Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Санкт-Петербургский государственный университет промышленных технологий и дизайна»

|  |
| --- |
| **Институт дополнительного профессионального образования** |

|  |  |
| --- | --- |
|  |  |
| Направление подготовки: | 38.03.01 экономика |
| Профиль подготовки: | Международный бизнес |

|  |
| --- |
| **ОТЧЕТ** |
| о прохождении практики |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Руководитель  от профильной организации / структурного подразделения СПбГУПТД\*: |  | Бахов Герман Геннадьевич | | | | |  |  |
| *(наименование организации)* |  | *(должность, Ф.И.О., печать организации)* | | | | |  | *(подпись, печать)* |
| Руководитель  от СПбГУПТД: |  | Зубов Александр Михайлович | | | | |  |  |
|  |  | *(должность, ученая степень / звание, Ф.И.О.)* | | | | |  | *(подпись)* |
| Обучающийся: |  | Елеусизова Азиза Маликовна | | | | |  |  |
|  |  | *(Ф.И.О.)* | | | | |  | *(подпись)* |
| Курс |  | 2 |  | Учебная группа: |  | 2-ЭД-4 | |

Санкт-Петербург

2023

**Реферат**

Отчет содержит страниц 15 и 1 таблицу. Список использованных источников содержит 5 наименований.

**PYTHON, PYCHARM, ПАРОЛЬ, КОД, ПРОГРАММИРОВАНИЕ, ГЕНЕРАТОР.**

Цель работы – научиться создавать программный код на языке python.

Задачи исследования:

- понять как работает библиотека tkinter;

- научиться использовать на практике программные комбинации;

- создать логически связанный код.

**СОДЕРЖАНИЕ**

[Глава 1. Теоретические аспекты решаемой задачи. Описание предметной области 5](#_Toc133523483)

[1.1.Знакомство с tkinter и осоновными операторами решаемой задачи 5](#_Toc133523484)

[1.2.Последовательность написания программы 9](#_Toc133523485)

[Глава 2. Решение задачи 10](#_Toc133523486)

[2.1. Продумываем и описываем алгоритм программы 10](#_Toc133523487)

[2.2. Алгоритмизация 10](#_Toc133523488)

[2.3. Создание самого алгоритма кода 11](#_Toc133523489)

[2.4. Код программы 13](#_Toc133523490)

[Заключение 14](#_Toc133523491)

[Список использованных источников 15](#_Toc133523492)

**Ведение**

**Генератор паролей** - это онлайн-инструмент, позволяющий генерировать сложные и надежные пароли онлайн. Данные пароли досточно хорошо защищены от подбора злоумышленниками. Они генерируются случайным образом и нигде не хранятся. Для генерации пароля введите его длину и нажмите "Сгенерировать". После этого скопируйте безопасный пароль и используйте его в любой Вашей информационной системе, например, при регистрации на каком-либо сайте, в почтовом ящике, учетной записи операционной системы и т.п.

Например, в последнее время участились случаи взлома сайтов, использующих слабые пароли. Чаще всего, это приводит к плачевным последствиям для работы самого сайта. После взлома злоумышленники обычно заражают сайт, поисковые системы отслеживают данную ситуацию и исключают сайт из поисковой выдачи, со временем сайт может выпасть из индексов поисковых систем.

Цель работы – научиться создавать программный код на языке python.

Задачи исследования:

- понять как работает библиотека tkinter;

- научиться использовать на практике программные комбинации;

- создать логически связанный код.

# **Глава 1. Теоретические аспекты решаемой задачи. Описание предметной области**

## **1.1.Знакомство с tkinter и осоновными операторами решаемой задачи**

Tkinter - это стандартная библиотека Python, которая предоставляет графический интерфейс пользователя (GUI) для создания приложений с помощью инструментов Tk. Tkinter включает в себя виджеты для создания окон, кнопок, текстовых полей, меню, списков и других элементов интерфейса.

Tkinter позволяет создавать кроссплатформенные приложения с графическим интерфейсом, которые могут работать на большинстве операционных систем, таких как Windows, macOS и Linux. Tkinter также имеет простой и понятный синтаксис, что делает его доступным для начинающих разработчиков.

