

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](http://jumpto.cc/18CpLPy) (ang.) co trzeba zrobić, by to zmienić.

## Wprowadzenie:

W tym projekcie nauczysz się jak napisać swój własny program szyfrujący, którym będzie możliwa wysyłanie tajnych wiadomości do przyjaciół.



### Zadania do wykonania

Wykonaj te **POLECENIA** krok po kroku



### Przetestuj swój projekt

Kliknij na zieloną flagę, aby  
**PRZETESTOWAĆ** swój kod



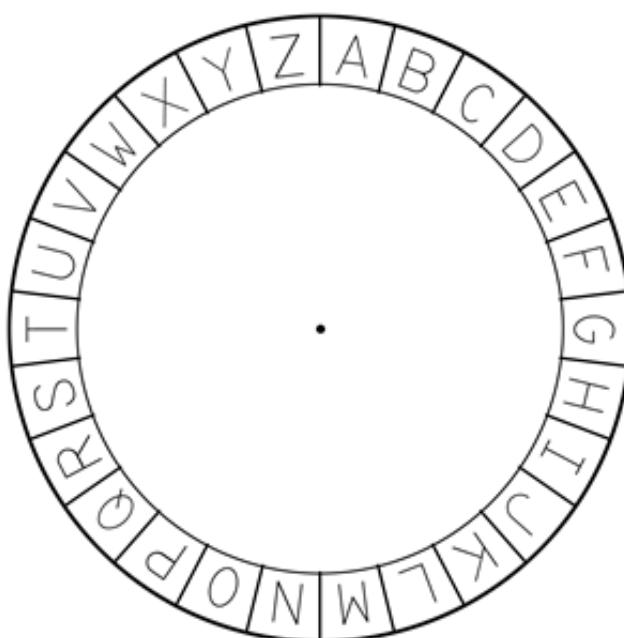
### Zapisz swój projekt

Teraz **ZAPISZ** swój projekt

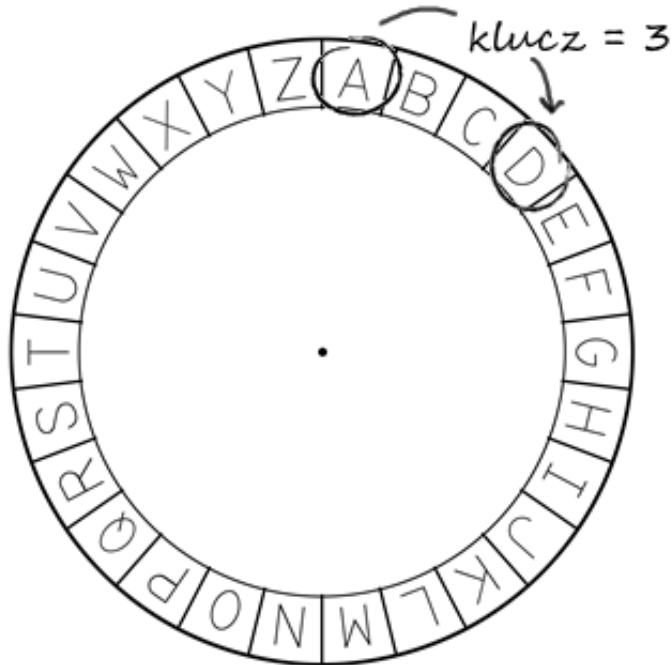
## Krok 1: Szyfrowanie liter

Szyfrowanie jest typem tajnego kodu, w którym zamienia się litery tak, aby nikt niepożądany nie mógł odczytać wiadomości. Będziemy używać najstarszej i najbardziej znanej metody szyfrowania, szyfru Cezara, którego nazwa pochodzi od gościa, który nazywał się Juliusz Cezar.

Zaczyna się od narysowania liter w kole, jak poniżej:



Aby uzyskać zaszyfrowaną literę, musisz posiadać tajny klucz. W naszym przykładzie użyjmy cyfry 3. Aby zakodować literę "a" po prostu przesuwasz 3 litery zgodnie z kierunkiem wskazówek zegara, co daje literę "d":



Aby odszyfrować wiadomość trzeba każdą literę przesunąć z powrotem o 3 miejsca, w przeciwnym kierunku.

