|  |  |  |  |
| --- | --- | --- | --- |
| Obraz zawierający symbol, design  Opis wygenerowany automatycznie | Politechnika Bydgoska im. J. J. Śniadeckich  **Wydział Telekomunikacji,**  **Informatyki i Elektrotechniki** | | Obraz zawierający symbol, logo, Grafika, Czcionka  Opis wygenerowany automatycznie |
| **Przedmiot** | Skryptowe języki programowania | | |
| **Prowadzący** | mgr inż. Martyna Tarczewska | | |
| **Temat** | *Regex - drugie starcie* | | |
| **Student** | Marcin Ogórkiewicz | | |
| **Nr ćw.** | 8 | **Data wykonania** | 22.11.2023 |
| **Ocena** |  | **Data oddania spr.** | 22.11.2023 |

**Zadanie 1**

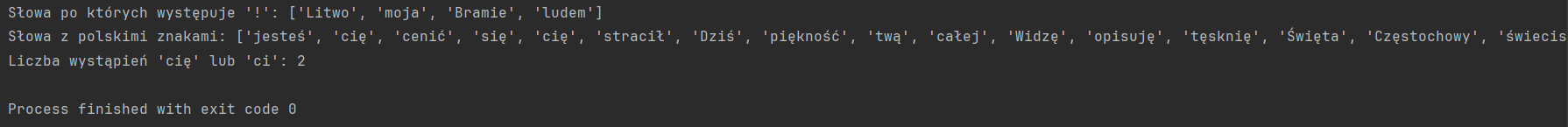
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 import re  
 text = 'aaaa'  
 print("Kwantyfikator '+':", re.findall('a+', text))  
 print("Kwantyfikator '+?':", re.findall('a+?', text))  
 print("Kwantyfikator '\*':", re.findall('a\*', text))  
 print("Kwantyfikator '\*?':", re.findall('a\*?', text))  
 print("Kwantyfikator '?':", re.findall('a?', text))  
 print("Kwantyfikator '??':", re.findall('a??', text))  
  
  
main()  
"""Kwantyfikator '+': ['aaaa'] - Dopasowuje jedno lub więcej wystąpień 'a'.  
  
Kwantyfikator '+?': ['a', 'a', 'a', 'a'] - Leniwy odpowiednik '+', dopasowuje jak najmniejszą ilość 'a'.  
  
Kwantyfikator '\*': ['aaaa', ''] - Dopasowuje zero lub więcej wystąpień 'a'.  
  
Kwantyfikator '\*?': ['', 'a', 'a', 'a', 'a', ''] - Leniwy odpowiednik '\*', dopasowuje jak najmniejszą ilość 'a'.  
  
Kwantyfikator '?': ['a', 'a', 'a', 'a', ''] - Dopasowuje zero lub jedno wystąpienie 'a'.  
  
Kwantyfikator '??': ['', 'a', '', 'a', '', 'a', '', 'a', ''] - Leniwy odpowiednik '?', dopasowuje jak najmniejszą ilość 'a'."""

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 2**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 with open('../Lab07/inwokacja.txt', 'r', encoding='utf-8') as file:  
 text = file.read()  
 words\_before\_exclamation = re.findall(r'\b\w+\b(?=\s\*!)', text)  
 print("Słowa po których występuje '!':", words\_before\_exclamation)  
 words\_with\_polish\_chars = re.findall(r'\b\w\*[ąćęłńóśźż]\w\*\b', text)  
 print("Słowa z polskimi znakami:", words\_with\_polish\_chars)  
 count\_ci = len(re.findall(r'\bcię\b|\bci\b', text, flags=re.IGNORECASE))  
 print("Liczba wystąpień 'cię' lub 'ci':", count\_ci)  
  
  
main()

****

**Zadanie 3**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 with open('adresy.txt', 'r', encoding='utf-8') as file:  
 lines = file.readlines()  
 address\_pattern = re.compile(  
 r"ul\. (?P<street\_name>[\w\s]+) (?:(?P<house\_number>\d+)(?:/(?P<apartment\_number>\d+))?)? (\d{2}-\d{3})")  
 for line in lines:  
 match = address\_pattern.search(line)  
 if match:  
 street\_name = match.group('street\_name').strip()  
 apartment\_number = match.group('apartment\_number').strip() if match.group('apartment\_number') else "None"  
 postal\_code = match.group(4).strip()  
 print(  
 f"Nazwa ulicy: {street\_name}, Numer mieszkania: {apartment\_number}, Kod pocztowy: {postal\_code}")  
  
  
main()

