

Każdy Klub Kodowania musi być zarejestrowany. Zarejestrowane kluby można zobaczyć na mapie na stronie codeclubworld.org - jeżeli nie ma tam twojego klubu sprawdź na stronie jumpto.cc/18CpLPy (ang.) co trzeba zrobić, by to zmienić.

Wprowadzenie:

Nauczymy się korzystać z list, aby zapisywać wiele danych w jednej zmiennej.



Zadania do wykonania

Wykonaj te **POLECENIA** krok po kroku



Przetestuj swój projekt

Kliknij na zieloną flagę, aby
PRZETESTOWAĆ swój kod



Zapisz swój projekt

Teraz **ZAPISZ** swój projekt

Step 1: Miło jest być miłym

W tym projekcie napiszesz program, który będzie wyświetlał użytkownikowi losowy komplement!

✓ Lista zadań

- Do tej pory w projektach korzystaliśmy ze zmiennych aby zapisać pojedynczą informację, taką jak imię lub ilość punktów. Co, jeśli chcielibyśmy zapisać większą ilość informacji? W Pythonie można użyć *listy* do zapisania większej ilości danych w jednej zmiennej:

```
duzePlanety = [ "jowisz" , "saturn" , "uran" , "neptun" ]
```

Ta lista tekstu jest również znana pod nazwą *tablica* tekstu (po angielsku *array*). Aby uzyskać dostęp do konkretnej wartości, wystarczy znać jej pozycję w tablicy. Uruchom poniższy program żeby lepiej zrozumieć, jak działają listy:

```
duzePlanety = [ "jowisz" , "saturn" , "uran" , "neptun" ]
print( duzePlanety )
print( duzePlanety[0] )
print( duzePlanety[1] )
print( duzePlanety[2] )
print( duzePlanety[3] )
```

```
main.py
1 duzePlanety = [ "jowisz" , "saturn" , "uran" , "neptun" ]
2 print( duzePlanety )
3 print( duzePlanety[0] )
4 print( duzePlanety[1] )
5 print( duzePlanety[2] )
6 print( duzePlanety[3] )

Powered by trinket
['jowisz', 'saturn', 'uran', 'neptun']
jowisz
saturn
uran
neptun
```

Jak widzisz, pozycje zaczynają się od 0 a nie od 1, dlatego `duzePlanety[1]` to “saturn” (drugi element) a nie “jowisz”.

- Możesz skorzystać z listy `komplementy` do zapisania

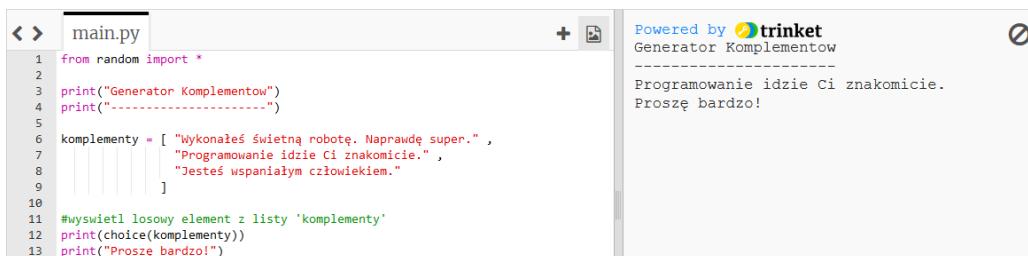
wszystkich możliwych komplementów do twojego generatora komplementów, a potem użyć `choice(komplementy)` do wybrania losowego komplementu dla użytkownika:

```
from random import *

print("Generator Komplementow")
print("-----")

komplementy = [ "Wykonałeś świetną robotę. Naprawdę super." ,
                "Programowanie idzie Ci znakomicie." ,
                "Jesteś wspaniałym człowiekiem."
            ]

#wyswietl losowy element z listy 'komplementy'
print(choice(komplementy))
print("Proszę bardzo!")
```



```
main.py
1 from random import *
2
3 print("Generator Komplementow")
4 print("-----")
5
6 komplementy = [ "Wykonałeś świetną robotę. Naprawdę super." ,
7                 "Programowanie idzie Ci znakomicie." ,
8                 "Jesteś wspaniałym człowiekiem."
9             ]
10
11 #wyswietl losowy element z listy 'komplementy'
12 print(choice(komplementy))
13 print("Proszę bardzo!")

Powered by trinket
Generator Komplementow
-----
Programowanie idzie Ci znakomicie.
Proszę bardzo!
```

