

# Praktyczny Python

Projekt 4 / 10

Połączony Plik



sekurak.pl

securITum



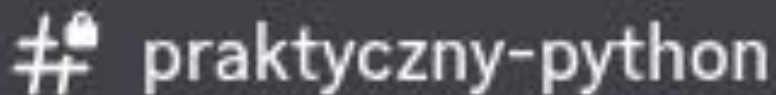
HexArcana



# Sprawy organizacyjne

**Serwer Discord:**

<https://discord.gg/AcEMbBUhwX>

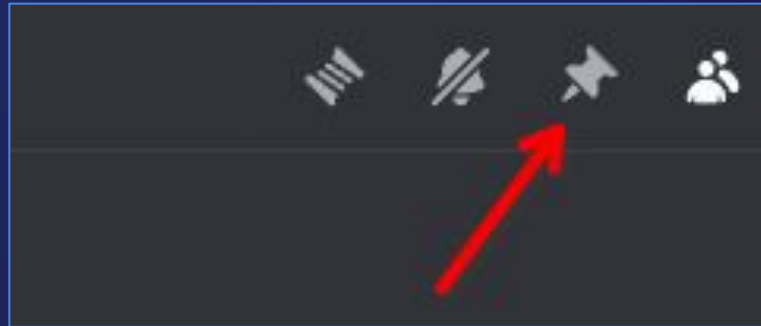
A dark grey rectangular button with rounded corners. On the left is a white Discord server icon (a hash symbol with a lock). To its right is the text 'praktyczny-python' in a white, sans-serif font.

# praktyczny-python

Instrukcje jak dostać się na Tajny Kanał  
dostaliście na email ponad tydzień temu.

## Serwer Discord:

Ważne → Przypięte wiadomości



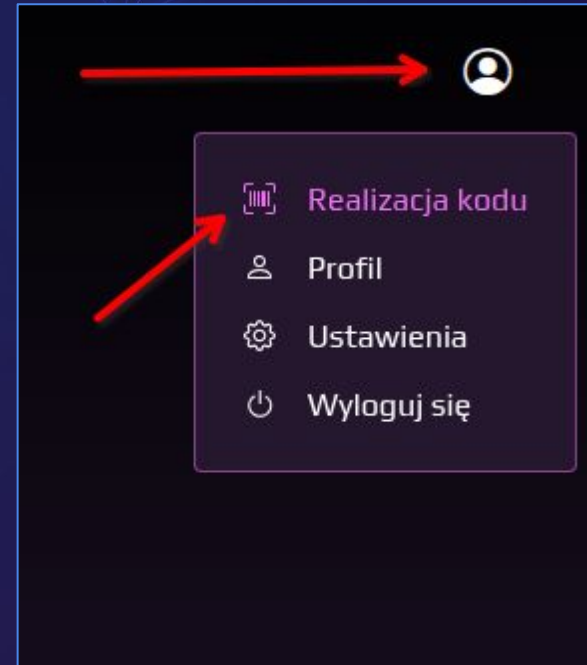
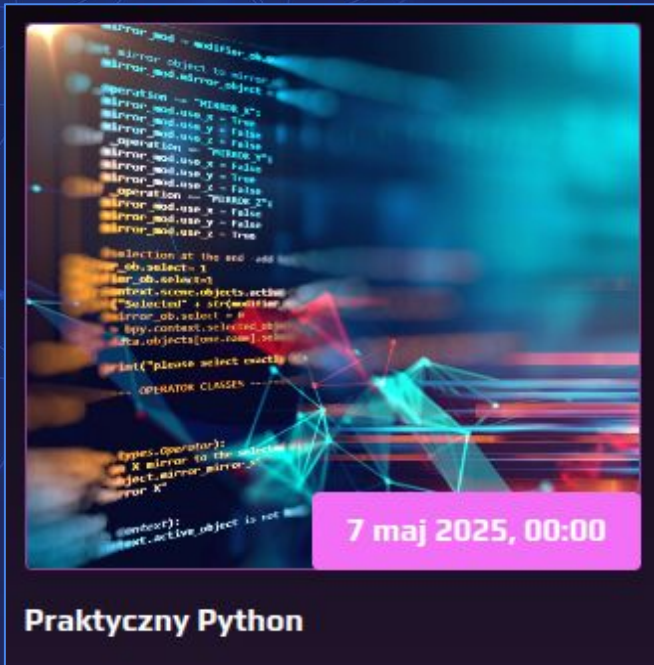
**Dostęp do materiałów (slajdy, nagrania, ...):**