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 4**

r"(?P<street\_name>\D+)\s"

**Zadanie 5**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 print(re.match("[a-z]{3}", "Ala ma kota a kot ma Ale"))  
 print(re.match("[a-z]{3}", "Ala ma kota a kot ma Ale", flags=re.I))  
  
  
main()  
"""Flaga re.I albo inaczej re.IGNORECASE pozwala na ignorowanie wielkości liter w dopasowaniach."""

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

**Zadanie 6**

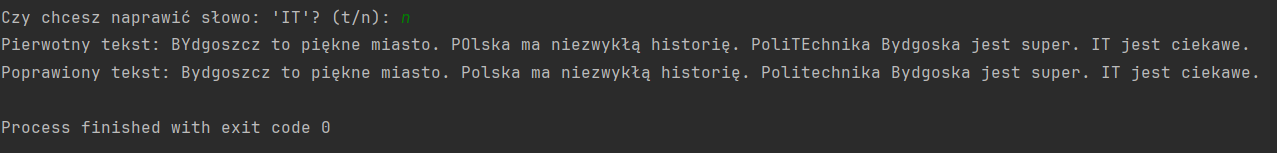
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def validate\_password(password):  
 # Słabe hasło: Minimum 8 znaków  
 weak\_pattern = re.compile(r'^.{8,}$')  
 # Średnie hasło: Minimum 8 znaków, przynajmniej jedna duża litera, jedna mała litera i jedna cyfra  
 medium\_pattern = re.compile(r'^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d).{8,}$')  
 # Mocne hasło: Minimum 10 znaków, przynajmniej jedna duża litera, jedna mała litera, jedna cyfra i jeden znak specjalny  
 strong\_pattern = re.compile(r'^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*\W).{10,}$')  
 if strong\_pattern.match(password):  
 return "Password is strong"  
 elif medium\_pattern.match(password):  
 return "Password is medium"  
 elif weak\_pattern.match(password):  
 return "Password is weak"  
 else:  
 return "Password does not meet the criteria for weak, medium, or strong"  
  
  
def main() -> None:  
 password1 = "weakPass"  
 password2 = "MediumPass1"  
 password3 = "Strong@Pass123"  
 print(validate\_password(password1))  
 print(validate\_password(password2))  
 print(validate\_password(password3))  
  
  
main()

**Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie**

**Zadanie 7**

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import re  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def fix\_capitalization(match):  
 word = match.group(0)  
 if len(word) > 2:  
 return word.capitalize()  
 else:  
 user\_input = input(f"Czy chcesz naprawić słowo: '{word}'? (t/n): ")  
 if user\_input.lower().strip() == 't':  
 return word.capitalize()  
 else:  
 return word  
  
  
def correct\_capitalization(text):  
 pattern = re.compile(r'\b([A-Z]{2}[a-z]\*|[a-z]+[A-Z]+[a-z]\*|[A-Z][a-z]\*[A-Z]+[a-z]\*)\b')  
 corrected\_text = pattern.sub(fix\_capitalization, text)  
 return corrected\_text  
  
  
def main() -> None:  
 text = \  
 "BYdgoszcz to piękne miasto. POlska ma niezwykłą historię. PoliTEchnika Bydgoska jest super. IT jest ciekawe."  
 corrected\_text = correct\_capitalization(text)  
 print("Pierwotny tekst:", text)  
 print("Poprawiony tekst:", corrected\_text)  
  
  
main()

****

**Wnioski**

Ćwiczenie pomogło mi bardziej zaznajomić się z działaniami na ciągach znaków przy pomocy funkcji wbudowanych oraz biblioteki re. Przepraszam za opóźnienie w przesłaniu. Sprawozdanie jest już gotowe od tygodnia, tylko zapomniałem go wysłać.