

Санкт-Петербургское государственное бюджетное  
профессиональное образовательное учреждение  
«Академия управления городской средой, градостроительства и печати»



УТВЕРЖДАЮ  
Заместитель директора  
по учебно-производственной работе  
О.В. Фомичева  
2023 г.

**МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ**  
по выполнению практических работ  
по учебной дисциплине  
**ОП.04 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ**

для специальности

**09.02.06 Сетевое и системное администрирование**

Санкт-Петербург  
2023 г.

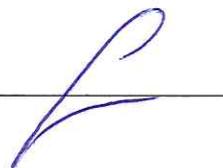
Методические рекомендации рассмотрены на заседании методического совета  
СПб ГБПОУ «АУГСГиП»

Протокол № 2 от «29» 11 2025 г.

Методические рекомендации одобрены на заседании цикловой комиссии информацион-  
ных технологий

Протокол № 4 от «29» 11 2025 г.

Председатель цикловой комиссии: Караченцева М.С. \_\_\_\_\_



Разработчики: преподаватели СПб ГБПОУ «АУГСГиП»

## СОДЕРЖАНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА .....	4
1. Перечень практических работ по дисциплине «Основы алгоритмизации и программирования» .....	6
2. Описание порядка выполнения практических работ .....	8
2.1. Практическая работа № 1 Построение линейных и разветвляющихся алгоритмов ....	8
2.2. Практическая работа № 2 Построение циклических алгоритмов.....	8
2.3. Практическая работа № 3 Разработка алгоритмов шифрования .....	9
2.4. Практическая работа № 4 Построение алгоритмов различных конструкций .....	10
2.5. Практическая работа № 5 Использование языка программирования Python для создания программ с линейным алгоритмом.....	10
2.6. Практическая работа № 6 Использование языка программирования Python для создания программ в отдельном файле с использованием строк и операторов отношений .....	11
2.7. Практическая работа № 7 Использование языка программирования Python для создания программ с цикла-ми while и for .....	12
2.8. Практическая работа № 8 Использование языка программирования Python для создания программ по работе с файлами и файловой структурой .....	13
2.9. Практическая работа № 9 Использование языка программирования Python для создания программ с использованием классов .....	13
2.10. Практическая работа № 10 Построение логически правильных и эффективных программ на языке программирования Python .....	14
2.11. Практическая работа № 11 Использование виртуальных окружений для изолирования различных проектов. Полезные функции и модули языка. ....	14
2.12. Практическая работа № 12 Использование регулярных выражений для проверки конфигурации сетевого оборудования. ....	15
2.13. Практическая работа № 13 Чтение и запись данных в форматах CSV, JSON и YAML. ....	15
2.14. Практическая работа № 14 Подключение к оборудованию по SSH и Telnet .....	15
2.15. Практическая работа № 15 Одновременное подключение к нескольким устройствам.....	16
2.16. Практическая работа № 16 Создание шаблонов конфигурации с помощью Jinja2 ..	16
2.17. Практическая работа № 17 Обработка вывода команд с помощью TextFSM .....	16
2.18. Практическая работа № 18 Использование объектно-ориентированного программирования для чтения «чужого» кода, кода netmiko. ....	17
2.19. Практическая работа № 19 Использование наследования для создания новых классов на основе существующих.....	17
2.20. Практическая работа № 20 Работа с базами данных.....	18

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Методические рекомендации по выполнению практических работ предназначены для организации работы на практических занятиях по учебной дисциплине «Основы алгоритмизации и программирования», которая является важной составной частью в системе подготовки специалистов среднего профессионального образования по специальности 09.02.06 «Сетевое и системное администрирование».

Практические занятия являются неотъемлемым этапом изучения учебной дисциплины и проводятся с целью:

- формирования практических умений в соответствии с требованиями к уровню подготовки обучающихся, установленными рабочей программой учебной дисциплины;
- обобщения, систематизации, углубления, закрепления полученных теоретических знаний;
- готовности использовать теоретические знания на практике.

Практические занятия по учебной дисциплине «Основы алгоритмизации и программирования» способствуют формированию в дальнейшем при изучении профессиональных модулей, следующих общих и профессиональных компетенций:

ОК 01. Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам.

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде.

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

ПК 1.2. Поддерживать работоспособность аппаратно-программных средств устройств инфокоммуникационных систем.