- Możesz sprawić, że Twoje komplementy będą odrobinę bardziej interesujące dzięki połączeniu elementów 2 różnych list:



```
from random import *

print("Generator Komplementow")
print("-----")

przymiotniki = [ "wspaniale" , "znakomicie" , "doskonale"
]
zainteresowania = [ "jeździsz na rowerze" , "programujesz"
, "parzysz herbatę" ]
```

```

imie = input("Jak masz na imię?: ")
print( "Oto Twój komplement" , imie , ":" )



#pobierz dowolny element z obu list i połącz je w komplement


print( imie , choice(przymiotniki) ,
choice(zainteresowania) )
print( "Proszę bardzo!" )

```

```

main.py
1 from random import *
2
3 print("Generator Komplementow")
4 print("-----")
5
6 przymiotniki = [ "wspaniale" , "znakomicie" , "doskonale" ]
7 zainteresowania = [ "jeździsz na rowerze" , "programujesz" , "parzysz herbatę" ]
8
9 imie = input("Jak masz na imię?: ")
10 print( "Oto Twój komplement" , imie , ":" )
11
12 #pobierz dowolny element z obu list i połącz je w komplement
13 print( imie , choice(przymiotniki) , choice(zainteresowania) )
14 print( "Proszę bardzo!" )
15

```

Powered by trinket
Generator Komplementow

Jak masz na imię?: Dawid
('Oto Twój komplement', 'Dawid', ':')
('Dawid', 'znakomicie', 'parzysz herbatę')
Proszę bardzo!



Zapisz Swój Projekt

Wyzwanie: Dodanie większej ilości komplementów

Spróbuj wymyślić więcej komplementów i dodaj je do programu! Pamiętaj, że musisz dodać przecinek (,) pomiędzy elementami na twojej liście.



Zapisz Swój Projekt

Krok 2: Niekończące się komplementy



Lista zadań

- Wykorzystując to, co wiesz o pętlach `while` i warunkach `if`,

możesz zmienić program w taki sposób, żeby prawił komplementy do momentu, kiedy użytkownik nie zdecyduje, aby przestał:

```
from random import *

#program jest zapętlony tak dugo, jak ta zmienna jest
'True'
generujDalej = True

przymiotniki = [ "wspaniale" , "znakomicie" , "doskonale"
]
zainteresowania = [ "jeździsz na rowerze" , "programujesz"
, "parzysz herbatę" ]

print("Generator Komplementow")
print("-----")

name = input("Jak masz na imię?: ")

print('''
Menu
    n = nastepny komplement
    k = koniec
''')

while generujDalej == True:

    wyborMenu = input("\n>_").lower()

    #'n' aby wygenerowac nowy komplement
    if wyborMenu == 'n':

        print( "Oto twoj komplement" , name , ":" )

        #pobierz losowe elementy z dwóch list i polacz je w
        komplement
        print( name , choice(przymiotniki) ,
choice(zainteresowania) )
        print( "Prosze bardzo!" )

    #'k' aby zakonczyc
```

```
elif wyborMenu == 'k':  
  
    generujDalej = False  
  
else:  
  
    print("Proszę wybrać poprawną opcję!")
```

The screenshot shows a Python code editor with a file named `main.py`. The code defines a generator function `Generator Komplementow` that asks the user for their name and then prints three random complements. The code uses `yield` statements to return multiple values. The terminal window to the right shows the interaction with the generator, where it asks for a name, provides three complements ('Dawid', 'znakomicie', 'parzysz herbatę'), and then ends with a message 'Proszę bardzo!'. The code editor has syntax highlighting and a status bar at the bottom.

```
1 from random import *
2
3 #program jest zapętlony tak dugo, jak ta zmienna jest 'True'
4 generujDalej = True
5
6 przymiotniki = [ "wspaniale", "znakomicie", "doskonale" ]
7 zainteresowania = [ "jeździsz na rowerze", "programujesz", "parzysz herbatą" ]
8
9 print("Generator Komplementow")
10 print("-----")
11
12 name = input("Jak masz na imię?: ")
13
14 print('''
15     Menu
16     n = nastepny komplement
17     k = koniec
18     ''')
19
20 < >_ < >
```

```
Generator Komplementow
-----
Jak masz na imię?: Dawid

        Menu
        n = nastepny komplement
        k = koniec

>_ n
('Oto twoj komplement', 'Dawid', ':')
('Dawid', 'znakomicie', 'parzysz herbatę')
Proszę bardzo!

>_ n
('Oto twoj komplement', 'Dawid', ':')
('Dawid', 'doskonale', 'programujesz')
Proszę bardzo!

>_ k
```

