Изображение выглядит как Графика, графический дизайн, графическая вставка, творческий подход

Автоматически созданное описание

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГАОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт**  **информационных**  **технологий** | **Кафедра**  **управления и информатики**  **в технических системах** |

**ЛАБОРАТОРНАЯ РАБОТА №2**  
ПО ДИСЦИПЛИНЕ

«ПРИКЛАДНОЕ ПРОГРАММИРОВАНИЕ»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТА | 2 | КУРСА | | БАКАЛАВРИАТА | ГРУППЫ | ИДБ-24-14 |
|  | | | *(уровень профессионального образования)* | |  | |

|  |
| --- |
| **ДРОБЯЗКИНА КОНСТАНТИНА АЛЕКСЕЕВИЧА** |
| *(ФИО)* |

ТЕМА РАБОТЫ

|  |
| --- |
| «Регулярные выражения. Unit-тестирование» |

|  |  |
| --- | --- |
| Направление: | 09.03.03 Прикладная информатика |

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет сдан «\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г. | | | |
|  |  |  |  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | |
|  | | | |
| Преподаватель | Малова Яна Игоревна |  |  |
|  | *(Ф.И.О., должность, степень, звание.)* |  | *(подпись)* |

МОСКВА 2025

Содержание

Оглавление

[Описание работы 2](#_Toc209953379)

[UML-диаграмма 3](#_Toc209953380)

[Программная реализация 4](#_Toc209953381)

[Листинг программы 4](#_Toc209953382)

[Обработка исключений 4](#_Toc209953383)

[Сериализация и десериализация данных 5](#_Toc209953384)

[Вывод 6](#_Toc209953385)

[Ссылка на удаленный репозиторий 7](#_Toc209953386)

# Описание работы

**Цель работы**: изучить возможности регулярных выражений в Python, познакомиться с unit-тестированием.

**Вариант**: 15. Проверка и поиск двоичных записей чисел, кратных 3.

**Технологии**: python, формат данных TXT.

**Инструменты**: VSCode, GitHub.

**Задача**:

1. Составить регулярное выражение по своему варианту.
2. Реализовать по спроектированному регулярному рабочую программу с проверкой файла txt, пользовательским вводом и поиском по веб-странице.
3. Написать unit-тесты, уровень покрытия высокий.
4. Подготовить отчет о выполнении лабораторной работы

# Программная реализация

## Листинг программы

|  |
| --- |
| Листинг 1. Текст binary\_divisible\_by\_three.py |
| import re  def is\_divisible\_by\_three(binary\_string: str) -> bool:      """      Паттерн выглядит примерно как      Либо 0      Либо начинается с 1 и заканчивается на 1 и между ними есть или может не быть 01\*0         01\*0 значит что между 0 0 может быть любое количество 1, в том числе и не быть вообще      И таких вот комбинаций 1 или больше раз      """      # Регулярное выражение, которое соответствует числам, кратным 3      # (0|1(01\*0)\*1)\* - это регулярка для чисел, кратных 3      pattern = r'^(0|1(01\*0)\*1)+$'        if not re.match(r'^[01]+$', binary\_string):          raise ValueError("Строка должна содержать только символы 0 и 1")        return bool(re.fullmatch(pattern, binary\_string))  if \_\_name\_\_ == "\_\_main\_\_":        # Примеры использования      test\_cases = [          "0",          "1",          "11",          "100",          "101",          "110",          "1001",          "1010",          "1100",          "1111",          "10010",          "110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010101101101111101101101101101110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010101101101111101101101101101110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010101101101111101101101101101110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010110101111111010101101101111101101101101101"      ]        print("\nПроверка бинарных чисел на кратность 3:")      print("=" \* 60)      b = test\_cases[-1]      print(f"Число {b} в десятичной системе: {int(str(b), 2)} кратно 3: {is\_divisible\_by\_three(str(b))}")      # a = 12000      # print(f"Число {bin(a)[2:]} в десятичной системе: {a} кратно 3: {is\_divisible\_by\_three(bin(a)[2:])}")      print("") |

