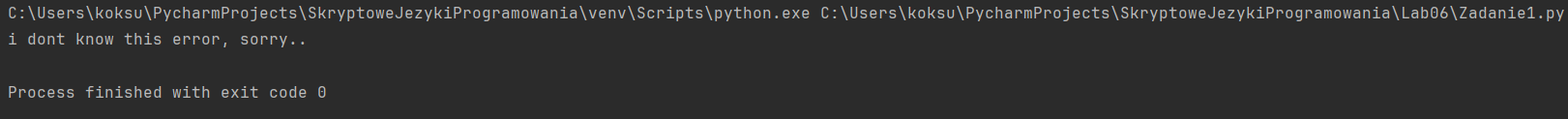
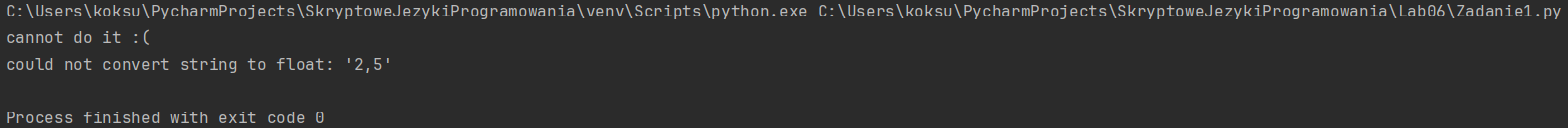
|  |  |  |  |
| --- | --- | --- | --- |
| Obraz zawierający symbol, design  Opis wygenerowany automatycznie | Politechnika Bydgoska im. J. J. Śniadeckich  **Wydział Telekomunikacji,**  **Informatyki i Elektrotechniki** | | Obraz zawierający symbol, logo, Grafika, Czcionka  Opis wygenerowany automatycznie |
| **Przedmiot** | Skryptowe języki programowania | | |
| **Prowadzący** | mgr inż. Martyna Tarczewska | | |
| **Temat** | *Wyjątki i pliki* | | |
| **Student** | Marcin Ogórkiewicz | | |
| **Nr ćw.** | 6 | **Data wykonania** | 06.11.2023 |
| **Ocena** |  | **Data oddania spr.** | 06.11.2023 |

Zadanie 1.

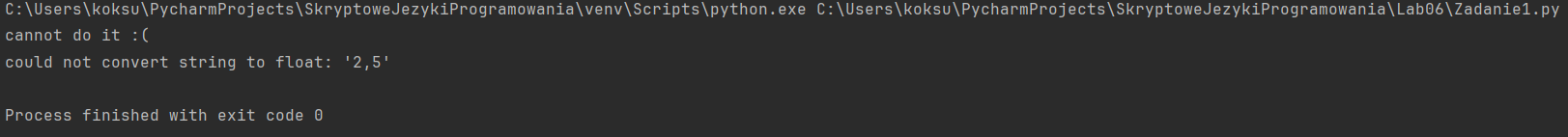
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 input\_string = "2,5"  
 try:  
 some\_number = 3 / 0  
 #some\_number = float(input\_string)  
 #some\_number = float(input\_string)  
 print(f"This number is: {some\_number}")  
 except (ValueError, UnicodeError) as ex1:  
 print("cannot do it :(")  
 print(ex1)  
 except NameError:  
 print("i dont know this name :(")  
 except:  
 print("i dont know this error, sorry..")  
  
  
main()



#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 input\_string = "2,5"  
 try:  
 #some\_number = 3 / 0  
 some\_number = float(input\_string)  
 #some\_number = float(input\_string)  
 print(f"This number is: {some\_number}")  
 except (ValueError, UnicodeError) as ex1:  
 print("cannot do it :(")  
 print(ex1)  
 except NameError:  
 print("i dont know this name :(")  
 except:  
 print("i dont know this error, sorry..")  
  
  
main()



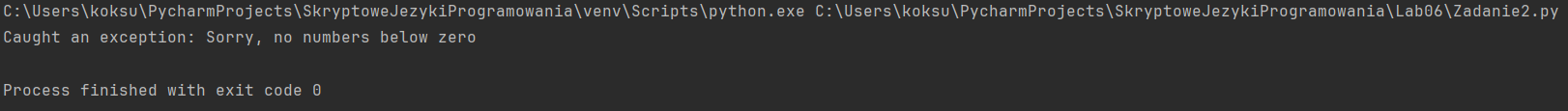
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 input\_string = "2,5"  
 try:  
 #some\_number = 3 / 0  
 #some\_number = float(input\_string)  
 some\_number = float(input\_string)  
 print(f"This number is: {some\_number}")  
 except (ValueError, UnicodeError) as ex1:  
 print("cannot do it :(")  
 print(ex1)  
 except NameError:  
 print("i dont know this name :(")  
 except:  
 print("i dont know this error, sorry..")  
  
  
main()



Odpowiedzi do pytania znajdują się na zrzutach ekranu z wynikami działania programu.