ПК 2.3. Осуществлять сбор данных для анализа использования и функционирования программно-технических средств компьютерных сетей.

В методических рекомендациях предлагаются к выполнению практические работы, предусмотренные учебной рабочей программой дисциплины «Основы алгоритмизации и программирования».

При разработке содержания практических работ учитывался уровень сложности освоения студентами соответствующей темы, общих и профессиональных компетенций, на формирование которых направлена дисциплина.

Выполнение практических работ в рамках учебной дисциплины «Основы алгоритмизации и программирования» позволяет освоить комплекс работ по выполнению практических заданий по всем темам дисциплины «Основы алгоритмизации и программирования».

Методические рекомендации по учебной дисциплине «Основы алгоритмизации и программирования» имеют практическую направленность и значимость. Формируемые в процессе практических занятий умения могут быть использованы студентами в будущей профессиональной деятельности.

Методические рекомендации предназначены для студентов, изучающих учебную дисциплину «Основы алгоритмизации и программирования».

Оценки за выполнение практических работ выставляются по пятибалльной системе. Оценки за практические работы являются обязательными текущими оценками по учебной дисциплине и выставляются в журнале теоретического обучения.

## 1. Перечень практических работ по дисциплине «Основы алгоритмизации и программирования»

№ раздела, темы	Освоение умений в процессе занятия	Тема практического занятия	Кол-во часов
Тема 1. Алгоритмизация	<ul style="list-style-type: none"> <li>— разрабатывать алгоритмы для конкретных задач.</li> <li>— использовать программы для графического отображения алгоритмов.</li> <li>— определять сложность работы алгоритмов.</li> </ul>	<b>Практическая работа № 1</b> Построение линейных и разветвляющихся алгоритмов	2
		<b>Практическая работа № 2</b> Построение циклических алгоритмов	2
		<b>Практическая работа № 3</b> Разработка алгоритмов шифрования	2
		<b>Практическая работа № 4</b> Построение алгоритмов различных конструкций	2
Тема 2. Программирование на языке Python	<ul style="list-style-type: none"> <li>— работать в среде программирования.</li> <li>— реализовывать построенные алгоритмы в виде программ на конкретном языке программирования.</li> <li>— оформлять код программы в соответствии со стандартом кодирования.</li> <li>— выполнять проверку, отладку кода программы.</li> </ul>	<b>Практическая работа № 5</b> Использование языка программирования Python для создания программ с линейным алгоритмом	2
		<b>Практическая работа № 6</b> Использование языка программирования Python для создания программ в отдельном файле с использованием строк и операторов отношений	2
		<b>Практическая работа № 7</b> Использование языка программирования Python для создания программ с циклами while и for	2
		<b>Практическая работа № 8</b> Использование языка программирования Python для создания программ по работе с файлами и файловой структурой	2
		<b>Практическая работа № 9</b> Использование языка программирования Python для создания программ с использованием классов	2
		<b>Практическая работа № 10</b> Построение логически правильных и эффективных программ на языке программирования Python	2
		<b>Практическая работа № 11</b> Использование виртуальных окружений для изолирования различных проектов. Полезные функции и модули языка.	2
		<b>Практическая работа № 12</b>	2

		Использование регулярных выражений для проверки конфигурации сетевого оборудования.	
		<b>Практическая работа № 13</b> Чтение и запись данных в форматах CSV, JSON и YAML.	2
		<b>Практическая работа № 14</b> Подключение к оборудованию по SSH и Telnet	2
		<b>Практическая работа № 15</b> Одновременное подключение к нескольким устройствам	2
		<b>Практическая работа № 16</b> Создание шаблонов конфигурации с помощью Jinja2	2
		<b>Практическая работа № 17</b> Обработка вывода команд с помощью TextFSM	2
		<b>Практическая работа № 18</b> Использование объектно-ориентированного программирования для чтения «чужого» кода, кода netmiko.	2
		<b>Практическая работа № 19</b> Использование наследования для создания новых классов на основе существующих.	2
		<b>Практическая работа № 20</b> Работа с базами данных	2

,

## 2. Описание порядка выполнения практических работ

### 2.1. Практическая работа № 1

#### Построение линейных и разветвляющихся алгоритмов

##### Задание:

Вариант № 1

Оформление работы:

