и натуральным показателем **b** обозначается как

$$a^b = \underbrace{a \cdot a \cdot \ldots \cdot a}_{b},$$

где **b** – количество множителей (умножаемых чисел)

Вычитание (-) – арифметическое действие двух аргументов (уменьшаемого и вычитаемого), результатом которой является новое число (разность), получаемое уменьшением значения первого аргумента на значение второго аргумента.

Деление (/) – это арифметическое действие, обратное умножению. С помощью деления по произведению и одному из множителей определяется второй множитель. Компоненты деления:

Делимое — число, стоящее слева от знака деления, которое делят.

Делитель — число, стоящее справа от знака деления, число, на которое делят делимое.

Частное — число, стоящее после знака равно, результат деления, числовое выражение со знаком деления.

Деление с остатком (%) – это деление одного натурального числа на другое, при котором остаток меньше делителя. В языке программирования Python деление с остатком возвращает остаток. Результатом работы кода print(10 % 3) будет 1

Дистрибутив (англ. distribution, от англ. distribute — распространять) — форма распространения программного обеспечения. Согласно ГОСТ Р 53623-2009 – Форма распространения программного обеспечения, обычно содержащая

программу-установщик (для выбора режимов и параметров установки) и набор файлов, содержащих отдельные части программного средства.

Индекс предложений по улучшению Python (Index of Python Enhancement Proposals (PEP)) – документ содержит индексы всех предложений по улучшению Python, известных как PEP. Номера PEP назначаются редакторами PEP, и после присвоения они никогда не изменяются. История контроля версий текстов PEP представляет собой их историческую запись.

Интегрированная среда разработки (Integrated Development Environment (IDE)) – программа, для написания, проверки, тестирования и запуск кода, а также ведения больших проектов. Она включает в себя сразу несколько инструментов: редактор для написания кода, сервисы для его проверки и запуска, расширения для решения дополнительных задач разработки. Примеры: PyCharm, Visual Studio Code, Spyder и т.д.

Индексация строк – обращение к определенному символу в строке. В Python используются квадратные скобки [], в которых указывается индекс (номер) нужного символа в строке.

Интерпретатор – языковый процессор, который построчно анализирует исходную программу и одновременно выполняет предписанные действия, а не формирует на машинном языке скомпилированную программу, которая выполняется впоследствии.

Интерпретируемый язык программирования – язык программирования, выполнение кода которого осуществляется непосредственно и построчно, то есть команда за командой. В этом случае программа не может быть запущена без интерпретатора. Пример языков: PHP, JavaScript, Python, Lava, MATLAB, Perl и другие.

Итерация – единичное повторение того или иного действия. В программирование указывает количество повторений цикла.

Каскадный условный оператор – применяется для проверки нескольких условий. Пример:

if *условие1*:

блок кода

elif *условие2*:

блок кода

else:

блок кода

Компилируемый язык программирования – язык программирования, исходный код которого преобразуется компилятором в машинный код и записывается в файл с особым заголовком и/или расширением для последующей идентификации этого файла, как исполняемого операционной системой (в отличие от интерпретируемых языков программирования, чьи программы выполняются программой-интерпретатором).

Компилятор – это программа, которая предназначена для перевода (трансляции) высокоуровневого языка программирования в машинный код или на язык, близкий к машинному (например, Ассемблер). Входной информацией для компилятора является текст программы. На выходе получаем машинно-ориентированный язык.

Компиляция – трансляция программы на язык, близкий к машинному, либо трансляция программы, составленной на исходном языке, в объектный модуль. Иными словами, формируется программа, обычно с расширением .exe, которая обрабатывается непосредственно ЭВМ.

Конкатенация строк – операция сложения строк.

Операционная система – совокупность программа, обеспечивающий организацию вычислительного процесса на ЭВМ и организации взаимодействия с пользователем при помощи графического (GUI) или командного (CLI) интерфейсов.

Ревьювирование (инспекция) – систематический и периодический анализ программного кода, направленный на поиск необнаруженных на ранних стадиях разработки программного продукта ошибок, а также, на выявление некачественных архитектурных решений и критических мест в программе.

Руководство по стилю для кода Python (Style Guide for Python Code (PEP8)) – документ в котором приводятся соглашения по кодированию кода Python, входящего в стандартную библиотеку основного дистрибутива Python. Создан в июле 2001 года, авторами являются Гвидо ван Россум (Guido van Rossum), Барри Варшава (Barry Warsaw), Алиса Коглан (Alyssa Coghlan).

Синтаксис языка программирования – набор правил, описывающий комбинации символов алфавита, считающиеся правильно структурированной программой (документом) или её фрагментом. Для языка программирования Python набор правил оговаривается в PEP8.

Сложение (+) – арифметическое действие, посредством которого из двух или нескольких чисел получают новое, содержащее столько единиц, сколько было во всех данных числах вместе. Слагаемые – числа, которые складывают. Сумма – результат сложения.

Списки – структура данных (data structure) — программная единица, позволяющая хранить и обрабатывать множество однотипных и/или логически связанных данных. Пример: numbers = [2, 4, 6, 8, 10]

Срезы строк (slice) – извлечение из данной строки одного символа или некоторого фрагмента подстроки или под последовательности.

Умножение (*) – математическая операция над двумя аргументами, которые называются множителями или сомножителями. Результат умножения называется их произведением.

Целочисленное деление (//) – целая часть от деления одного числа на другое. В языке программирования Python целочисленное деление (для положительных чисел) возвращает целое число «отбрасывая» остаток. Результатом работы кода print(10 // 3) будет 3

Целочисленный тип данных – данные представляющие только числа без дробных частей.

Цикл - повторяющаяся во времени последовательность событий, процессов или явлений.

Числа с плавающей точкой – это численное представление вещественного числа в программировании, которое является его приближённым значением. Двоичное представление чисел с плавающей точкой содержит три части:

- Знаковый бит. Указывает, положительное число или отрицательное.
- Экспонента. Показывает, на какое число нужно умножать мантиссу.
- Мантисса. Фиксированное количество битов, которое выражает точность числа.

Числа с плавающей точкой бывают с одинарной и двойной точностью. С одинарной точностью состоят из 32 битов: одного знакового бита, восьми битов для экспоненты и 23 битов для мантиссы. С двойной точностью — из 64 битов: одного знакового бита, 11 битов для экспоненты и 52 битов для мантиссы.

Элемент списка – значения, заключенные в квадратные скобки и отделенные запятыми.

Список литературы

Основная

Гниденко, И. Г. Технологии и методы программирования : учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 248 с.

Федоров, Д. Ю. Программирование на python : учебное пособие для вузов / Д. Ю. Федоров. — 6-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 187 с.

Чернышев, С. А. Основы программирования на Python : учебное пособие для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 349 с.

Дополнительная

Любанович, Б. Простой Python. Современный стиль программирования. 2-е изд. — СПб.: Питер, 2021. — 592 с.: ил.

Содержание

Жамалетдинов Рустам Ирфанович