Imię i nazwisko:

Numer indeksu:

Kolokwium 1a

Zakreśl wszystkie błędy składniowe w poniższym skrypcie Pythona:

```
ii = 10

def bar():
   ii=0

def baz(ii):
   if ii < 10 && ii % 5 == 0:
     print('Warunek spełniony')

def foo(func=bar)
   for jj in range(ii):
      ii = 0
        print(ii)
      func()
```

Zadanie 2

Rozważając poniższy kod:

```
zoo=[
{'imie': 'Koralgol', 'masa': 5},
{'imię': 'puchatek', 'masa': 5
}]

a = (e['masa'] for e in zoo)
b = [e['masa'] for e in zoo]

print(sum(a) == sum(b))
```

Zaznacz poprawne odpowiedzi:

- **Skrypt ten wyświetli True**
- Obiekt pod zmienną a jest tożsamy z obiektem pod zmienną b
- Obiekt pod zmienną b jest inny niż ten pod zmienną a --- ponieważ jest on jednorazowego użytku
- **Interpreter Pythona potrafi łatwo określić długość obiektu b**

Zadanie 3

Napisz na odwrocie funkcję która liczy średnią liczb. Funkcja ta może przyjmować zarówno listę, jak i generator, czyli powinna zadziałać dla takiego wywołania:

```
def foo(input):
  N = 0
  sum = 0
  for val in input:
    sum+=val
    N+=1
  return sum / N

print(foo(range(10000)))
```

Uwaga rozwiązanie:

```
def foo(input):
     return sum(input)/len(input)
```

nie jest poprawne, ponieważ generator nie ma zdefiniowanej długości

Zadanie 4

Typowa implementacja zadania 3a z poprzednich zajęć wyglądała tak:

```
def furier(args, range,
           cos='cos')):
   result = []
   for val in range:
     tmp = 0
     for ii, arg in enumerate(args):
       if cos == 'cos':
          tmp+=arg*math.cos(
             ii*range*2*math.pi)
      else:
          tmp+=arg*math.sin(
             ii*range*2*math.pi)
```

Nie winikając w poprawność tej implementacji

Imię i nazwisko: Numer indeksu:

przepisz tą funkcję by działała tak samo, ale nie posiadała wewnątrz słów kluczowych if oraz else.

Oryginalnie myślałem o takim rozwiązaniu:

```
def furier(args, range,
           cos=math.cos)):
   result = []
   for val in range:
     tmp = 0
     for ii, arg in enumerate(args):
          tmp+=arg*cos(
             ii*range*2*math.pi)
```

Ktoś zaproponował takie:

```
func_dict = {'cos': math.cos,
'sin': math.sin}

def furier(args, range,
           cos='cos')):
   result = []
   for val in range:
     tmp = 0
     for ii, arg in enumerate(args):
       tmp+=arg*func_dict[cos](
             ii*range*2*math.pi)
```

Generalnie: sprawdzałem czy pamiętacie że można użyć funkcji jako obiektu.