Pamiętaj, że pętla `while` będzie trwała tak długo, aż zmienna `generujDalej` będzie ustawiona na `True`. Kiedy użytkownik wybierze `k`, `generujDalej` jest ustawione na `False`.



Zapisz Swój Projekt

Krok 3: Dopasowanie komplementów



Lista zadań

- Twój generator komplementów zaczyna wyglądać porządnie, ale jest jeden problem: co, jeśli użytkownik nie potrafi jeździć na rowerze ani parzyć herbaty? W takim przypadku komplementy nie będą prawdziwe i nie podnoszą go na duchu!

Zmodyfikujmy Twój program w taki sposób, aby użytkownik mógł zmieniać zawartość listy `zainteresowania` i dopasować

do siebie komplementy, które otrzymuje:

```
from random import *

generujDalej = True
przymiotniki = [ "wspaniale" , "znakomicie" , "doskonale"
]
zainteresowania = [ "jeździsz na rowerze" , "programujesz"
, "parzysz herbatę" ]

print("Generator Komplementow")
print("-----")

name = input("Jak masz na imię?: ")

print('''
Menu
    n = nastepny komplement
    d = dodaj zainteresowanie
    u = usun zainteresowania z listy
    p = pokaz zainteresowania
    k = koniec
''')

while generujDalej == True:

    wyborMenu = input("\n>_").lower()

    #'n' aby wygenerowac nowy komplement
    if wyborMenu == 'n':

        print( "Oto Twój komplement" , name , ":" )

        #pobierz losowe elementy z dwóch list i polacz je w
        komplement
        print( name , choice(przymiotniki) ,
choice(zainteresowania) )
        print( "Prosze bardzo!" )

    #'d' aby dodac zainteresowanie
    elif wyborMenu == 'd':

        elementDoDodania = input("Wprowadź zainteresowanie
```

```

        do dodania do listy: ")
            zainteresowania.append(elementDoDodania)

        #'u' aby usunac zainteresownie z listy
        elif wyborMenu == 'u':
            elementDoUsuniecia = input("Wprowadź
zainteresownie do usunięcia z listy: ")
            zainteresowania.remove(elementDoUsuniecia)

        #'p' wyświetl listę zainteresowań
        elif wyborMenu == 'p':
            print(zainteresowania)

        #'k' aby zakończyć
        elif wyborMenu == 'k':
            generujDalej = False

        else:
            print("Proszę wybrać poprawną opcję!")

```

Jak widzisz, można używać `append()` do dodawania elementów do listy i `remove()` do ich kasowania. Uruchom ten program i dopasuj zainteresowania do Twoich potrzeb. Proś program o komplementy, aż poprawi Ci się humor!

- Czy podczas testowania powyższego programu pojawiły się jakieś problemy? W tej chwili, Twój program zawiesza się, kiedy próbujesz usunąć element, który nie jest na liście:

```

main.py
36     zainteresowania.append(elementDoDodania)
37
38     #'u' aby usunac zainteresownie z listy
39     elif wyborMenu == 'u':
40
41         elementDoUsuniecia = input("Wprowadź zainteresownie do usunięcia z listy:
42             zainteresowania.remove(elementDoUsuniecia)
43
44     #'p' wyświetl listę zainteresowań
45     elif wyborMenu == 'p':
46         print(zainteresowania)
47
48     #'k' aby zakończyć
49     elif wyborMenu == 'k':
50
51         generujDalej = False
52
53

```

Jak masz na imię?: Dawid

Menu

n = następny komplement
d = dodaj zainteresowanie
u = usun zainteresowania z listy
p = pokaz zainteresowania
k = koniec

>_ p
['jeżdzisz na rowerze', 'programujesz', 'parzysz herbatę']