Некоторые из основных преимуществ Tkinter включают в себя:

- Простота использования и понятный синтаксис

- Кроссплатформенность

- Богатый набор виджетов для создания интерфейса

- Гибкость и возможность настройки

Tkinter широко используется в различных областях, таких как разработка игр, приложений для научных исследований, бизнес-приложений и многих других. Если вы хотите создать приложение с графическим интерфейсом на Python, Tkinter - это один из лучших выборов.

**Практические преимущества tkinter**

Но как бы то ни было, существуют более прагматичные причины, объясняющие, почему библиотека tkinter де-факто стала в мире Python стандартом разработки переносимых графических интерфейсов. Такие преимущества, как доступность, переносимость, простота получения, документированность и наличие расширений, делают ее наиболее широко используемым решением для Python в области GUI в течение многих лет:

*Доступность*

Библиотека tkinter в целом рассматривается как *облегченный набор инструментов* и одно из наиболее простых решений в области разработки графических интерфейсов для Python из имеющихся на сегодняшний день. В отличие от более крупных фреймворков, библиотека tkinter позволяет сразу же приступить к работе с ней, без необходимости предварительно осваивать значительно более крупные модели взаимодействия классов. Как будет показано далее, программист может создать с помощью tkinter простой интерфейс, написав всего несколько строк программного кода, и постепенно наращивать его возможности до достижения промышленного уровня. Несмотря на всю простоту прикладного интерфейса библиотеки tkinter, она позволяет добавлять новые виджеты, написанные на языке Python, или подключать дополнительные расширения, такие как Pmw, Tix и ttk.

*Переносимость*

Сценарий на языке Python, в котором графический интерфейс строится с помощью библиотеки tkinter, будет работать без изменений на всех основных современных оконных платформах: Microsoft Windows, X Window (в Unix и Linux) и Macintosh OS X (а также в классической версии Mac). Более того, этот сценарий создаст интерфейс, внешний вид которого будет привычен пользователям каждой из этих платформ. Эта особенность развивалась по мере того как библиотека Tk становилась все более зрелой. Графический интерфейс, реализованный сценарием Python/tkinter, в Windows выглядит, как должен выглядеть интерфейс программы для Windows; в Unix и Linux обеспечивает такое же взаимодействие с пользователем, но демонстрирует внешний вид, знакомый пользователям X Window; и на Mac он выглядит так, как должна выглядеть программа Mac.

*Простота получения*

tkinter является модулем стандартной библиотеки Python, поставляемой вместе с интерпретатором. Если Python установлен на вашем компьютере, то у вас есть доступ и к библиотеке tkinter. Более того, в большинство пакетов установки Python (включая стандартный пакет установки Python для Windows, пакет установки для Mac и пакеты установки для большинства дистрибутивов Linux) уже включе- на поддержка tkinter. Благодаря этому сценарии, написанные с использованием модуля tkinter, сразу могут работать с большинством интерпретаторов Python, не требуя дополнительных действий по установке. Библиотека tkinter также в целом лучше поддерживается, чем существующие сегодня альтернативные пакеты. Поскольку задействованная в ней библиотека Tk используется также языками программирования Tcl и Perl (и многими другими), ей уделяется больше внимания и усилий разработчиков, чем другим имеющимся инструментариям.

Ключевым строительным блоком в графическом приложении являются различные элементов управления, с которыми взаимодействует пользователь, как кнопки, метки, поля ввода. В Tkinter имеется богатая палитра различных элементов управления, которые называются виджетами. Основные из них:

* Button: кнопка
* Label: текстовая метка
* Entry: однострочное текстовое поле
* Text: многострочное текстовое поле
* Checkbutton: флажок
* Radiobutton: переключатель или радиокнопка
* Frame: фрейм, который организует виджеты в группы
* Listbox: список
* Combobox: выпадающий список
* Menu: элемент меню
* Scrollbar: полоса прокрутки
* Treeview: позволяет создавать древовидные и табличные элементы
* Scale: текстовая метка
* Spinbox: список значений со стрелками для перемещения по элементам
* Progressbar: текстовая метка
* Canvas: текстовая метка
* Notebook: панель вкладок

Tkinter предоставляет виджеты в двух вариантах: виджеты, которые располагаются непосредственно в пакете tkinter, и виджеты из пакета tkinter.ttk. С одной стороны, оба пакета предоставляют практически одни и те же виджеты, например, виджет Button есть в обоих пакетах. Но с другой стороны, ttk предоставляет чуть больше функциональности по настройке виджетов, в частности, по их стилизации. И считается, что виджеты из ttk несколько современнее, чем стандартные, в то же время с ttk, возможно, чуть сложнее работать. Что именно использовать остается на выбор разработчика.

