


# TEST 'FUNDAMENTY PROGRAMOWANIA' #PYTHON - 2

Dziękuję za odpowiedź na wszystkie pytania! Przed Tobą informacja o wyniku oraz wyjaśnienia dotyczące poszczególnych pytań..

 **Twój wynik to 0%. Podsumowanie:** Ten test poszedł Ci obiektywnie słabo :(. Udało się zgromadzić mniej niż 25% możliwych punktów. Co to oznacza? No niestety musisz zmierzyć się z materiałem jeszcze raz! Dobra wiadomość? Wystarczy się trochę nauczyć i będzie dobrze!

## **Rada dla Ciebie:**

Tutaj znajdziesz kod z tego testu do przeklikania: [Tutaj](#)

Python fajny jest, taki elastyczny i przyjazny. Już w poniedziałek poznasz język zupełnie inny, taki, hmm, sztywny bardziej, ale dla wielu jest to zaleta a nie wada. Java, ruszamy!

SZCZEGÓŁOWE WYNIKI TESTU Z WYJAŚNIENIAMI

**Zadanie 1** - wskazałeś: `print(F"Mam na imię {user_name}")` - zła odpowiedź ⓘ

Zadanie brzmiało: Która z poniższych instrukcji  
NIE wyświetli w konsoli "Mam na imię Jan"?

```
user_name = "Jan"
```

Do wyboru mieliśmy:

1. `print(f"Mam na imię {user_name}")`
2. `print(F"Mam na imię {user_name}")`
3. `print("Mam na imię ${user_name}")`
4. `print(f'Mam na imię {user_name}')`

Mamy tutaj przykład tzw. f-ciągu (ciągu formatowanego). W Pythonie musimy użyć `f` (małe lub wielkie - bez znaczenia) przed stringiem.

Odpowiedzi 1,2 i 4 spowodują więc wyświetlenie oczekiwanego komunikatu. Nie uda się to przy odpowiedzi trzeciej gdyż w pythonie konstrukcja `${zmienna}` nie spowoduje umieszczenia zawartości zmiennej.

Po co i kiedy używać f-stringa? Wtedy gdy chcemy umieścić w stringu zawartość zmiennej. Jeśli nie chcemy używać f-stringa (choć warto!) to możemy posłużyć się konkatencją czyli:

```
print('Mam na imię ' + user_name)
```

**Zadanie 2** - wskazałeś: `tekst[1:3]` - zła odpowiedź ⓘ

Zadanie brzmiało: Która z poniższych instrukcji zwróci ciąg znaków "bcd" z ciągu "abcdefg"

tekst = "abcdefg"?

1. tekst[1:4]
2. tekst["b":"d"]
3. tekst[2:3]
4. tekst[1:3]

Mamy tutaj przykład wyodrębnienia z jednego ciągu znaków innego ciągu (nowy ciąg jest zwracany, a ciąg na którym pracujemy nie jest zmieniany).

Podając w nawiasach kwadratowych indeks np.

tekst[1] wyodrębnimy znak z indeksem 1 a więc drugi znak, w naszym przykładzie byłaby to litera 'b'. Jeśli podamy też drugi indeks i rozdzielimy je dwukropkiem to możemy zwrócić więcej niż jeden znak np. tekst[0:2] zwróci nam stringa "ab".

Uwaga! Druga wartość podana w nawiasie kwadratowym oznacza pobranie znaków do tego indeksu, ale bez niego! Czyli [0:2] oznacza tutaj znak o indeksie 0 i 1, ale nie 2.

Prawidłowa odpowiedź, która pozwoli nam zwrócić ciąg "bcd" to tekst[1:4]

### **Zadanie 3 - wskaż: boolean() - zła odpowiedź**

Zadanie brzmiało: Jaka funkcja przekształca wartość innych typów na wartość logiczną?

Podaliśmy cztery różne funkcje, z czego trzy nazwy

były nieprawidłowe, a prawidłowa była oczywiście `bool()`

#### **Zadanie 4 - wskazałeś: +20 - zła odpowiedź**

Zadanie brzmiało: Która z tych liczb jest zapisana nieprawidłowo?

1. +20
2. 02
3. 3\_000\_122

Zapisanie liczby z plusem jest dozwolone, oznacza po prostu że liczba jest dodatnia, tak więc 20 + +20 jest poprawnym zapisem. Poprawny jest także zapis z podkreśleniami, które pełnią jedynie rolę separatora i nie mają znaczenia dla interpretera (co więcej, mogą być używane w dowolnym miejscu liczby, poza pierwszym)

Prawidłową odpowiedzią, czyli nieprawidłową liczbą (błędem składniowym) jest zapis 02. Przy czym da się użyć zera na początku liczby, z tym, że jako prefiks do zapisu w innym systemie niż dziesiętny np. 0b (zapis dwójkowy) czy 0x (zapis szesnastkowy)

#### **Zadanie 5 - wskazałeś: // - to operator dzielnia całkowitego, który zwróci liczbę całkowitą, usuwając resztę - zła odpowiedź**

Zadanie brzmiało: Które z opisów operatorów jest

## NIEPRAWDZIWE?

Do wyboru mieliśmy:

1. `//` - to operator dzielnia całkowitego, który zwróci liczbę całkowitą, usuwając resztę
2. `/` - to operator dzielenia, który zwróci zawsze wartość float (liczbę zmiennoprzecinkową) nawet gdy wynik nie zawiera reszty a obie liczby są liczbami całkowitym
3. `**` - operator potęgowania - po lewej stronie wykładnik po prawej potęga.
4. `===` - operator porównania z typem - porównuje dwie wartości, ale w taki sposób by sprawdzać także typ czyli `2=="2"` zwróci `True`, ale `2===2` zwróci `False`

Jedyna nieprawidłowa odpowiedź to odpowiedź numer

4. W Pythonie nie występuje bowiem operator porównania z trzema znakami równości. Operator z dwoma znakami równości `==` porównuje bowiem także typy tzn. `2=='2'` zwróci `False`, bo nie dokona się tutaj konwersja `"2"` na `2`

Sprawdzian w ramach nauki w bootcampie "Studiuje IT". Więcej informacji znajdziesz na **Stronie Bootcampu**