- Задание выполняется с использованием стандартного приложения ОС Windows Paint или Microsoft Word. Сохранить под именем Пр1\_№ своего варианта.
- В отчёте к заданию должна быть титульная страница с названием и номером работы.
- Каждое задание в отчёте на отдельной странице.
- Страница должна начинаться с заголовка Задание №\_\_
- Вторая страница в отчёте — оглавление с заголовками Задание №\_\_

Задание:

1. Составить алгоритм вычисления площади боковой поверхности цилиндра ( $S_b = Ph$ ,  $P = 2\pi R$ ), если высота цилиндра 5 см, а радиус 2 см

Составить алгоритм для следующей системы:

$$y = \begin{cases} x^3 - 4x, & \text{если } x \geq 5 \\ x^2 + 5x, & \text{если } 0 \leq x < 5 \\ x - 3, & \text{если } x < 0 \end{cases}$$

Вариант № 2

1. Вычислить площадь боковой поверхности цилиндра ( $S_b = Ph$ ,  $P = 2\pi R$ ), если высота цилиндра 6 см, а радиус 3 см

Составить алгоритм для следующей системы:

$$y = \begin{cases} x^2 + x, & \text{если } x \geq 2 \\ x^4 + 2x, & \text{если } 0 \leq x < 2 \\ x + 5, & \text{если } x < 0 \end{cases}$$

### 2.2. Практическая работа № 2

#### Построение циклических алгоритмов

##### Задание:

Вариант № 1

Оформление работы:

- Задание выполняется с использованием стандартного приложения ОС Windows Paint или Microsoft Word. Сохранить под именем Пр2\_№ своего варианта.
- В отчёте к заданию должна быть титульная страница с названием и номером работы.
- Каждое задание в отчёте на отдельной странице.
- Страница должна начинаться с заголовка Задание №\_\_
- Вторая страница в отчёте — оглавление с заголовками Задание №\_\_

Задание:

1. Составить блок-схему вычисления суммы всех целых нечётных чисел от 1 до 100

2. Составить блок-схему вычисления функции  $y = a / (a + x)$  при  $x$ , изменяющимся от  $x = 1$  до  $x = 4$  с шагом  $Dx = 0,5$
3. Вычислите  $(2-x)^2 + (2-x)^4 + (2-x)^6 + (2-x)^8 + (2-x)^{10}$

### Вариант № 2

Задание:

1. Составить блок-схему вычисления суммы всех целых чётных чисел от 1 до 100
2. Составить блок-схему вычисления функции  $x = b / (b - y)$  при  $y$ , изменяющимся от  $y = 4$  до  $y = 1$  с шагом  $Dy = 0,6$
3. Вычислите  $(2t+3) + (2t+6) + (2t+9) + \dots + (2t+30)$

## 2.3. Практическая работа № 3 Разработка алгоритмов шифрования

Задание 1 Составить алгоритм шифрования текста с помощью шифра Цезаря.

Описание шифра: чтобы зашифровать текст, записанный с помощью русских букв и знаков препинания, его можно переписать, заменив каждую букву непосредственно следующей за ней буквой по алфавиту (буква Я заменяется на А). Обобщив этот способ шифровки, можно производить сдвиг не на одну букву, а на  $N$  букв ( $N$  – натуральное число).

Задание 2 Составить алгоритм шифрования текста с помощью шифра «Матричная шифровка». Описание шифра: чтобы зашифровать текст из 121 буквы, его можно записать в квадратную матрицу порядка 11 по строкам, а затем прочитать по спирали, начиная с центра (т.е. с элемента, имеющего индексы 6,6). Такой способ можно обобщить и для произвольной длины текста, подбирая нужный размер матрицы.

Задание 3 Составить алгоритм шифрования текста с помощью шифра «Шифровка решеткой». Шифровка текста с помощью решетки заключается в следующем. Решетка, т.е. квадрат из клетчатой бумаги  $10 \times 10$  клеток, некоторые клетки в котором вырезаны, совмещается с целым квадратом  $10 \times 10$  клеток, и через прорезы на бумагу наносятся первые буквы текста. Затем решетка поворачивается на  $90$  градусов и через прорезы записываются следующие буквы. Это повторяется еще дважды. Таким образом на бумагу наносятся 100 букв текста. Решетку можно изобразить квадратной матрицей порядка 10 из нулей и единиц (ноль изображает прорезь). Доказано, что матрица  $[A_{ij}]$ ,  $i = 1, \dots, 10$ ,  $j = 1, \dots, 10$  может служить ключом шифра, если из элементов в точности один равен нулю.