## Lista zadań

- Zacznijmy od napisania programu do szyfrowania liter. Uruchom poniższy program i sprawdź czy działa, gdy wpiszesz literę "a":



```
# lista liter do szyfrowania
alfabet = "abcdefghijklmnopqrstuvwxyz"

# tajny klucz to 3
klucz = 3

litera = input("Wprowadz litere do zaszyfrowania: ")

# znajdz pozycje litery w alfabetie
# na przyklad "a" jest na pozycji 0,
# "e" jest na pozycji 4, itd.
pozycja = alfabet.find(litera)

# dodaj tajny klucz, aby otrzymac pozycje
# zaszyfrowanej litery
```

```

# % 26 oznacza "wroc do 0 kiedy osiągniesz 26"
nowaPozycja = (pozycja + klucz) % 26

# zaszyfrowana litera znajduje się
# na pozycji nowaPozycja w alfabetie
zaszyfrowanaLitera = alfabet[nowaPozycja]

print("Twoja litera po zaszyfrowaniu to " +
      zaszyfrowanaLitera)

```

```

main.py
1 #lista liter do szyfrowania
2 alfabet = "abcdefghijklmnopqrstuvwxyz"
3
4 #tajny klucz to 3
5 klucz = 3
6
7 litera = input("Wprowadź litere do zaszyfrowania: ")
8
9 #znajduje pozycję litery w alfabetie
10 #na przykład 'a' jest na pozycji 0, 'e' jest na pozycji 4, itd.
11 pozycja = alfabet.find(litera)
12
13 #dodaj tajny klucz aby otrzymać pozycję zaszyfrowanej litery
14 # % 26 oznacza 'wroc do 0 kiedy osiągniesz 26'
15 nowaPozycja = (pozycja + klucz) % 26
16
17 #zaszyfrowana litera znajduje się na pozycji nowaPozycja w alfabetie
18 zaszyfrowanaLitera = alfabet[nowaPozycja]
19
20 print("Twoja litera po zaszyfrowaniu to" , zaszyfrowanaLitera)
21
22 |

```

Powered by trinket  
Wprowadź litere do zaszyfrowania: a  
('Twoja litera po zaszyfrowaniu to', 'd')

- W Pythonie tekst może być rozumiany jako wiele pojedynczych liter złączonych razem (nazywanych *tablicą* znaków)

```
pozycja = alfabet.find(litera)
```

znajduje pozycję znaku w `alfabecie`. W większości języków programowania pozycje zawsze zaczynają się od 0 a nie od 1, więc w tekście “abcdefghijklmnopqrstuvwxyz”, “a” jest na pozycji 0, “b” jest na pozycji 1 i tak dalej.

**alfabet** = “

0	1	2	3	4
a	b	c	d	e

Następnie tajny klucz jest dodany do `pozycji`, przez co otrzymujemy `nowaPozycje` zaszyfrowanej litery. W naszym

przykładzie, “a” jest na pozycji 0, więc po dodaniu tajnego klucza mamy  $0 + 3 = 3$ .

Kod `% 26` oznacza, że podczas szukania numeru pozycji zaszyfrowanej litery, numer jest cofany do 0 jeśli osiągnie wartość 26. To oznacza, że litera “z” w naszym kodzie jest tak jakby na pozycji przy literze “a”.

Następnie, używamy obliczonego numeru nowej pozycji do znalezienia zakodowanej litery w `alfabecie` i wyświetlenia jej na ekranie.

```
alfabet[nowaPozycja]
```

wyszukuje literę na danej pozycji, dlatego `alfabet[0]` to “a”, `alfabet[3]` to “d”.

Zauważ też, że w programie skorzystaliśmy z krótszej wersji wczytywania komend użytkownika. Zamiast napisać:

```
print("Wprowadz litere do zaszyfrowania: ")
litera = input()
```

możesz użyć jednej linii:

```
litera = input("Wprowadz litere do zaszyfrowania: ")
```

- Możesz użyć tego samego programu do odszyfrowania litery przez użycie klucza `-3` zamiast `3`. To znaczy, że aby odszyfrować literę, poruszasz się w odwrotnym kierunku po alfabetie, wracając do “z” po “a”.

```
main.py
1 #lista liter do szyfrowania
2 alfabet = "abcdefghijklmnopqrstuvwxyz"
3
4 #tajny klucz to 3
5 #klucz do odszyfrowania = -3
6 klucz = -3
7
8 litera = input("Wprowadz litere do odszyfrowania: ")
9
10 #znajdz pozycje litery w alfabcie
11 #na przyklad 'a' jest na pozycji 0, 'e' jest na pozycji 4, itd.
12 pozycja = alfabet.find(litera)
13
14 #dodaj tajny klucz aby otrzymac pozycje odszyfrowanej litery
15 # % 26 oznacza 'wroc do 0 kiedy osiągniesz 26'
16 nowaPozycja = (pozycja + klucz) % 26
17
18 #zaszyfrowana litera znajduje sie na pozycji nowaPozycja w alfabcie
19 odszyfrowanaLitera = alfabet[nowaPozycja]
20
```

