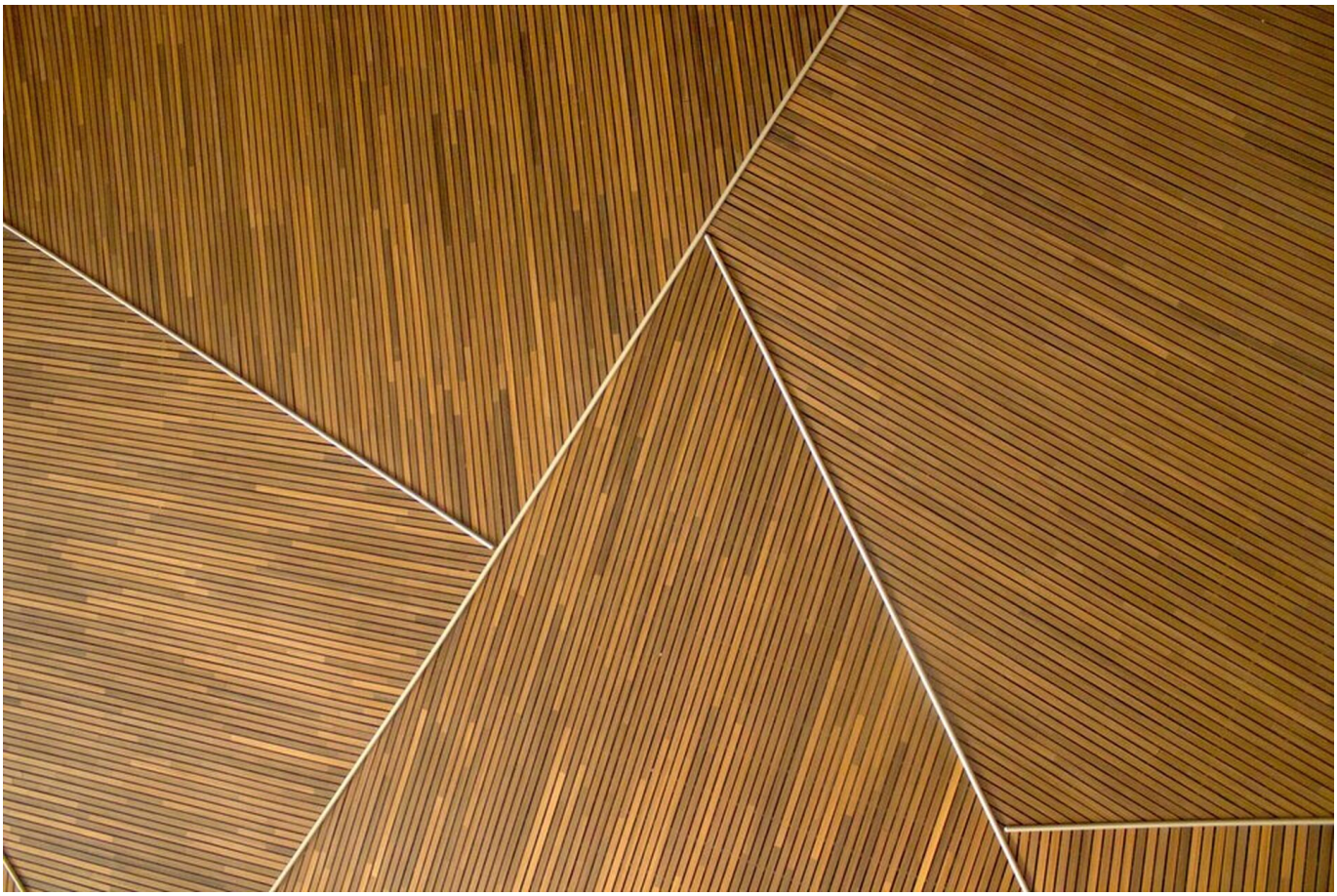


Algorytmy iteracyjne w języku Python

Źródło: Teo Duldulao, domena publiczna.

> W tym e-materiale powtarzamy wiadomości ze szkoły podstawowej.

Jak już wiemy z e-materiału Algorytmy iteracyjne, iteracja to powtarzanie zapisanych w programie instrukcji w pętli z góry określoną ilość razy lub aż do spełnienia określonego warunku. Mechanizm ten jest podstawową operacją wykorzystywaną w programowaniu.

W tym e-materiale zapoznamy się z mechanizmem iteracji w języku Python.

Ciekawi cię, jak wyglądają implementacje w innych językach programowania? Możesz się z nimi zapoznać w dwóch pozostałych e-materiałach z tej serii:

- Algorytmy iteracyjne w języku C++,
- Algorytmy iteracyjne w języku Java.

Więcej zadań? Sięgnij do Algorytmy iteracyjne – zadania maturalne.

> **Twoje cele**
>
> - Przeanalizujesz algorytm znajdowania minimum i maksimum w zbiorze.
> - Prześledzisz różnice pomiędzy pętlami `for` i `while`.
> - Rozwiążesz proste problemy przy pomocy algorytmu iteracyjnego.

Schemat interaktywny

Polecenie 1

Stwórz algorytm obliczania silni za pomocą pętli.

Specyfikacja problemu:

Dane:

- `liczba` – liczba naturalna, dla której będzie liczona silnia

Wynik:

Na standardowym wyjściu program wypisuje wartość silni `liczba!`.

