DEFENSIVE PROGRAMMING

BEFORE CLASS

1. Zapoznaj się z materiałem dotyczącym programowania defensywnego - lekcje od 40 do 48 na kanale „Introduction to Computer Science and Programming Using Python (MIT)”.

2. Zapoznaj się z materiałami dostępnymi w witrynie w3schools.com dotyczącymi obsługi wyjątków.

3. Zapoznaj się z materiałem dotyczącym obsługi wyjątków w języku Python „Exceptions in Python”:

   https://youtu.be/nlCKrKGHSSk

4. Zapoznaj się z materiałami dotyczącymi błędów oraz obsługi i zgłaszania wyjątków w języku Python

   https://docs.python.org/3/tutorial/errors.html
   https://wiki.python.org/moin/UsingAssertionsEffectively

DURING CLASS

Assertions

Asercja to instrukcja sprawdzająca wewnętrzny stan programu. Zawiera warunek logiczny, którego niespełnienie (fałsz) powoduje przerwanie działania programu.

5. Napisz program, który wyznaczy rezultat dzielenia dwóch liczb całkowitych. Zastosuj asercję dla sprawdzenia czy mianownik jest różny od zera. Sprawdź działanie programu. Zmień wartość b przypisując 0 i zobacz, jaki będzie rezultat działania programu.

```
a = 5
b = 3
assert b!=0, 'Wartość b równa 0!'
print(f'{a}/{b} = {a/b}')
```

6. W programie wyznaczającym iloraz dwóch wartości dokonaj również sprawdzenia typu wartości zmiennych a i b (wartości całkowite). Rozbuduj warunek asercji wykorzystując funkcję type:

```
type(a) == int
```

w taki sposób, aby zarówno wartość zmiennej b była różna od zera, jak i typ wartości a i b wskazywał na liczby całkowite.

7. Zmienne ‘wzrost’ i ‘waga’ zawierają informacje o wzroście i wadze osoby. Napisz program obliczający wskaźnik BMI. Dane dotyczące wzrostu i wagi odczytaj z klawiatury. Stosując asercje sprawdź, czy wzrost wyrażony w cm jest liczbą całkowitą i mieści się w przedziale <150;220>, natomiast waga wyrażona w kg jest liczbą rzeczywistą i mieści się w przedziale <40.0; 150.0>

   Następnie sprawdź działanie programu dla poniższych danych, Zastanów się, dla których danych program powinien działać poprawnie.

   a. wzrost 182, waga 79.6
   b. wzrost 182, waga 79
   c. wzrost 182.3 waga 79.6
   d. wzrost 192, waga 170
   e. wzrost 148, waga 55

Exception handling

W trakcie wykonywania programu mogą wystąpić zdarzenia (sytuacje wyjątkowe), które uniemożliwiają jego prawidłowe działanie. Przyczynami mogą być np. niepoprawny rodzaj czy wartość danych, problem z wykonaniem operacji na pliku w systemie plików, czy też brak połączenia z siecią komputerową. Obsługa sytuacji wyjątkowych (obsługa wyjątków) to mechanizm pozwalający zareagować na te zdarzenia przy użyciu wyspecjalizowanych konstrukcji języka programowania.

8. Napisz program, który obliczy i wyświetli pierwiastek kwadratowy z liczby wprowadzonej z klawiatury. Zastosuj obsługę wyjątków dla sprawdzenia poprawności wprowadzanych danych (liczba >= 0). Sprawdź działanie programu wprowadzając z klawiatury wartość:

   a. 256
   b. 0
   c. -4
   d. ‘cztery’

   Zaobserwuj, jaki będzie rezultat działania programu w każdym z tych przypadków.

```
import math

try:
    number = float(input('Enter any number: '))
    print (f'sqrt({number}) = {math.sqrt(number)}' )
except:
    print('Please enter a valid number greater than 0')
```

9. W grupach 2-3 osobowych, zmodyfikuj powyższy program, aby działał on do momentu wprowadzenia poprawnej liczby większej od 0. Zastosuj obsługę wyjątków oraz instrukcję pętli. Pomocne może być przeanalizowanie przykładów zawartych w materiale:

https://www.youtube.com/watch?v=ij9F6z0gppI&list=PLRJdqdXieSHN0U9AdnmwD-9QcR9hmw04d&index=47&t=0s

10. Napisz program, który wyświetli zawartość pliku NoEducation.txt. Uwzględnij sytuację wyjątkową (brak pliku o podanej nazwie lub jego niepoprawne otwarcie). Zastosuj obsługę wyjątków. Następnie sprawdź działanie programu w różnych sytuacjach, np. gdy podany plik istnieje lub nie istnieje.

Raising exceptions

Zgłaszanie sytuacji wyjątkowych jest jednym z mechanizmów pozwalających na kontrolowanie poprawności działania programu. W przypadku wystąpienia zdarzenia, które może spowodować niepoprawne działanie programu, możliwe jest, przy użyciu odpowiedniej instrukcji, poinformowanie o sytuacji wyjątkowej (zgłoszenie wyjątku). Następnie taki zgłoszony wyjątek może zostać obsłużony przy użyciu konstrukcji programistycznych przeznaczonych do obsługi wyjątków (np. try ... except).