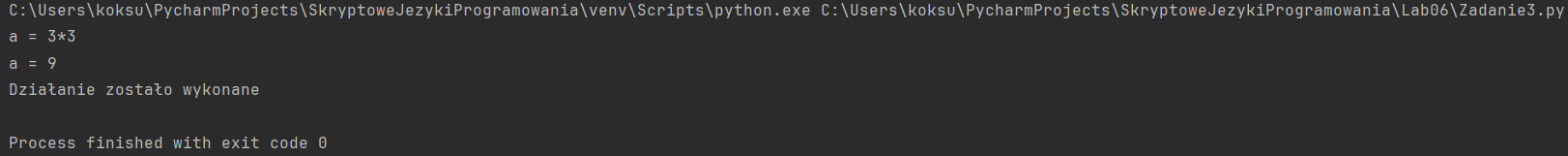
Zadanie 2

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 x = -1  
 try:  
 if x < 0:  
 raise ValueError("Sorry, no numbers below zero")  
 except ValueError as e:  
 print(f"Caught an exception: {e}")  
  
  
main()



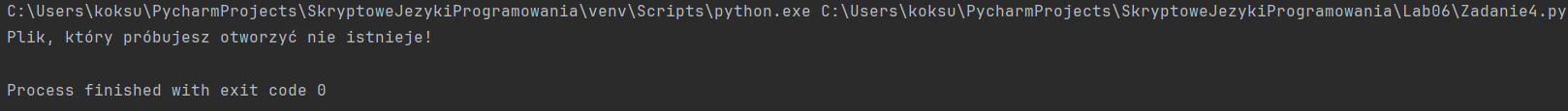
Zadanie 3

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 try:  
 a = 3\*3  
 if a != 9:  
 raise ValueError("Coś poszło mocno nie tak!")  
 except ValueError:  
 print(f"{ValueError}")  
 else:  
 print("a = 3\*3\na = 9")  
 finally:  
 print("Działanie zostało wykonane")  
  
  
main()



Zadanie 4

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 try:  
 f = open("/sciezka/sciezka/plik.txt", "r")  
 print(f.read()) # odczyt całego pliku do końca  
 f = open("plik.txt", "r")  
 print(f.readline()) # odczyt jednej linii  
 f = open("plik.txt", "r")  
 print(f.read(10)) # odczyt pierwszych 10 znaków  
 f = open("thefile.txt", "r")  
 for x in f:  
 print(x) # odczyt w pętli  
 f = open("plik.txt", "r")  
 list = f.readlines() # wszystkie linie do listy  
 list = f.readlines(10) # 10 pierwszych do listy  
 f.close() # otwarte pliki należy zamykać  
 except FileNotFoundError:  
 print("Plik, który próbujesz otworzyć nie istnieje!")  
  
  
main()



Błąd, który się wyświetla po próbie uruchomienia programu z brakującym plikiem:

FileNotFoundError: [Errno 2] No such file or directory: '/sciezka/sciezka/plik.txt'