Powered by trinket  
Wprowadz litere do odszyfrowania: d  
('Twoja litera po odszyfrowaniu to', 'a')

Jeśli chcesz mieć osobne programy do szyfrowania i odszyfrowania, po prostu zamień kod powyżej tak, aby poruszał się wstecz po alfabetie:

```
# odejmij wartosc klucza aby sie cofnac
nowaPozycja = (pozycja - klucz) % 26
```



Zapisz Swój Projekt

## Wyzwanie: Zmienne klucze

Zmodyfikuj swój program szyfrujący tak, żeby użytkownik mógł wprowadzić swoją wartość klucza. Wczytaj dane wpisywane przez użytkownika i zapisać je do zmiennej `klucz`. Pamiętaj o użyciu funkcji `int()` do zamiany wczytywanych danych na liczbę.



Zapisz Swój Projekt

## Wyzwanie: Zaszyfruj i odszyfruj litery

- Użyj swojego programu do zaszyfrowania:
  - Litery "d", korzystając z klucza 7;
  - Litery "x", korzystając z klucza 4;
- Czy z pomocą twojego programu możesz odszyfrować tą wiadomość:
  - oqlmd (tajny klucz wynosi 12)



Zapisz Swój Projekt

## Krok 2: Szyfrowanie wiadomości

Zamiast szyfrować i odszyfrowywać wiadomości litera po literze, napiszmy program który będzie sam szyfrował i odszyfrowywał całe wiadomości!



### Lista Zadań

- Do tej pory korzystaliśmy z pętli do wykonania kodu:
  - określoną liczbę razy,
  - do momentu aż coś się stanie w programie.



Jest jeszcze jedna metoda korzystania z pętli polegająca na powtarzaniu wykonania kodu dla każdego elementu w zbiorze danych. Na przykład, jeśli chcesz wyświetlić każdą literę w czymś imieniu poprzez wyświetlenie każdej z liter po kolei:

```
imie = input("Jak masz na imię? ")

# wyświetl każdą literę imienia po kolejno
for litera in imie:
    print(litera)
```

```
main.py
1 imie = input("Jak masz na imie? ")
2
3 #wyświetl każdą literę imienia po kolei
4 for litera in imie:
5     print(litera)
6
7
```

Powered by trinket  
Jak masz na imie? Andrzej  
A  
n  
d  
r  
z  
e  
j

W powyższym programie, każda litera imienia jest kolejno zapisywana do zmiennej `litera` i wyświetlana. `litera` jest zwyczajną zmienną, więc możesz zmienić jej nazwę jeśli chcesz. Uruchom powyższy program i sprawdź jak działa.

- Możesz użyć tego typu pętli do zaszyfrowania całej wiadomości i odszyfrowania jej literze:



```
# lista liter do szyfrowania
alfabet = "abcdefghijklmnopqrstuvwxyz"

# wczytaj wiadomosc uzytkownika
wiadomosc = input("Prosze wprowadzic wiadomosc do zaszyfrowania: ").lower()

# ta zmienna bedzie zawierac zaszyfrowana
# wiadomosc
zaszyfrowanaWiadomosc = ""

# wczytaj klucz
klucz = input("Podaj klucz: ")
# ta akcja jest potrzebna aby miec pewnosc
# ze program wczytal klucz jako liczbe
klucz = int(klucz)

# wykonaj petle na kazdej literze w wiadomosci
for litera in wiadomosc:

    if litera in alfabet:

        # znajdz pozycje litery w alfabetie
        # na przyklad "a" jest na pozycji 0,
```

```

# "e" jest na pozycji 4, itd.
pozycja = alfabet.find(litera)

# dodaj tajny klucz aby otrzymać pozycje
# zaszyfrowanej litery
# % 26 oznacza "wroc do 0 kiedy osiągniesz 26"
nowaPozycja = (pozycja + klucz) % 26

# dodaj zaszyfrowana litere do wiadomosci
# zaszyfrowana litera znajduje sie
# na pozycji nowaPozycja w alfabetie
zaszyfrowanaWiadomosc = zaszyfrowanaWiadomosc +
alfabet[nowaPozycja]

else:

    # niektore litery (np. 'E', '?') nie sa
    # uwzglednione w alfabetie, dlatego po prostu
    # dodaj je w niezaszyfrowanej formie
    zaszyfrowanaWiadomosc = zaszyfrowanaWiadomosc +
litera

print("Twoja zaszyfrowana wiadomosc:" ,
zaszyfrowanaWiadomosc)