11. Napisz program, który wyświetla wiek osoby w latach. Sprawdź, czy informacja o liczbie lat to liczba całkowita. Jeśli nie, zgłoś sytuację wyjątkową TypeError.

```
wiek = '25'
if type(wiek) != int:
    raise TypeError('Wiek powinien być liczbą całkowitą!')
print(f'Masz {wiek} lat')
```

12. Uzupełnij program wyświetlający wiek osoby, sprawdzając również, czy podany wiek jest liczbą dodatnią, mniejszą od 120. Jeśli nie, zgłoś wyjątek ValueError. Następnie sprawdź działanie programu dla poprawnych i niepoprawnych wartości wieku.

13. Napisz program, który dla podanej ceny netto wyznacza cenę brutto, tj. cenę netto powiększoną o podatek VAT (23%). Sprawdź, czy cena netto jest dodatnią liczbą rzeczywistą lub całkowitą. Jeśli tak, oblicz i wyświetl cenę brutto z dwoma miejscami dziesiętnymi oraz symbolem waluty zł (np. 15.60 zł). Jeśli cena netto zawiera niepoprawną wartość, zgłoś odpowiedni wyjątek. Następnie sprawdź działanie programu dla następujących wartości ceny netto:

```
cena_netto = 15.6
cena_netto = 15
cena_netto = -7
cena_netto = ”9.20”
```

14. Utwórz funkcję pole_prostokata(a,b) do obliczania pola powierzchni prostokąta o wymiarach boków a oraz b. Sprawdź w funkcji, czy przekazane wartości a oraz b to liczby całkowite lub rzeczywiste większe od 0. Jeśli nie, zgłoś odpowiedni wyjątek. Następnie napisz program, w którym oblicz i wyświetl pole prostokąta. Dodaj w programie obsługę zgłoszonego wyjątku. Sprawdź działanie programu, dla poniższych wartości a oraz b:

```
a=3,   b=4
a=3.1, b=4.0
```

```
a=3,    b='4'
a=3,    b=-2
```

AFTER CLASS

15. W grupach 2-3 osobowych zastanów się, gdzie w poniższym programie komputerowym należy użyć asercji, aby zapobiec niepoprawnemu i nielogicznemu działaniu programu. Następnie umieść asercje w programie i sprawdź jego działanie dla różnych kombinacji danych.

```
student = 'Mateusz'
stypendium = 950
wydatki = 920

print('Student: {}'.format(student.upper()))
print('stypendium: {} zł'.format(stypendium))
print('Wydatki: {} zł'.format(wydatki))
print('Oszczędności: {} zł'.format(stypendium-wydatki))
```

16. Utwórz funkcję zwracającą losową wartość rzutu kostką i reprezentującą liczbę wyrzuconych oczek. Zastosuj asercję weryfikującą poprawność wartości zwracanej przez funkcję (liczba naturalna z przedziału od 1 do 6). Napisz program wyświetlający rezultat rzutu kostką.

17. Napisz program, który na podstawie numeru dnia tygodnia wprowadzonego z klawiatury wyświetli pełną nazwę dnia tygodnia. Zastosuj obsługę wyjątków dla niepoprawnej wartości wprowadzonej z klawiatury.

18. Napisz program, który umożliwia wprowadzenie z klawiatury imienia oraz nazwiska osoby. Wyświetl wprowadzone imię i nazwisko w formacie:

    ```
    Imię, NAZWISKO
    ```

    Zastosuj asercje, aby sprawdzić, czy wprowadzone imię oraz nazwisko zawierają co najmniej 3 znaki.

19. Nr pesel składa się z 11 cyfr. Napisz program, który wyświetla nr pesel wprowadzony z klawiatury. Zastosuj asercje, aby sprawdzić, czy wprowadzony nr pesel zawiera dokładnie 11 znaków i każdy z nich jest cyfrą.

20. Poniższa klasa Pracownik zawiera informacje o pracowniku, tj. jego imię i nazwisko (osoba), staż pracy w latach oraz wynagrodzenie. Uzupełnij klasę o metodę obliczająca i zwracającą dodatek stażowy w zależności od przepracowanych lat. Pracownik otrzymuje dodatek stażowy w wysokości 1% wynagrodzenia za każdy przepracowany rok powyżej pięciu, nie więcej jednak, niż 20%.

    W konstruktorze klasy dodaj asercje sprawdzające poprawność informacji o tworzonym pracowniku.

Uzupełnij klasę Pracownik o metodę zwracającą tekstową reprezentację obiektu.

Utwórz obiekt opisujący pracownika oraz wyświetl dane o pracowniku (imię i nazwisko, staż pracy, wynagrodzenie, dodatek stażowy oraz łączne wynagrodzenie powiększone o dodatek stażowy. Wyświetl kwoty z dwoma miejscami dziesiętnymi oraz symbolem waluty.

```
class Pracownik:
    def __init__(self, osoba, staz_pracy, wynagrodzenie):
        self.osoba = osoba
        self.staz_pracy = staz_pracy
        self.wynagrodzenie = wynagrodzenie
```

21. Utwórz funkcję iloczyn_zbiorow(zbior1,zbior2), która zwraca iloczyn dwóch zbiorów. Sprawdź w funkcji poprawność przekazanych danych. Gdy dane są niepoprawne, zgłoś wyjątek. Następnie napisz program, który wyświetla zawartość dwóch zbiorów oraz ich iloczyn. Dodaj w programie obsługę wyjątków.

22. Plik students.csv zawiera listę studentów. Każdy student opisany jest poprzez imię, nazwisko, identyfikator oraz nazwę uniwersytetu. Napisz program wyświetlający listę studentów. W przypadku, gdy którykolwiek wiersz nie zawiera wszystkich czterech wymienionych informacji, zgłoś wyjątek. Sprawdź działanie programu usuwając z pliku pojedynczą informację.