Discord (przypięta wiadomość)

Mailing

[cwiczenia.hackArcana.pl](https://cwiczenia.hackArcana.pl)





Menu / Realizacja kodu:

**162e276989940185** ("Praktyczny Python")

**4e5350046a7fe3e6** ("Wstęp do programowania i Pythona")

# Sprawy edukacyjne





# Projekt 4

## Czas trwania spotkania:

2-3h

## Przerwy:

co 50 minut, 10 minut

## Pytania:

z **Q**: na początku  
podczas sesji **Q&A** (koniec)  
na Discordzie

## Materiały:

discord, mailing,  
cwiczenia.hackArcana.pl  
dostępne przez rok

## Certyfikaty:

po ostatnim spotkaniu (lipiec)

## Połamany Plik

Duży plik +  
Wolne łącze =  
Problem

Hasze  
Pliki

Sockety (gniazda sieciowe)  
I może troszkę HTTP

# Problem



1234 TB

Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(



1234 TB

Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(

W praktyce nie cały plik jest uszkodzony, tylko fragmenty.





1234 TB

Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(

W praktyce nie cały plik jest uszkodzony, tylko fragmenty.

Cel: namierzyć fragmenty i je pobrać ponownie!



1234 TB

Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(

W praktyce nie cały plik jest uszkodzony, tylko fragmenty.

Cel: namierzyć fragmenty i je pobrać ponownie!

Ale... skąd mamy wiedzieć co jest uszkodzone?



1234 TB

Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(

W praktyce nie cały plik jest uszkodzony, tylko fragmenty.

Cel: namierzyć fragmenty i je pobrać ponownie!

Ale... skąd mamy wiedzieć co jest uszkodzone?

A gdyby tak jakoś dało się pobrać mniej danych o tym co w danym fragmencie jest...



Ściągnęliśmy DUŻY plik, ale jest uszkodzony :(

W praktyce nie cały plik jest uszkodzony, tylko fragmenty.

Cel: namierzyć fragmenty i je pobrać ponownie!

Ale... skąd mamy wiedzieć co jest uszkodzone?

A gdyby tak jakoś dało się pobrać mniej danych o tym co w danym fragmencie jest...



1234 MB

Mały uszkodzony plik, ale wolne łącze





# **Teoria (i trochę praktyki)**

# Pliki



Składają się z 0 lub więcej bajtów



```
with open("asdf", "rb") as f:
```

Wczytanie 10 bajtów.

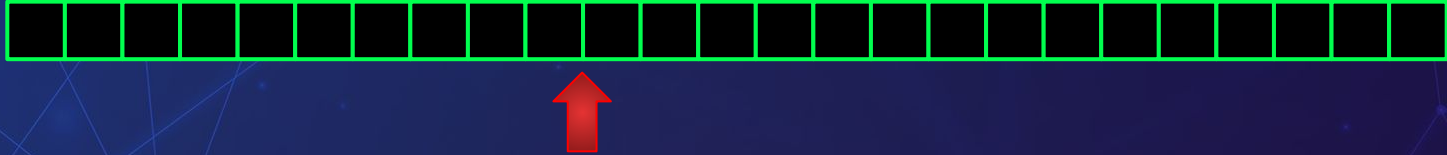


```
with open("asdf", "rb") as f:  
    data = f.read(10)
```

**data** 

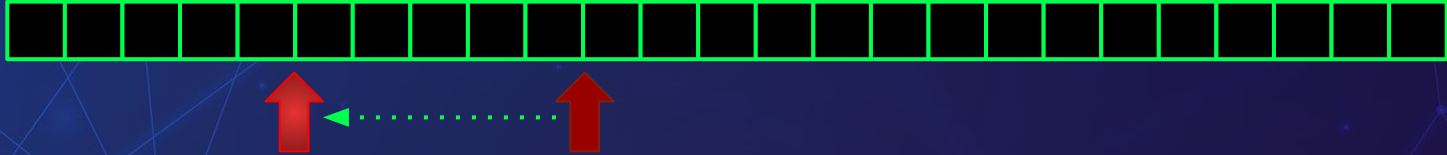
Wczytanie 10 bajtów.