```

```

main.py
1 #lista liter do szyfrowania
2 alfabet = "abcdefghijklmnopqrstuvwxyz"
3
4 #wczytaj wiadomosc użytkownika
5 wiadomosc = input("Proszę wprowadzić wiadomość do zaszyfrowania: ").lower()
6
7 #ta zmienna bedzie zawierac zaszyfrowana wiadomosc
8 zaszyfrowanaWiadomosc = ""
9
10 #wczytaj klucz
11 klucz = input("Podaj klucz: ")
12 #ta akcja jest potrzebna aby mieć pewność że program wczytał klucz jako liczbę
13 klucz = int(klucz)
14
15 #wykonaj pętle na każdej literze w wiadomości
16 for litera in wiadomosc:
17     if litera in alfabet:
18         #znajdz pozycje litery w alfabetie

```

Powered by trinket  
Proszę wprowadzić wiadomość do zaszyfrowania: super tajna wiadomosc  
Podaj klucz: 3  
('Twoja zaszyfrowana wiadomosc:', 'vxshu wdmqd zdgprprvf')

W tym programie każda litera w wiadomości jest szyfrowana po kolej i dołączana do zmiennej `zaszyfrowanaWiadomosc`. Na końcu programu wyświetlana jest cała wiadomość.

Istnieją litery, które mogą być wprowadzone przez użytkownika, a które nie znajdują się w naszym `alfabecie`. Na

przykład: spacje, przecinki, znaki zapytania jak i polskie znaki takie jak ą i ę. Wyrażenie `if litera in alfabet` oznacza że tylko litery, które pojawiają się w `alfabecie` są szyfrowane. Inne znaki są po prostu dołączane, bez szyfrowania.



Zapisz Swój Projekt

## Wyzwanie: Szyfrowanie i odszyfrowanie wiadomości

Zaszyfruj pewne wiadomości i przekaż je koledze/koleżance razem z tajnym kluczem. Sprawdź czy mogą je odszyfrować ich programem.



Zapisz Swój Projekt

## Wyzwanie: Ulepszenie szyfru

Czy da się odszyfrować swoją wiadomość bez klucza? Czy możesz zmienić swój program tak, aby było trudniej złamać twój szyfr i odczytać wiadomość? Oto kilka pomysłów:

- Poprzestawiaj litery w zmiennej `alfabet`;
- Dodaj 1 do klucza, za każdym razem kiedy litera jest szyfrowana;
- Usuń spacje i znaki z poza alfabetu z szyfrowanej wiadomości.



Zapisz Swój Projekt

# Wyzwanie: Kalkulator miłości

Napisz program, który pokazuje jak zgodne są 2 osoby przez obliczenie ilości punktów zgodności.

```
main.py
1 print(" Kalkulator Miłosci ")
2 print("<3 <3 <3 <3 <3 <3")
3
4 imiona = input("\nWprowadź imiona dwóch ludzi: ")
5 punkty = 0
6
7 for litera in imiona:
8
9     if litera in "kocha":
10         punkty += 10
11
12     if litera in "fhwy":
13         punkty += 5
14
15     if litera in "aeiou":
16         punkty += 3
17
18     if litera in "z":
19         punkty += 10
```

Powered by trinket  
Kalkulator Miłosci  
<3 <3 <3 <3 <3 <3 <3  
Wprowadź imiona dwóch ludzi: zenon i basia  
('Wasze punkty zgodności wynoszą ', 58)

Program może analizować kolejne litery z 2 imion i za każdym razem kiedy program znajdzie pewne litery dodawać punkty do zmiennej `punkty`. Możesz sam zdecydować na jakich zasadach będą liczone punkty. Na przykład, możesz dodawać punkty za samogłoski, albo za litery występujące w słowie “kocha”:

```
if litera in "aeiou":  
    punkty = punkty + 5  
  
if litera in "kocha":  
    punkty = punkty + 10
```

Możesz dodać wyświetlanie wiadomości, w zależności od osiągniętego wyniku:

```
if punkty < 10:  
    print("Zapomnij!")
```



Zapisz Swój Projekt