|  |
| --- |
| Листинг 2. Текст binary\_divisible\_by\_three.py |
| import re  import urllib.request  import urllib.error  from binary\_divisible\_by\_three import is\_divisible\_by\_three  def load\_and\_check\_from\_file(filename: str):      # Загружает двоичные числа из файла и проверяет их на кратность 3.      try:          with open(filename, "r", encoding="utf-8") as f:              lines = f.readlines()          results = []          for i, line in enumerate(lines, 1):              binary = line.strip()              if binary:                  try:                      result = is\_divisible\_by\_three(binary)                      decimal = int(binary, 2)                      results.append(                          {                              "line": i,                              "binary": binary,                              "decimal": decimal,                              "divisible": result,                          }                      )                  except ValueError as e:                      print(f"Строка {i}: {binary} - Ошибка: {e}")          return results      except FileNotFoundError:          print(f"Файл {filename} не найден.")          return []  def user\_input\_check():      # Позволяет пользователю вводить одно двоичное число для проверки.      binary = input("\nВведите двоичное число: ").strip()      if not binary:          print("Пустая строка.")          return      try:          result = is\_divisible\_by\_three(binary)          decimal = int(binary, 2)          print(              f"  {binary} (десятичное: {decimal}) {'делится на 3' if result else 'не делится на 3'}"          )      except ValueError as e:          print(f"  Ошибка: {e}")  def print\_results(results):      # Выводит результаты проверки бинарных чисел.      if not results:          print("Нет результатов для вывода.")          return      print("\nРезультаты проверки:")      print("-" \* 60)      for r in results:          status = "да" if r["divisible"] else "нет"          print(              f"Строка {r['line']}: {r['binary']} = {r['decimal']} - делится на 3: {status}"          )      print()  def find\_binary\_numbers(text: str):      # Извлекает двоичные числа из текста.      # Ищем числа длиной от 3 символов, где начинается с 1      pattern = r"\b1[01]{2,}\b"      return re.findall(pattern, text)  def analyze\_url\_content(url: str = "https://www.convertbinary.com/numbers/"):      # Открывает URL, находит все двоичные числа и проверяет их на кратность 3.      try:          with urllib.request.urlopen(url, timeout=10) as response:              encoding = response.headers.get\_content\_charset() or "utf-8"              html = response.read().decode(encoding)              binary\_numbers = find\_binary\_numbers(html)              results = []              for binary in binary\_numbers:                  try:                      result = is\_divisible\_by\_three(binary)                      decimal = int(binary, 2)                      results.append(                          {"binary": binary, "decimal": decimal, "divisible": result}                      )                  except ValueError:                      # Пропускаем некорректные числа                      continue              return results      except urllib.error.URLError as e:          print(f"Проблема открытия {url}: {e}")          return []      except Exception as e:          print(f"Произошла ошибка: {e}")          return []  def main():      # Главное меню программы.      while True:          print("\nМеню:")          print("1 - Проверка из файла data.txt")          print("2 - Пользовательский ввод")          print("3 - Поиск двоичных чисел на веб-странице")          print("0 - Выход")          choice = input("\nВыберите пункт меню: ").strip()          if choice == "1":              results = load\_and\_check\_from\_file("data.txt")              print\_results(results)          elif choice == "2":              user\_input\_check()          elif choice == "3":              url = input("Введите URL: ").strip()              if url:                  print("\nОбработка...")                  results = analyze\_url\_content(url)                  countLimit = 50                  if results:                      divisible\_count = sum(                          1 for r in results if r["divisible"]                      )  # if r["divisible"]                      print(f"\nНайдено двоичных чисел: {len(results)}")                      print(f"Кратных 3: {divisible\_count}")                      print("\nПримеры чисел, кратных 3:")                      examples = [r for r in results if r["divisible"]][                          :countLimit                      ]  # if r["divisible"]                      for r in examples:                          print(f"  {r['binary']} = {r['decimal']} {r['divisible']}")                      if divisible\_count > countLimit:                          print(f"  ... и еще {divisible\_count - countLimit} чисел")                  else:                      print("\nДвоичные числа не найдены.")          elif choice == "0":              print("Выход из программы.")              break          else:              print("Неверный выбор. Попробуйте снова.")  if \_\_name\_\_ == "\_\_main\_\_":      main() |