Задание 4 Составить алгоритм шифрования текста с помощью шифра «Шифровка зафиксированной перестановкой». Описание шифра. Зафиксируем натуральное  $k$  и перестановку чисел  $1, \dots, k$  (ее можно задать с помощью последовательности натуральных чисел  $r_1, \dots, r_k$ , в которую входит каждое из чисел  $1, \dots, k$ ). При шифровке в исходном тексте  $k$  каждой из последовательных групп по  $k$  символов применяется зафиксированная перестановка. Пусть  $k = 4$  и перестановка есть  $3, 2, 4, 1$  Тогда группа символов  $s_1, s_2, s_3, s_4$  заменяется на  $s_3, s_2, s_4, s_1$ . Если в последней группе меньше четырех символов, то к ней добавляются пробелы.

Задание 5 Составить алгоритм шифрования текста с помощью шифра «Атбаш». Описание шифра. Шифр простой замены, использованный для еврейского алфавита и получивший оттуда свое название. Шифрование происходит заменой первой буквы алфавита на последнюю, второй на предпоследнюю (алеф (первая буква) заменяется на тав (по-

следнюю), бет (вторая) заменяется на шин (предпоследняя); из этих сочетаний шифр и получил свое название).

Шифр Атбаш для английского алфавита:

Исходный алфавит: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Алфавит замены: Z Y X W V U T S R Q P O N M L K J I H G F E D C B A

Задание 6 Составить алгоритм шифрования текста с помощью шифра «Тарабарская грамота».

Суть шифровки, которая использовалась в XV–XVI веках на Руси, в следующем. Все согласные буквы русской азбуки записывались в два ряда; одна половина букв вверху, другая половина – внизу, причем в обратном порядке (одна буква под другой):

Б В Г Д Ж З К Л М Н

Щ Ш Ч Ц Х Ф Т С Р П

При зашифровке слов согласные взаимно заменялись, а гласные, Й и буквы Ъ, Ь вписывались без изменений. Слова записывались без промежутков между ними, как вообще писался любой текст до XVI века, и это еще больше затрудняло разгадывание.

## 2.4. Практическая работа № 4

### Построение алгоритмов различных конструкций

Оформление работы:

- Задание выполняется с использованием стандартного приложения ОС Windows Paint или Microsoft Word. Сохранить под именем Пр3\_№ своего варианта.
- В отчёте к заданию должна быть титульная страница с названием и номером работы.
- Каждое задание в отчёте на отдельной странице.
- Страница должна начинаться с заголовка Задание №\_\_
- Вторая страница в отчёте — оглавление с заголовками Задание №\_\_

**Задание 1.** Создание линейного алгоритма

**Задание 2.** Создание алгоритма ветвления

**Задание 3.** Создание алгоритма цикла

**Задание 4.** Создание алгоритма массива

## 2.5. Практическая работа № 5

### Использование языка программирования Python для создания программ с линейным алгоритмом

**Задание:**

**Вариант № 1**

1. Написать программу на языке Python для вычисления площади боковой поверхности цилиндра ( $S_b = Ph$ ,  $P = 2\pi R$ ), если высота цилиндра 5 см, а радиус 2 см

Написать программу на языке Python для следующей системы:

$$y = \begin{cases} x^3 - 4x, & \text{если } x \geq 5 \\ x^2 + 5x, & \text{если } 0 \leq x < 5 \\ x - 3, & \text{если } x < 0 \end{cases}$$

Вариант № 2

1. Вычислить площадь боковой поверхности цилиндра ( $S_b = Ph$ ,  $P = 2\pi R$ ), если высота цилиндра 6 см, а радиус 3 см

Написать программу для следующей системы:

$$y = \begin{cases} x^2 + x, & \text{если } x \geq 2 \\ x^4 + 2x, & \text{если } 0 \leq x < 2 \\ x + 5, & \text{если } x < 0 \end{cases}$$

Примерный образец выполнения работы:

Практическая работа № 16

Мой вариант № 1

**Задание 1**

Введите значение высоты:

Введите значение радиуса:

Вычислить

**Задание 2**

Введите значение x

Вычислить y

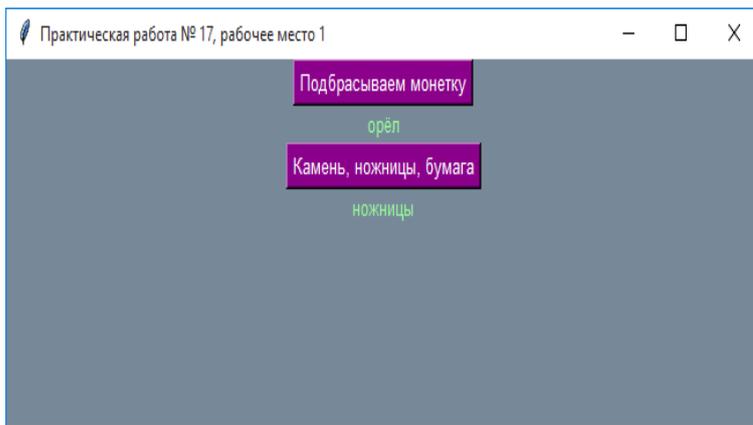
## 2.6. Практическая работа № 6

**Использование языка программирования Python для создания программ в отдельном файле с использованием строк и операторов отношений**

**Задание:**

Написать программу с графическим интерфейсом, используя метки, кнопки, подключить модуль random для генерации случайных значений.

Примерный образец готовой работы:



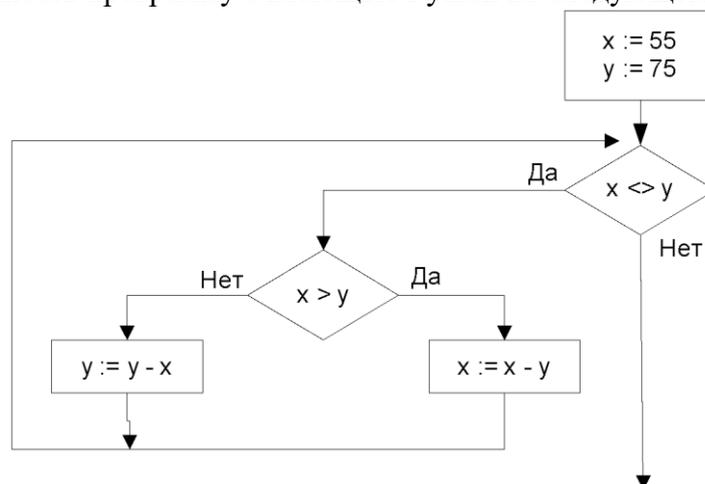
## 2.7. Практическая работа № 7

### Использование языка программирования Python для создания программ с циклами while и for

**Задание:**

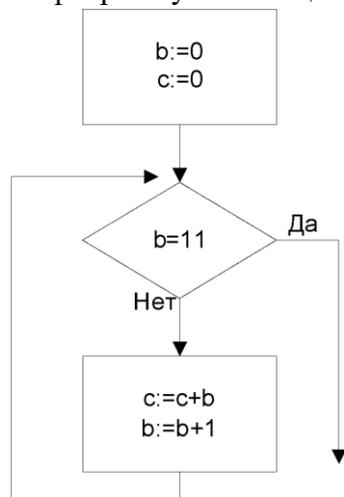
Вариант № 1

1. Написать программу с помощью Python по следующему алгоритму:



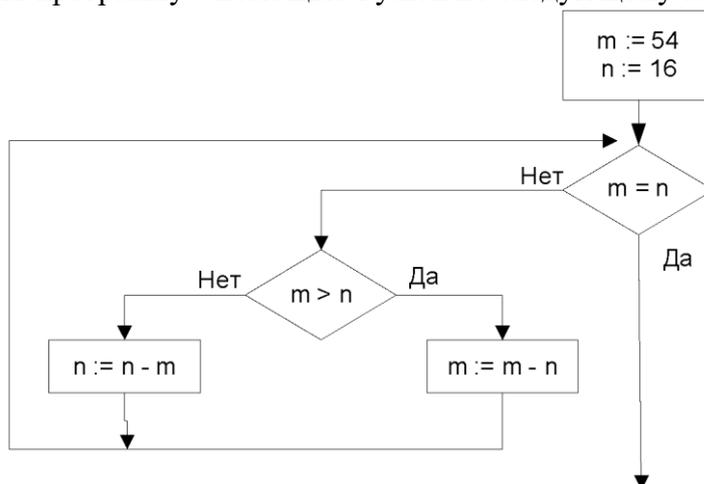
2.