Таблица 1 – операторы Python

|  |  |
| --- | --- |
| Функция | Пояснение |
| init() | Инициализирует все модули Pygame (возвращает кортеж в случае успеха или неудачи). |
| isplay.set\_mode() | Для создания поверхности принимает в качестве параметра либо список, либо кортеж (кортеж предпочтительней). |
| update() | Обновляет экран. |
| quit() | Используется для деинициализации всех модулей. |
| set\_caption() | Устанавливает текст заголовка в верхней части экрана |
| event.get() | Возвращает список всех событий. |
| Surface.fill() | Заполняет пространство сплошным цветом. |
| time.Clock() | Отслеживание времени |
| font.SysFont() | Задает шрифт Pygame, используя системные ресурсы. |

## **1.2.Последовательность написания программы**

Для создания программ с графическим интерфейсом вам необходимо выполнить следующие этапы:

1. Импорт библиотеки
2. Создание главного окна
3. Создание виджет
4. Установка их свойств
5. Определение событий
6. Определение обработчиков событий
7. Расположение виджет на главном окне
8. Отображение главного окна.

# **Глава 2. Решение задачи**

## **2.1. Продумываем и описываем алгоритм программы**

При создании кода на Python для генерации сложных паролей можно использовать следующий алгоритм:

1. Определить длину пароля, который нужно сгенерировать.

2. Создать список символов, которые могут быть использованы в пароле. Это могут быть буквы (как заглавные, так и строчные), цифры и специальные символы.

3. Случайным образом выбирать символы из списка и добавлять их к паролю до тех пор, пока не достигнута нужная длина пароля.

4. Перемешать символы в пароле, чтобы сделать его более надежным.

5. Вывести сгенерированный пароль.

## **2.2. Алгоритмизация**

Для начало работы нужно скачать программу PyCharm. Создание графического пользовательского интерфейса (GUI) на Python может быть достигнуто с использованием стандартной библиотеки Tkinter. Tkinter предоставляет набор виджетов для создания окон, кнопок, текстовых полей, меню и других элементов интерфейса. Вот несколько этапов, которые необходимо выполнить для создания GUI на Python:

1. Импортирование библиотеки Tkinter: Для использования Tkinter в Python, сначала необходимо импортировать библиотеку. Это можно сделать, написав следующую строку кода в начале программы:

from tkinter import \*

2. Создание окна: Для создания окна на Tkinter нужно создать экземпляр класса Tk() и назначить его переменной, например так:

root = Tk()

3. Добавление виджетов: В Tkinter есть множество виджетов, которые можно добавить на окно. Например, для добавления кнопки на окно, можно использовать следующий код:

button = Button(root, text="Нажми меня")

button.pack()

4. Настройка размещения виджетов: Для управления размещением виджетов на окне можно использовать менеджер геометрии. Например, для размещения кнопки по центру окна можно использовать следующий код:

button.pack(side="bottom", fill="both", expand=True)

5. Обработка событий: Для обработки событий, таких как нажатие кнопки или изменение текста в текстовом поле, можно использовать функции обратного вызова. Например, для создания функции, которая будет вызываться при нажатии кнопки, можно использовать следующий код:

def button\_click():

print("Кнопка нажата!")

button = Button(root, text="Нажми меня", command=button\_click)

button.pack()

6. Запуск приложения: После создания окна и добавления виджетов, необходимо запустить приложение. Это можно сделать, вызвав метод mainloop() объекта Tk().

root.mainloop()

Это лишь общий обзор этапов создания графического пользовательского интерфейса на Python с использованием библиотеки Tkinter. Однако, если вы захотите создать более сложный интерфейс, вам может потребоваться использовать дополнительные инструменты и библиотеки.