>_ u
Wprowadź zainteresownie do usunięcia z listy: spiewasz

ValueError: list.index(x): x not in list on line 44 in main.py

Można to naprawić, sprawdzając najpierw, czy element który chcesz usunąć jest już na liście. Podmień kod usuwający zainteresowanie na taki:

```
#'u' aby usunac zainteresownie z listy
elif wyborMenu == 'u':

    elementDoUsuniecia = input("Wprowadz
zainteresownie do usuniecia z listy: ")
    #usun element tylko wtedy kiedy znajduje sie na
liscie
    if elementDoUsuniecia in zainteresowania:
        zainteresowania.remove(elementDoUsuniecia)
    else:
        print("Zainteresownie nie znajduje sie na
liscie!")
```

Uruchom program i spróbuj usunąć zainteresowanie z poza listy:

```
main.py
1 from random import *
2
3 generujDalej = True
4 przyjemniki = [ "wspaniale", "znakomicie", "doskonale" ]
5 zainteresowania = [ "jeżdzisz na rowerze", "programujesz", "parzysz herbatę" ]
6
7 print("Generator Komplementow")
8 print("-----")
9
10 name = input("Jak masz na imię?: ")
11
12 print('..')
13 Menu
14     n = nastepny komplement
15     d = dodaj zainteresowanie
16     u = usun zainteresowania z listy
17     p = pokaz zainteresowania
18
19 Jak masz na imię?: Dawid
20
21 Menu
22     n = nastepny komplement
23     d = dodaj zainteresowanie
24     u = usun zainteresowania z listy
25     p = pokaz zainteresowania
26     k = koniec
27
28 >_ p
29 ['jeżdzisz na rowerze', 'programujesz',
30 'parzysz herbatę']
31
32 >_ u
33 Wprowadz zainteresownie do usuniecia z
34 listy: spiewasz
35 Zainteresownie nie znajduje sie na liscie!
36
37 >_
```



Zapisz Swój Projekt

Wyzwanie: Powtarzające się zainteresowania

Innym problemem jest to, że możliwe jest dodanie kilku takich samych zainteresowań:

The screenshot shows a code editor with a Python file named `main.py`. The code handles user input for a menu system. A terminal window is open to the right, displaying the menu options and their descriptions. The menu includes options for printing the list, adding an interest, removing an interest, and exiting.

```
main.py
41:     elif wyborMenu == 'u':
42:         elementDoUsuniecia = input("Wprowadź zainteresowanie do usunięcia z listy:")
43:         zainteresowania.remove(elementDoUsuniecia)
44:
45:     #'p' wyświetli listę zainteresowań
46:     elif wyborMenu == 'p':
47:         print(zainteresowania)
48:
49:     #'k' aby zakończyć
50:     elif wyborMenu == 'k':
51:
52:         generujDalej = False
53:
54:     else:
55:
56:         print("Proszę wybrać poprawną opcję!")

menu
n = następny komplement
d = dodaj zainteresowanie
u = usuń zainteresowania z listy
p = pokaz zainteresowania
k = koniec

>_ p
['jeździsz na rowerze', 'programujesz',
'parysz herbatę']

>_ d
Wprowadź zainteresowanie do dodania do
listy: programujesz

>_ p
['jeździsz na rowerze', 'programujesz',
'parysz herbatę', 'programujesz']

>_ 
```

Czy umiesz naprawić ten problem w taki sposób, aby można było dodać zainteresowanie tylko, jeśli nie ma go jeszcze na liście:

```
if elementDoDodania not in zainteresowania:
    #dodaj tutaj swój kod...
```



Zapisz Swój Projekt

Wyzwanie: Usługa nazywania zwierzaków

Napisz program, który pomoże właścicielom nazywać ich podopiecznych:

Powered by  trinket
Uslugi nazywania zwierzakow

```
-----  
('Nazwij swojego zwierzaka', 'Azor')
```



Twój program może:

- pozwalać użytkownikowi dodawać i usuwać imiona z listy;
- nadawać różne imiona chłopcom i dziewczynkom albo różnym gatunkom zwierząt;
- zapytać użytkownika, ile imion będzie im potrzebne, na wypadek gdyby mieli kilka zwierząt.



Zapisz Swój Projekt