3. Написать программу с помощью Python по следующему алгоритму:

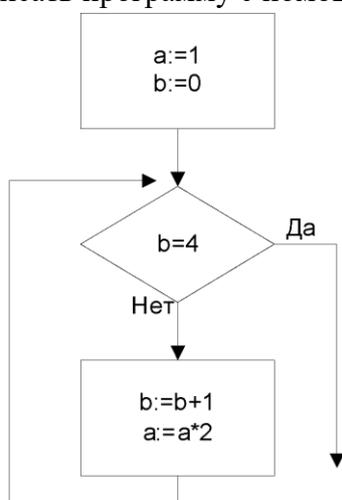


Вариант № 2

1. Написать программу с помощью Python по следующему алгоритму:



2. Написать программу с помощью Python по следующему алгоритму:



## 2.8. Практическая работа № 8

### Использование языка программирования Python для создания программ по работе с файлами и файловой структурой

**Задание:**

1. В своей папке создать файл 1.txt, 2.txt. Сделать несколько строк записей в этих файлах.
2. С помощью модуля os в Python получить список папок/файлов в вашей папке.
3. С помощью модуля os в Python создать папку 1 на диске D.
4. Создать дерево каталогов MY\Python\os на диске D
5. С помощью модуля os в Python переместить файлы 1.txt и 2.txt в папку 1 на диске D и переименовать в my.txt и в my\_1.txt
6. Открыть файл с помощью Python.
7. Все набранные команды в окне программирования на языке Python в виде скриншота вставить в отчет, выполненный в формате docx.

## 2.9. Практическая работа № 9

### Использование языка программирования Python для создания программ с использованием классов

**Задание:**

Создать программу на языке Python по следующему пояснению:

Создайте класс Student. Определите атрибуты name (имя), group (номер группы) и spec (специальность). Добавьте метод под названием study («учит»). Создайте объект класса Murg, установите атрибуты, вызовите метод study.

## 2.10. Практическая работа № 10

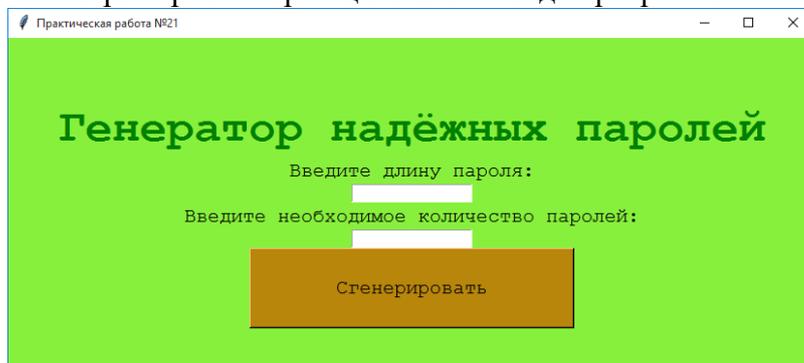
### Построение логически правильных и эффективных программ на языке программирования Python

**Задание:**

Создать программу Генератор паролей с графическим интерфейсом с запросом о количестве паролей и записью в файл всех сгенерированных паролей.

Допустимые символы: заглавные буквы латиницы, маленькие буквы латиницы, знаки \_@#!%&\*

Примерный образец внешнего вида программы:



## 2.11. Практическая работа № 11

### Использование виртуальных окружений для изолирования различных проектов. Полезные функции и модули языка.

**Задание:**

1. Установить Python 3.7 на Debian
2. Задать использование Python 3.7 по умолчанию в виртуальном окружении.
3. Установите с помощью pip virtualenvwrapper

Отредактируйте файл .bashrc, дописав в него:

```
export VIRTUALENVWRAPPER_PYTHON=/usr/local/bin/python3.7
export WORKON_HOME=~/.venv
. /usr/local/bin/virtualenvwrapper.sh
```

4. Перезапустите командный интерпретатор
5. Создайте новое виртуального окружения, в котором Python 3.7 используется по умолчанию
6. Укажите список установленных пакетов в виртуальном окружении

## 2.12. Практическая работа № 12

### Использование регулярных выражений для проверки конфигурации сетевого оборудования.

#### Задание:

Создать функцию `get_ip_from_cfg`, которая ожидает как аргумент имя файла, в котором находится конфигурация устройства.