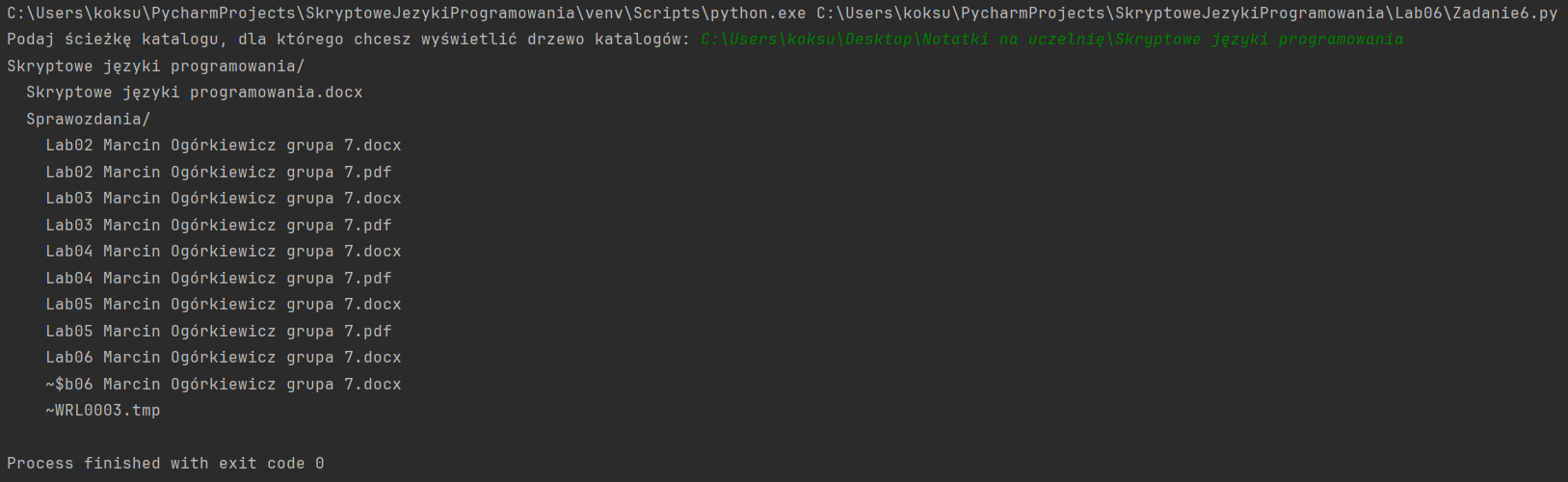
Przy próbie zapisu do nieistniejącego pliku, zostaje on utworzony w katalogu z podanej ścieżki.

Zadanie 5

|  |  |  |
| --- | --- | --- |
| Parametr funkcji open | Plik istnieje | Plik nie istnieje |
| o | Otwarcie pliku w trybie do  pisania | Błąd ... |
| w | Otwarcie i nadpisanie całej zawartości pliku | Utworzenie nowego pliku i zapisanie w nim zawartości |
| a | Otwarcie pliku do zapisu i dopisanie nowych danych po istniejących | Utworzenie nowego pliku i zapisanie w nim zawartości |
| x | Błąd | Utworzenie nowego pliku i zapisanie w nim zawartości |

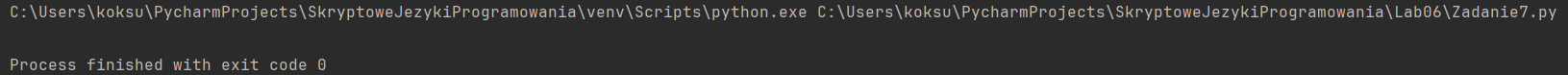
Zadanie 6

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import os  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def tree(directory, indent=""):  
 if os.path.isdir(directory):  
 print(indent + os.path.basename(directory) + "/")  
 for item in os.listdir(directory):  
 item\_path = os.path.join(directory, item)  
 if os.path.isdir(item\_path):  
 tree(item\_path, indent + " ")  
 else:  
 print(indent + " " + item)  
  
  
def main() -> None:  
 target\_directory = input("Podaj ścieżkę katalogu, dla którego chcesz wyświetlić drzewo katalogów: ")  
 tree(target\_directory)  
  
  
main()



Zadanie 7

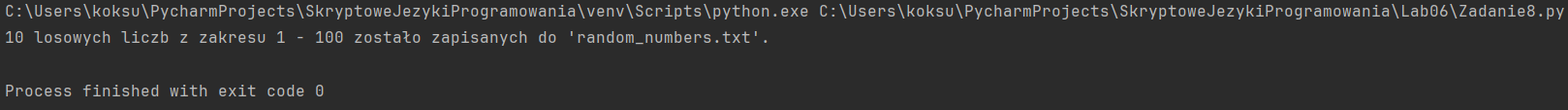
#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import shutil  
  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 shutil.copy("test\_file\_a.txt", "../Lab05")  
 shutil.copy("test\_file\_b.txt", "../Lab05")  
 shutil.copy("test\_file\_b.txt", "../nieistnieje")  
  
  
main()



Kiedy plik w docelowym katalogu nie istnieje, polecenie po prostu wkleja do niego skopiowany plik. Kiedy plik w docelowym katalogu istnieje, polecenie nadpisuje istniejący plik skopiowanym plikiem. Kiedy docelowy katalog nie istnieje, polecenie tworzy nowy plik z zawartością skopiowanego pliku.