## **2.3. Создание самого алгоритма кода**

Создание генератора сложных паролей на языке Python может быть достигнуто с использованием генераторов случайных чисел и строковых операций. Вот несколько этапов, которые необходимо выполнить для создания кода генератора сложных паролей на Python:

1. Импортирование библиотеки random: Для использования генератора случайных чисел в Python, необходимо импортировать библиотеку random. Это можно сделать, написав следующую строку кода в начале программы:

import random

2. Определение списка символов: Для генерации пароля необходимо определить список символов, которые будут использоваться в пароле. Например, список может содержать символы в верхнем и нижнем регистре, цифры и специальные символы. Это можно сделать, написав следующую строку кода:

symbols = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890!@#$%^&\*()"

3. Определение длины пароля: Для генерации пароля необходимо определить длину пароля. Например, можно определить, что длина пароля будет 10 символов. Это можно сделать, написав следующую строку кода:

length = 10

4. Генерация пароля: Для генерации пароля необходимо использовать генератор случайных чисел и операции со строками. Например, вы можете создать цикл, который будет выбирать случайный символ из списка символов и добавлять его в пароль нужное количество раз. Это можно сделать, написав следующую строку кода:

password = "".join(random.choice(symbols) for i in range(length))

5. Вывод пароля: После генерации пароля необходимо вывести его на экран. Это можно сделать, написав следующую строку кода:

print("Сгенерированный пароль: ", password)

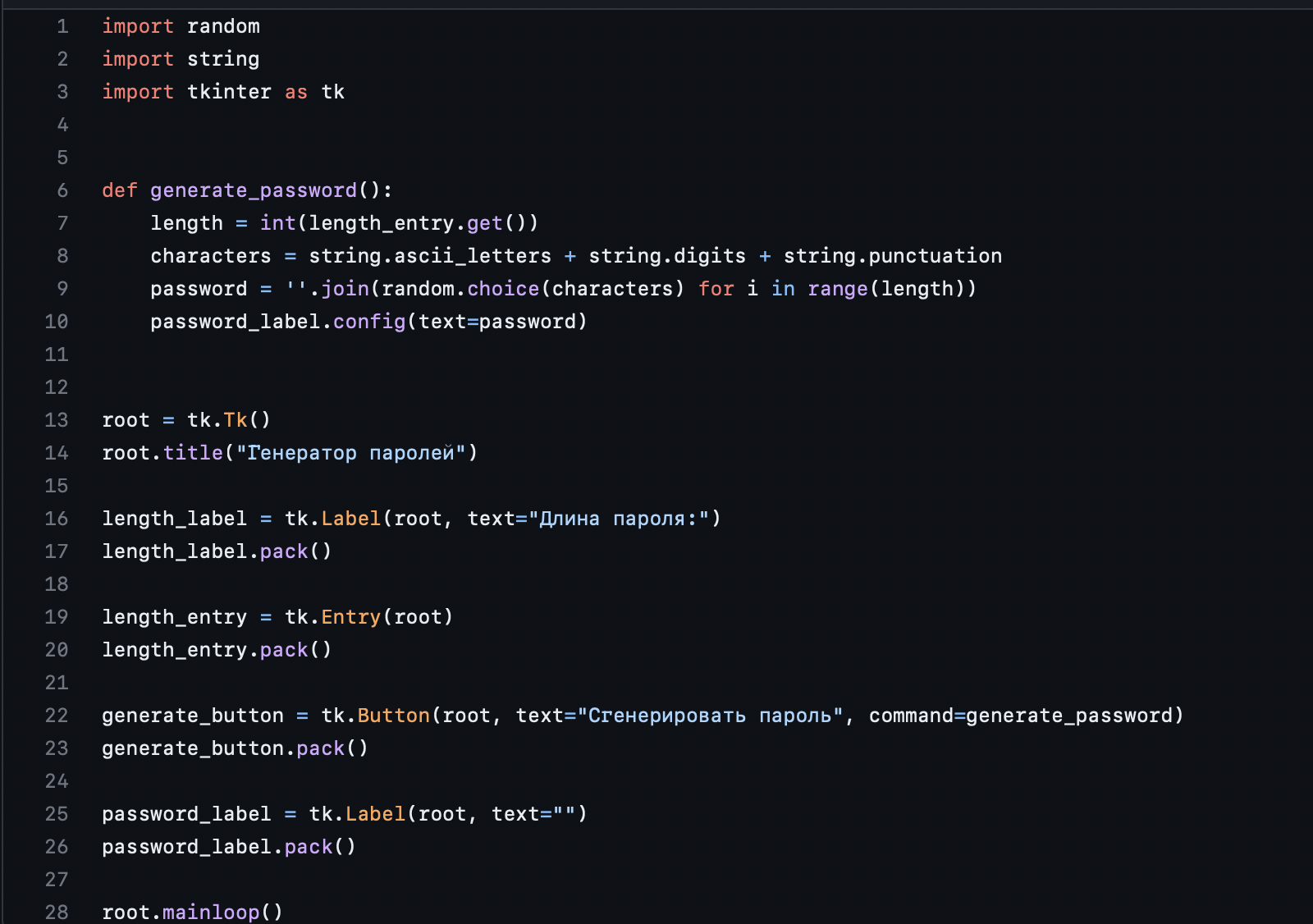
6. Дополнительные опции: Для улучшения генератора паролей можно добавить дополнительные опции, такие как выбор длины пароля, выбор списка символов и т.д. Например, вы можете создать функцию, которая будет принимать на вход длину пароля и список символов, и генерировать пароль на основе этих параметров. Это можно сделать, написав следующую функцию:

def generate\_password(length, symbols):

return "".join(random.choice(symbols) for i in range(length))

Таким образом, создание генератора сложных паролей на языке Python не так сложно, как может показаться на первый взгляд. С помощью генераторов случайных чисел и операций со строками можно легко создать качественный генератор паролей, который будет обеспечивать безопасность ваших учетных записей.

## **2.4. Код программы**

****

# **Заключение**

В этом отчете был представлен алгоритм генерации сложных паролей на языке программирования Python. Алгоритм состоит из нескольких шагов, включая определение длины пароля, создание списка символов, которые могут быть использованы в пароле, и случайный выбор символов из этого списка. После этого символы перемешиваются, чтобы создать случайный порядок символов в пароле, и пароль выводится на экран.

В результате был представлен пример кода на Python для генерации безопасных паролей. Этот код может быть использован как основа для создания более сложных и безопасных паролей. Кроме того, в отчете были представлены рекомендации по созданию безопасных паролей, такие как использование длинных паролей, содержащих различные типы символов, избегание использования персональных данных в качестве пароля, и использование менеджеров паролей.

Таким образом, генератор сложных паролей на языке программирования Python является полезным инструментом для создания безопасных паролей. При этом не стоит забывать о том, что безопасность пароля также зависит от того, как мы его используем и храним. Поэтому, помимо генерации безопасных паролей, важно соблюдать правила безопасности в интернете и использовать надежные методы хранения паролей.

# **Список использованных источников**

1. Гуриков, С.Р. Основы алгоритмизации и программирования на Python / С.Р. Гуриков. - М.: Форум, 2018. - **991** c.  
   Гуриков, С.Р. Основы алгоритмизации и программирования на Python. Учебное пособие. Гриф МО РФ / С.Р. Гуриков. - М.: Инфра-М, Форум, 2018. - **707** c.
2. Инструкция и выражения для начинающих // PythonRU 2018-2021гг // (электронный ресурс). Режим доступа : <https://pythonru.com/uroki/instrukcii-i-vyrazhenija-dlja-nachinajushhih?ysclid=lgntpj92r864883064>
3. Краткий курс ООП на Python: как избежать путаницы в коде // skiilbox media 2023г // (электронный ресурс). Режим доступа : https://skillbox.ru/media/code/kak-izbezhat-putanitsy-v-kode-ili-kratkiy-kurs-oop-na-python/?ysclid=lgnu06ihh6659998908
4. Метод Time clock() в Python – подробно на примерах // Помощник Python 2022г // (электронный ресурс). Режим доступа : <https://pythonpip.ru/osnovy/metod-time-clock-v-python-podrobno-na-primerah>
5. Широков, А. И. Алгоритмизация и программирование на языке «Питон» (Python) : методические указания / А. И. Широков. - Москва, 2021. - 48 с. - Текст : электронный. - URL: https://znanium.com/catalog/product/1915590