Функция должна обрабатывать конфигурацию и возвращать IP-адреса и маски, которые настроены на интерфейсах, в виде списка кортежей:

- первый элемент кортежа - IP-адрес
- второй элемент кортежа - маска

Для получения такого результата, используйте регулярные выражения.

Проверить работу функции на примере файла `config_r1.txt`.

Обратите внимание, что в данном случае, можно не проверять корректность IP-адреса, диапазоны адресов и так далее, так как обрабатывается вывод команды, а не ввод пользователя

## 2.13. Практическая работа № 13

### Чтение и запись данных в форматах CSV, JSON и YAML.

#### Задание:

Создать функцию `write_dhcp_snooping_to_csv`, которая обрабатывает вывод команды `show dhcp snooping binding` из разных файлов и записывает обработанные данные в csv файл.

Аргументы функции:

- `filenames` - список с именами файлов с выводом `show dhcp snooping binding`
- `output` - имя файла в формате csv, в который будет записан результат. Функция ничего не возвращает.

Проверить работу функции на содержимом файлов `sw1_dhcp_snooping.txt`, `sw2_dhcp_snooping.txt`, `sw3_dhcp_snooping.txt`. Первый столбец в csv файле имя коммутатора надо получить из имени файла, остальные - из содержимого в файлах.

## 2.14. Практическая работа № 14

### Подключение к оборудованию по SSH и Telnet

#### Задание:

Создать функцию `send_show_command`.

Функция подключается по SSH (с помощью `netmiko`) к ОДНОМУ устройству и выполняет указанную команду.

Параметры функции:

- `device` - словарь с параметрами подключения к устройству
- `command` - команда, которую надо выполнить

Функция возвращает строку с выводом команды.

Скрипт должен отправлять команду `command` на все устройства из файла `devices.yaml` с помощью функции `send_show_command` (эта часть кода написана).

## 2.15. Практическая работа № 15

### Одновременное подключение к нескольким устройствам

#### Задание:

Создать функцию `ping_ip_addresses`, которая проверяет пингуются ли IP-адреса. Проверка IP-адресов должна выполняться параллельно в разных потоках.

Параметры функции:

- `ip_list` - список IP-адресов
- `limit` - максимальное количество параллельных потоков (по умолчанию 3)

Функция должна возвращать кортеж с двумя списками:

- список доступных IP-адресов
- список недоступных IP-адресов

Для выполнения задания можно создавать любые дополнительные функции.

Для проверки доступности IP-адреса, используйте `ping`.

## 2.16. Практическая работа № 16

### Создание шаблонов конфигурации с помощью Jinja2

#### Задание:

Создать функцию `generate_config`.

Параметры функции:

- `template` - путь к файлу с шаблоном (например, «`templates/for.txt`»)
- `data_dict` - словарь со значениями, которые надо подставить в шаблон

Функция должна возвращать строку с конфигурацией, которая была сгенерирована.

Проверить работу функции на шаблоне `templates/for.txt` и данных из файла `data_files/for.yml`.

```
import yaml

# так должен выглядеть вызов функции
if __name__ == "__main__":
    data_file = "data_files/for.yml"
    template_file = "templates/for.txt"
    with open(data_file) as f:
        data = yaml.safe_load(f)
    print(generate_config(template_file, data))
```

## 2.17. Практическая работа № 17

### Обработка вывода команд с помощью TextFSM

#### Задание:

Создать функцию `parse_command_output`. Параметры функции:

- `template` - имя файла, в котором находится шаблон TextFSM (`templates/sh_ip_int_br.template`)
- `command_output` - вывод соответствующей команды `show` (строка)

Функция должна возвращать список:

- первый элемент - это список с названиями столбцов
- остальные элементы это списки, в котором находятся результаты обработки вывода

Проверить работу функции на выводе команды `output/sh_ip_int_br.txt` и шаблоне `templates/sh_ip_int_br.template`.

```
from netmiko import ConnectHandler

# вызов функции должен выглядеть так
if __name__ == "__main__":
    r1_params = {
        "device_type": "cisco_ios",
        "host": "192.168.100.1",
        "username": "cisco",
        "password": "cisco",
        "secret": "cisco",
    }
    with ConnectHandler(**r1_params) as r1:
        r1.enable()
        output = r1.send_command("sh ip int br")
        result = parse_command_output("templates/sh_ip_int_br.template", output)
        print(result)
```

## 2.18. Практическая работа № 18

### Использование объектно-ориентированного программирования для чтения «чужого» кода, кода `netmiko`.

#### Задание:

Создать класс `MyNetmiko`, который наследует класс `CiscoIosSSH` из `netmiko`.