Zadanie 8

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import random  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def random\_numbers(n: int, a: int, b: int, filename: str) -> None:  
 with open(filename, "w") as file:  
 for i in range(n):  
 random\_number = random.randint(a, b)  
 file.write(str(random\_number) + "\n")  
  
  
def main() -> None:  
 n = 10   
 a = 1   
 b = 100   
 filename = "random\_numbers.txt"   
 random\_numbers(n, a, b, filename)  
 print(f"{n} losowych liczb z zakresu {a} - {b} zostało zapisanych do '{filename}'.")  
  
  
main()

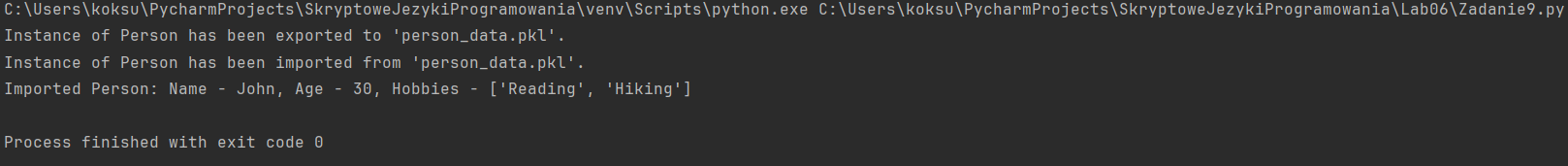


Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

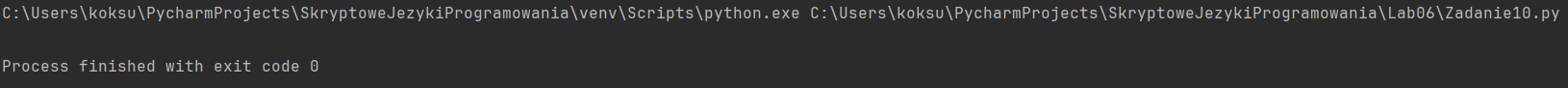
Zadanie 9

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
import pickle  
# stałe i zmienne globalne  
  
# funkcje  
  
  
class Person:  
 def \_\_init\_\_(self, name: str, age: int, hobbies: list) -> None:  
 self.name = name  
 self.age = age  
 self.hobbies = hobbies  
  
 def export\_to\_file(self, filename: str) -> None:  
 try:  
 with open(filename, "wb") as file:  
 pickle.dump(self, file)  
 print(f"Instance of {self.\_\_class\_\_.\_\_name\_\_} has been exported to '{filename}'.")  
 except Exception as e:  
 print(f"Error exporting instance: {e}")  
  
 @classmethod  
 def import\_from\_file(cls, filename: str) -> any:  
 try:  
 with open(filename, "rb") as file:  
 instance = pickle.load(file)  
 if isinstance(instance, cls):  
 print(f"Instance of {cls.\_\_name\_\_} has been imported from '{filename}'.")  
 return instance  
 else:  
 print(f"Error: The imported object is not an instance of {cls.\_\_name\_\_}.")  
 except FileNotFoundError:  
 print(f"Error: File '{filename}' not found.")  
 except Exception as e:  
 print(f"Error importing instance: {e}")  
  
  
def main() -> None:  
 person = Person("John", 30, ["Reading", "Hiking"])  
 person.export\_to\_file("person\_data.pkl")  
 imported\_person = Person.import\_from\_file("person\_data.pkl")  
 if imported\_person:  
 print(f"Imported Person: Name - {imported\_person.name}, Age - {imported\_person.age}, "  
 f"Hobbies - {imported\_person.hobbies}")  
  
  
main()



Zadanie 10

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
from Zadanie8 import random\_numbers  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def main() -> None:  
 n = 10 # Liczba losowych liczb do wygenerowania  
 a = 1 # Dolna granica przedziału  
 b = 100 # Górna granica przedziału  
 filename = "random\_numbers.txt" # Nazwa pliku, w którym zostaną zapisane liczby  
 random\_numbers(n, a, b, filename)  
 with open("random\_numbers.txt", "r") as file:  
 nums = file.readlines()  
 file.close()  
 for i in range(len(nums)):  
 nums[i].strip()  
 nums[i] = bin(int(nums[i]))  
 with open("random\_numbers\_bin.txt", "w") as bin\_file:  
 for i in range(len(nums)):  
 bin\_file.write(f"{nums[i]}\n")  
  
  
main()



Obraz zawierający zrzut ekranu, tekst, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Wnioski

Żadne zadanie, za wyjątkiem zadania nr 9, nie sprawiło mi problemu. Zadanie 9 skłoniło mnie do poczytania o serializacji i deserializacji obiektów Pythona.