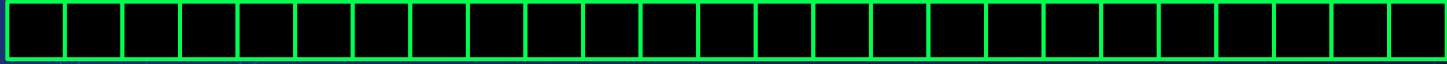
```
with open("asdf", "rb") as f:  
    data = f.read(10)
```

"Kursor" pozostaje  
przesunięty



```
with open("asdf", "rb") as f:  
    data = f.read(10)  
    f.seek(5)
```

Przesunięcie "kursora" / "główicy"



```
with open("asdf", "rb") as f:  
    data = f.read(10)  
    f.seek(5)
```

Przesunięcie "kursora" / "główicy"

# Patchowanie – parametry open()

*Do patchowania*

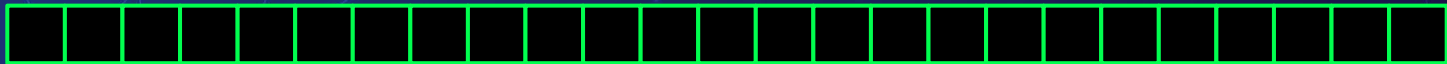
Jak odczyt, tylko w drugą stronę ;)

write() zamiast read()

Plik trzeba otworzyć "do zapisu" – **uwaga** na flagi:

Flagi	Tryb	Poprzednia zawartość	Początkowa pozycja kursora/główicy	Jeśli plik nie istnieje
"wb"	Tylko do zapisu	<b>JEST USUWANA</b>	Początek pliku (0)	Zostanie utworzony
"ab"	Tylko do zapisu	Bez zmian	Koniec pliku	Zostanie utworzony
"r+b"	Odczyt i zapis	Bez zmian	Początek pliku (0)	Błąd
"w+b"	Odczyt i zapis	<b>JEST USUWANA</b>	Początek pliku (0)	Zostanie utworzony

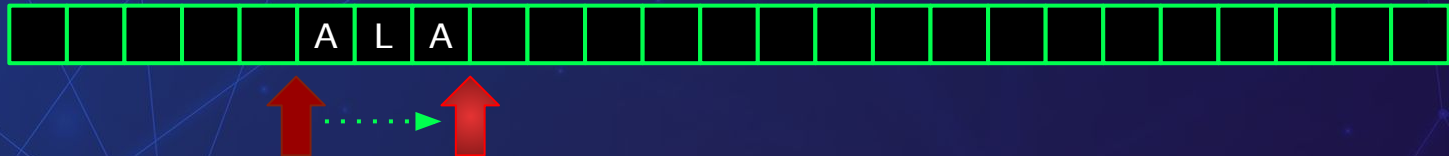
*Hint: Najlepiej zrobić kopię zapasową / eksperymentować na innym pliku*



```
with open("asdf", "r+b") as f:  
    f.seek(5)
```

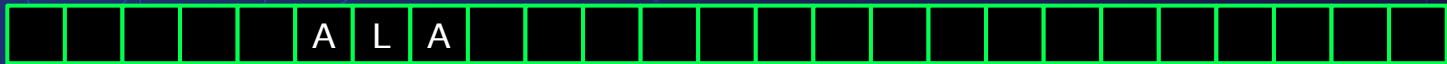
Patching!





```
with open("asdf", "r+b") as f:  
    f.seek(5)  
    f.write(b"ALA")
```