## Unit-тестирование

|  |
| --- |
| Листинг 3. Текст test\_main.py |
| import unittest  from binary\_divisible\_by\_three import is\_divisible\_by\_three  class TestBinaryDivisibleByThree(unittest.TestCase):      def test\_divisible\_zero(self):          # Тестирование числа 0 (делится на 3)          self.assertTrue(is\_divisible\_by\_three("0"))      def test\_not\_divisible\_one(self):          # Тестирование числа 1 (не делится на 3)          with self.assertRaises(ValueError):              is\_divisible\_by\_three("1")      def test\_divisible\_three(self):          # Тестирование числа 3 (11 в двоичной, делится на 3)          self.assertTrue(is\_divisible\_by\_three("11"))      def test\_divisible\_six(self):          # Тестирование числа 6 (110 в двоичной, делится на 3)          self.assertTrue(is\_divisible\_by\_three("110"))      def test\_divisible\_nine(self):          # Тестирование числа 9 (1001 в двоичной, делится на 3)          self.assertTrue(is\_divisible\_by\_three("1001"))      def test\_divisible\_twelve(self):          # Тестирование числа 12 (1100 в двоичной, делится на 3)          self.assertTrue(is\_divisible\_by\_three("1100"))      def test\_divisible\_fifteen(self):          # Тестирование числа 15 (1111 в двоичной, делится на 3)          self.assertTrue(is\_divisible\_by\_three("1111"))      def test\_not\_divisible\_two(self):          # Тестирование числа 2 (10 в двоичной, не делится на 3)          with self.assertRaises(ValueError):              is\_divisible\_by\_three("10")      def test\_not\_divisible\_five(self):          # Тестирование числа 5 (101 в двоичной, не делится на 3)          with self.assertRaises(ValueError):              is\_divisible\_by\_three("101")      def test\_not\_divisible\_ten(self):          # Тестирование числа 10 (1010 в двоичной, не делится на 3)          with self.assertRaises(ValueError):              is\_divisible\_by\_three("1010")      def test\_invalid\_characters(self):          # Тестирование недопустимых символов          with self.assertRaises(ValueError):              is\_divisible\_by\_three("123")          with self.assertRaises(ValueError):              is\_divisible\_by\_three("abc")          with self.assertRaises(ValueError):              is\_divisible\_by\_three("102")      def test\_empty\_string(self):          # Тестирование пустой строки          with self.assertRaises(ValueError):              is\_divisible\_by\_three("")      def test\_ok\_large\_number(self):          # Тестирование большого числа, кратного 3          large\_binary = "110"          self.assertTrue(is\_divisible\_by\_three(large\_binary))  if \_\_name\_\_ == "\_\_main\_\_":      unittest.main() |

# Данные для тестирования

|  |
| --- |
| Файл данных 1. Текст data.txt |
| 0  1  11  100  101  110  1001  1010  1100  1111  10010  11011  111001000010111011110010000101110111011 |

<https://www.convertbinary.com/numbers/> - ссылка на веб страницу с двоичными числами

# Вывод

В лабораторной работе мы познакомились с регулярными выражениями и реализовали на python проверку на кратность 3 среди двоичных записей чисел. Также познакомились с unit-тестированием и написали класс для его задач.

# Ссылка на удаленный репозиторий

https://github.com/67Konstantin/labs\_app\_programming/tree/lab2/lab2