Переписать метод `__init__` в классе `MyNetmiko` таким образом, чтобы после подключения по

SSH выполнялся переход в режим `enable`.

Для этого в методе `__init__` должен сначала вызываться метод `__init__` класса `CiscoIosBase`, а

затем выполнялся переход в режим `enable`.

Проверить, что в классе `MyNetmiko` доступны методы `send_command` и `send_config_set`

## 2.19. Практическая работа № 19

### Использование наследования для создания новых классов на основе существующих.

#### Задание:

Создать класс `CiscoSSH`, который наследует класс `BaseSSH` из файла `base_connect_class.py`.

Создать метод `__init__` в классе `CiscoSSH` таким образом, чтобы после подключения по SSH выполнялся переход в режим `enable`.  
 Для этого в методе `__init__` должен сначала вызываться метод `__init__` класса `ConnectSSH`, а затем выполняться переход в режим `enable`

```
In [2]: from task_24_1 import CiscoSSH

In [3]: r1 = CiscoSSH(**device_params)

In [4]: r1.send_show_command('sh ip int br')
Out[4]: 'Interface                               IP-Address      OK? Method Status
↪ Protocol\nEthernet0/0                          192.168.100.1   YES NVRAM  up
↪      up      \nEthernet0/1                          192.168.200.1   YES NVRAM  up
↪      up      \nEthernet0/2                          190.16.200.1   YES NVRAM
↪up      up      \nEthernet0/3                          192.168.230.1   YES
↪NVRAM  up      up      \nEthernet0/3.100                    10.100.0.1
↪ YES NVRAM  up      up      \nEthernet0/3.200                    10.200.
↪0.1      YES NVRAM  up      up      \nEthernet0/3.300
↪10.30.0.1      YES NVRAM  up      up      '
```

## 2.20. Практическая работа № 20 Работа с базами данных

### Задание:

Необходимо создать два скрипта:

1. `create_db.py`
2. `add_data.py`

Код в скриптах должен быть разбит на функции. Какие именно функции и как разделить код,

надо решить самостоятельно. Часть кода может быть глобальной.

1. `create_db.py` - в этот скрипт должна быть вынесена функциональность по созданию БД:

- должна выполняться проверка наличия файла БД
- если файла нет, согласно описанию схемы БД в файле `dhcp_snooping_schema.sql`, должна быть создана БД
- имя файла бд - `dhcp_snooping.db`

В БД должно быть две таблицы (схема описана в файле `dhcp_snooping_schema.sql`):

- `switches` - в ней находятся данные о коммутаторах
- `dhcp` - тут хранится информация полученная из вывода `sh ip dhcp snooping binding`

Пример выполнения скрипта, когда файла `dhcp_snooping.db` нет:

```
$ python create_db.py
```

Создаю базу данных...

После создания файла:

```
$ python create_db.py
```

База данных существует

2. `add_data.py` - с помощью этого скрипта, выполняется добавление данных в БД. Скрипт должен добавлять данные из вывода `sh ip dhcp snooping binding` и информацию о коммутаторах Соответственно, в файле `add_data.py` должны быть две части:

- информация о коммутаторах добавляется в таблицу switches
    - данные о коммутаторах, находятся в файле switches.yml
  - информация на основании вывода `sh ip dhcp snooping binding` добавляется в таблицу dhcp
    - вывод с трёх коммутаторов: файлы `sw1_dhcp_snooping.txt`, `sw2_dhcp_snooping.txt`, `sw3_dhcp_snooping.txt`
    - так как таблица dhcp изменилась, и в ней теперь присутствует поле switch, его нужно также заполнять. Имя коммутатора определяется по имени файла с данными
- Пример выполнения скрипта, когда база данных еще не создана:  
\$ python add\_data.py  
База данных не существует. Перед добавлением данных, ее надо создать
- Пример выполнения скрипта первый раз, после создания базы данных:  
\$ python add\_data.py  
Добавляю данные в таблицу switches...  
Добавляю данные в таблицу dhcp...