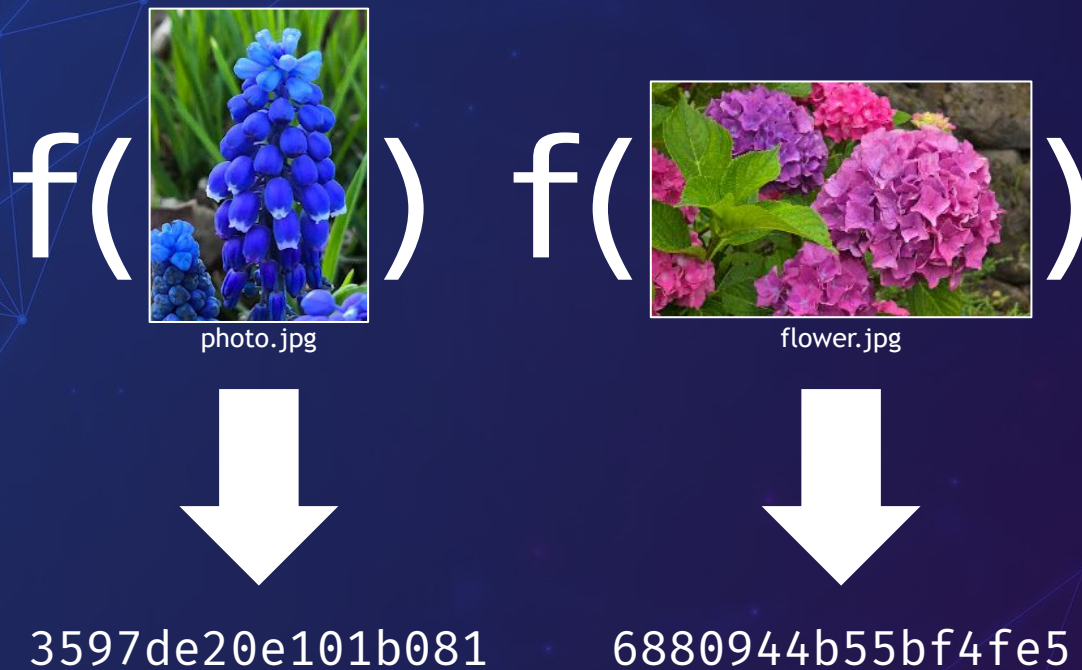
Patching!



```
with open("asdf", "r+b") as f:  
    f.seek(5)  
    f.write(b"ALA")
```

Patching!

# Hash



# Sockety

Jeśli dwie aplikacje chcą ze sobą porozmawiać (TCP)...





Aplikacja "serwer" musi stworzyć gniazdo nasłuchujące  
(Listening Socket)



Aplikacja "klient" musi stworzyć "wtyczkę"  
(po prostu socket)



**Aplikacja**

**Aplikacja**

**Kernel**

**Kernel**

NETWORK

Potem musi nastąpić połączenie...



Które zostanie nawiązane  
(u serwera pojawia się "połączony socket")



<https://docs.python.org/3/library/socket.html#example>

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:  
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)  
    s.bind((HOST, PORT))  
    s.listen(1)  
    conn, addr = s.accept()  
    with conn:  
        print('Connected by', addr)  
        while True:  
            data = conn.recv(1024)  
            if not data: break  
            conn.sendall(data)
```

←Server



Client→

```
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:  
    s.connect((HOST, PORT))  
    s.sendall(b'Hello, world')  
    data = s.recv(1024)  
    print('Received', data)
```



**Let's go!**



*Tu programowanie.*

Jak wolne łącze?  
Popsuty plik  
Haszujemy i patchujemy!

# Podsumowanie i Q&A



sekurak.pl

securITUM



HexArcana



# THANK YOU

## Q&A



hexarcana.pl



sekurak.pl



hexarcana.ch



securitum.pl

**Discord:**

(instrukcje dostaliście na mejla)

**Slajdy/ćwiczenia:**

<https://cwiczenia.hackarcana.pl/>

Menu / Realizacja kodu → **162e276989940185**