**Testy jednostkowe - zadania**

**Zad 1.**

Stwórz funkcję sprawdzającą, czy liczba jest pierwsza. Otestuj ją, pamiętając równocześnie o otestowaniu każdego edge-case’a (np. 1 nie jest liczbą pierwszą).

Spróbuj stworzone testy umieścić w klasie.

Uwaga:

Aby uruchomić testy znajdujące się w klasie, np. TestIsPrime, użyj polecenia:

>>> pytest tests/tests.py::TestIsPrime

**Zad 2.**

Napisz funkcję, która zwracać będzie “Fizz”, gdy prześlesz do niej wartość podzielną przez 3, “Buzz”, gdy podzielną przez 5, a “FizzBuzz”, gdy liczba będzie podzielna przez obie te wartości. Napisz do niej testy jednostkowe.

**Zad 3.**

Napisz funkcję sortującą metodą quick sort listę podanych elementów. Otestuj jej poprawne działanie.

**Zad 4.**

Poniższy kod realizuje funkcjonalność prostego notatnika TODO. Możemy do niego dodawać dowolne notatki, usuwać niektóre oraz czyścić całą listę. Umieść go w pliku src.py i otestuj każdą z funkcji. Zapewnij sprawdzenie przypadków, w których zostanie rzucony wyjątek.

| todos = ["Clean my room", "Make my bed", "Go to school", "Do school homework"]   def check\_pos(pos):  if len(todos) == 0:  raise Exception("No more todos!")  elif pos >= len(todos) or pos < 0:  raise Exception("No such item number!")     def add\_todo(content):  todos.append(content)   def remove\_todo(pos):  check\_pos(pos)   todos.pop(pos)   def edit\_todo(pos, content):  check\_pos(pos)   todos[pos] = content   def remove\_all():  todos.clear()   add\_todo("Go to bed") remove\_todo(0) edit\_todo(0, "Get up from bed") remove\_all() |
| --- |

**Zad 5.**

Rozważ poniższy program:

| from datetime import datetime, timezone  def calc\_diff(case):  end\_time = case['end\_time']  start\_time = case['start\_time']    start\_time\_obj = datetime.fromisoformat(start\_time)    if end\_time is None:  end\_time\_obj = datetime.now(timezone.utc)  else:  end\_time\_obj = datetime.fromisoformat(end\_time)    return (end\_time\_obj - start\_time\_obj).total\_seconds()     def main():  case = {  'start\_time': '2021-11-03T09:22:28+00:00',  'end\_time': None # None means that case is currently going on  }  print(calc\_diff(case))   if \_\_name\_\_ == "\_\_main\_\_":  main() |
| --- |

Odpowiedzialny jest on za wyliczanie różnicy czasowej (podanej w sekundach) między start\_time oraz end\_time.

Napisz test, który będzie sprawdzał poprawne działanie funkcji calc\_diff.

**Podpowiedź:**

Tylko tyle, albo aż tyle! Zauważ, że konieczne będzie zamockowanie datetime.now(). Dlaczego? To już zostawiam Twoim dywagacjom.