|  |  |  |  |
| --- | --- | --- | --- |
| Obraz zawierający symbol, Grafika, logo, Prostokąt  Opis wygenerowany automatycznie | Politechnika Bydgoska im. J. J. Śniadeckich  Wydział Telekomunikacji,  Informatyki i Elektrotechniki  Obraz zawierający symbol, logo, Grafika, Czcionka  Opis wygenerowany automatycznie**Zakład Systemów Teleinformatycznych** | |  |
| **Przedmiot** | Skryptowe języki programowania | | |
| **Prowadzący** | mgr inż. Martyna Tarczewska | | |
| **Temat** | Python - wprowadzenie | | |
| **Student** | Marcin Ogórkiewicz, grupa nr. 7 | | |
| **Nr ćw.** | 2 | **Data wykonania** | 12.10. |
| **Ocena** |  | **Data oddania spr.** | 12.10. |

Zadanie 1

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import platform  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def wersja() -> None:  
 ver = platform.python\_version()  
 print("Aktualnie wykorzystywana wersja Pythona: ", ver, "\n")  
  
  
def main() -> None:  
 wersja()  
  
  
main()

Zadanie 2

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import platform  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def wersja() -> None:  
 ver = platform.python\_version()  
 print("Aktualnie wykorzystywana wersja Pythona: ", ver, "\n")  
  
  
def main() -> None:  
 wersja()  
  
  
main()

Zadanie 3

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import datetime  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def dataczas\_mdy() -> str:  
 now = datetime.datetime.now()  
 data\_czas1 = now.strftime("%m.%d.%Y %H:%M:%S")  
 return data\_czas1  
  
  
def dataczas\_dmy() -> str:  
 now = datetime.datetime.now()  
 data\_czas2 = now.strftime("%d.%m.%Y %H:%M:%S")  
 return data\_czas2  
  
  
def dataczas\_ymd() -> str:  
 now = datetime.datetime.now()  
 data\_czas3 = now.strftime("%Y.%m.%d %H:%M:%S")  
 return data\_czas3  
  
  
def main() -> None:  
 print(dataczas\_dmy(), "\n", dataczas\_mdy(), "\n", dataczas\_ymd())  
  
  
main()

Zadanie 4

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
from math import pi  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def promien() -> float:  
 print("Podaj promień koła:")  
 r = float(input())  
 return r  
  
  
def obwod\_kola(r: float) -> float:  
 obw\_kol = 2\*pi\*r  
 return obw\_kol  
  
  
def pole\_kola(r: float) -> float:  
 pol\_kol = pi \* r \*\* 2  
 return pol\_kol  
  
  
def main() -> None:  
 r = promien()  
 print("Obwód koła dla podanego promienia wynosi: ", obwod\_kola(r), ", natomiast jego pole wynosi: ",  
 pole\_kola(r))  
  
  
main()

Zadanie 5

Zmienna „a” w funkcji przyjmuje wartość nadaną w samej funkcji, jeżeli ta została ponownie zdefiniowana. „A” ma taką samą wartość, jaką zdefiniowałem wcześniej. Interpreter rozróżnia „a” i „A”.

Zadanie 6

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def imie\_nazwisko() -> None:  
 imie = input("Podaj swoje imie\n")  
 nazwisko = input("Podaj swoje nazwisko\n")  
 print(nazwisko, imie)  
  
  
def main() -> None:  
 imie\_nazwisko()  
  
  
main()

Zadanie 7

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
from math import sqrt, pow  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def delta(a: float, b: float, c: float) -> float:  
 d = pow(b, 2) - 4 \* a \* c  
 return d  
  
  
def pierwiastki\_rownania\_kwadratowego(a: float, b: float, d: float) -> tuple:  
 pierwiastek1 = (-b-sqrt(d))/(2\*a)  
 pierwiastek2 = (-b+sqrt(d))/(2\*a)  
 return pierwiastek1, pierwiastek2  
  
  
def main() -> None:  
 a = float(input("Podaj wartość a równania\n"))  
 b = float(input("Podaj wartość b równania\n"))  
 c = float(input("Podaj wartość c równania\n"))  
 odp = pierwiastki\_rownania\_kwadratowego(a, b, delta(a, b, c))  
 print("Podane równanie ma dwa rozwiązania:\nx1 = ", odp[0], "\nx2 = ", odp[1])  
  
  
main()

Zadanie 8

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import datetime  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def data\_roznica\_dni(data1: datetime, data2: datetime) -> int:  
 roznica = data2-data1  
 roznica\_dni = roznica.days  
 return roznica\_dni  
  
  
def main() -> None:  
 data1 = datetime.datetime(2023, 10, 10)  
 data2 = datetime.datetime(2023, 12, 16)  
 print(data\_roznica\_dni(data1, data2))  
  
  
main()

Zadanie 9

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import calendar  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def generuj\_kalendarz(rok: int, miesiac: int) -> calendar:  
 try:  
 kalendarz = calendar.month(rok, miesiac)  
 return kalendarz  
 except IndexError:  
 return "Błąd: Nieprawidłowy rok lub miesiąc"  
  
  
def main() -> None:  
 rok = int(input("Podaj rok: "))  
 miesiac = int(input("Podaj miesiąc (1-12): "))  
 kalendarz = generuj\_kalendarz(rok, miesiac)  
 print(f"Kalendarz dla {miesiac}.{rok}:\n")  
 print(kalendarz)  
  
  
main()

Zadanie 10

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def calc() -> None:  
 a = input("Podaj pierwszą liczbę\n")  
 b = input("Podaj drugą liczbę\n")  
 a = float(a)  
 b = float(b)  
 dodawanie = a+b  
 odejmowanie = a-b  
 mnozenie = a\*b  
 dzielenie = a/b  
 dzielenie\_calk = a//b  
 modulo = a % b  
 potegowanie = a\*\*b  
 print("a+b = ", dodawanie, "\na-b = ", odejmowanie, "\na\*b = ", mnozenie, "\na/b = ", dzielenie, "\na//b = ",  
 dzielenie\_calk, "\na%b = ", modulo, "\na\*\*b = ", potegowanie, "\n")  
  
  
def main():  
 calc()  
  
  
main()

Przy dzieleniu przez zero program zwraca błąd.

Zadanie 11

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
# funkcje  
  
  
def ppunkta() -> None:  
 if 2 \* 3 / 4 == 1.5:  
 print("a) Mnożenie i dzielenie wykonuje się od lewej strony\n")  
 else:  
 print("a) Mnożenie i dzielenie wykonuje się od prawej strony\n")  
  
  
def ppunktb() -> None:  
 if 2 + 3 \* 4 == 14 and 2 \* 3 + 4 == 10:  
 print("b) Mnożenie wykona się wcześniej\n")  
 else:  
 print("b) Mnożenie i dodawanie są równe i wykonują się od lewej strony\n")  
  
  
def ppunktc() -> None:  
 print("c) 2\*2/4\*4=", 2\*2/4\*4, "\n")  
 print("(2\*2)/(4\*4)=", (2\*2)/(4\*4), "\n")  
 print("Nawiasy wpływają na kolejność wykonywania działań, nawiasy wykonują się w kolejności od lewej do prawej "  
 "i mają priorytet nad wszystkim innymi działaniami\n")  
  
  
def ppunktd() -> None:  
 if 2 \* 2 \*\* 5 == 64:  
 print("d) Potęgowanie jest ważniejsze od mnożenia")  
 else:  
 print("d) Mnożenie jest tak samo ważne jak potęgowanie")  
  
  
def main() -> None:  
 ppunkta()  
 ppunktb()  
 ppunktc()  
 ppunktd()  
  
  
main()

1. Mnożenie i dzielenie wykonuje się od lewej strony
2. Mnożenie wykona się wcześniej
3. Nawiasy wpływają na kolejność wykonywania działań, wykonywane są w kolejności od lewej do prawej i mają priorytet nad wszystkim innymi działaniami
4. Potęgowanie wykona się wcześniej

Zadanie 12

#!C:\Users\koksu\AppData\Local\Programs\Python\Python310  
  
# importy  
import typing  
  
# stałe i zmienne globalne  
  
#funkcje  
  
  
def dwainty() -> None:  
 a = 4/2  
 print("a) ", type(a))  
  
  
def intfloat() -> None:  
 b = 4/2.5  
 print("b) ", type(b))  
  
  
def dwafloaty() -> None:  
 c = 4.5/2.5  
 print("c) ", type(c))  
  
def main() -> None:  
 dwainty()  
 intfloat()  
 dwafloaty()  
  
  
main()

Wnioski:

Ćwiczenie pozwoliło mi przypomnieć sobie podstawowe informacje związane z Pythonem.