Polecenie 2

Zapoznaj się z filmem. Porównaj z nim swoje rozwiązanie.

Film dostępny pod adresem /preview/resource/RCLz4496bQZ8j

Nagranie filmowe przedstawiające proces stworzenia algorytmu

Kod programu zaprezentowanego w filmie:

Plik o rozmiarze 94.00 B w języku polskim

Polecenie 3

Napisz program w języku Python, realizujący algorytm gry w FizzBuzz dla liczb od 1 do `limit`.

Specyfikacja problemu:

Dane:

- `limit` – liczba naturalna, na której program powinien zakończyć działanie

Wynik:

Na standardowym wyjściu program wypisuje kolejne liczby od 1 do *limit*, zastępując liczby podzielne przez 3 słowem „Fizz”, liczby podzielne przez 5 „Buzz”, natomiast liczby podzielne przez 3 oraz przez 5 „FizzBuzz”.

```
# Tutaj dodaj własny kod. Użyj funkcji
 # print()
```

```
```

Polecenie 4

Porównaj swoje rozwiązanie z filmem.

Film dostępny pod adresem /preview/resource/R1XAihjU3wSLG

Film nawiązujący do treści materiału: Wprowadzenie do iteracji.

Kod programu (tabliczka mnożenia) zaprezentowanego w filmie:

Plik o rozmiarze 259.00 B w języku polskim

Kod programu (FizzBuzz) zaprezentowanego w filmie:

Plik o rozmiarze 294.00 B w języku polskim

Przeczytaj

W języku Python możemy korzystać z dwóch rodzajów pętli: `for` oraz `while`. W pętli `for` podajemy zbiór, po którym będziemy iterować, natomiast w pętli `while` definiujemy warunek – dopóki będzie spełniony, pętla będzie wykonywana.

Składnia pętli `for` wygląda następująco:

```
for (zmienna wewnętrzna) in (zbiór):
    instrukcja
```

Zmienna wewnętrzna to nazwa zmiennej, która będzie przechowywała aktualnie analizowany element ze zbioru. Jeśli zbiorem jest np. lista uczniów, to do zmiennej wewnętrznej przy każdej iteracji będzie przypisywany kolejny uczeń.

Zbiorem może być tablica, lista lub zakres (`range`).

Przykładowa pętla `for`, wypisująca po kolei liczby od 1 do 10 może wyglądać tak:

```
for x in range(1, 11): # zakres od 1 do 11, z wyłączeniem 11
    print(x)
```

Natomiast pętla `while` prezentuje się następująco:

```
while (warunek):
    instrukcja
```

Instrukcje wewnątrz pętli będą wykonywane, dopóki spełniony jest `warunek`.

Pętla `while` wypisująca po kolei liczby od 1 do 10 będzie wyglądała tak:

```
i = 1

while i <= 10:
    print(i)
    i = i + 1
```

> **Dla zainteresowanych**
>
> Pętla może być wykonywana w nieskończoność, do momentu przerwania programu. Aby napisać taką pętlę, wystarczy, że jej warunek cały czas będzie miał wartość True. Na przykład: `while True: print("test")`

Przykładowe zadania

Znajdowanie minimum w zbiorze liczb

Algorytm znajdowania minimum (lub maksimum) ma zastosowanie nie tylko w matematyce, ale np. przy wyborze najniższej ceny produktu czy ucznia z największą liczbą punktów. Korzysta się z niego również w algorytmach sortowania.

Zdefiniujmy najpierw tablicę z liczbami, spośród których będziemy szukać minimum. Utworzymy także zmienną `min`, do której będzie przypisana najmniejsza liczba ze zbioru. Na początek ustawmy jej wartość na pierwszą liczbę z tablicy `liczby`.

```
1 liczby = [10, 3, 8, 15, 2]
2 min = liczby[0]
```

Lista kroków algorytmu:

1. Rozpocznij algorytm.
2. Wczytaj pierwszą liczbę z tablicy.
3. Porównaj wczytaną liczbę ze zmienną `min`.
4. Jeśli wczytana liczba jest mniejsza od zmiennej `min`, to zmiennej `min` przypisz wczytaną liczbę.
5. Jeśli tablica zawiera kolejną liczbę, wczytaj ją i wróć do kroku 3. W przeciwnym razie przejdź do kolejnego kroku.
6. Wyprowadź wynik: zmienną `min`.
7. Zakończ algorytm.

> **Dla zainteresowanych**
>
> Jak napisać program, który będzie jednocześnie znajdował najmniejszą i największą liczbę? W najprostszej wersji wystarczy dodać zmienną `max` i w podobny sposób porównywać liczby. Jednak nie jest to najlepsze rozwiazanie – w jaki sposób można zoptymalizować taki algorytm? Podpowiedź: liczby, które dodajemy do zbioru mniejszych, na pewno nie będą w zbiorze większych liczb.

Słownik

iteracja

czynność powtarzania tej samej operacji w pętli z góry określoną liczbę razy lub aż do spełnienia określonego warunku

pętla

jedna z konstrukcji programowania strukturalnego; pozwala na cykliczne wykonywanie instrukcji określoną liczbę razy, do momentu zajścia pewnych warunków, dla każdego elementu zbioru lub w nieskończoność

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który będzie obliczał iteracyjnie wartość silni. Jego działanie przetestuj dla $11!$.

Specyfikacja problemu:

Dane:

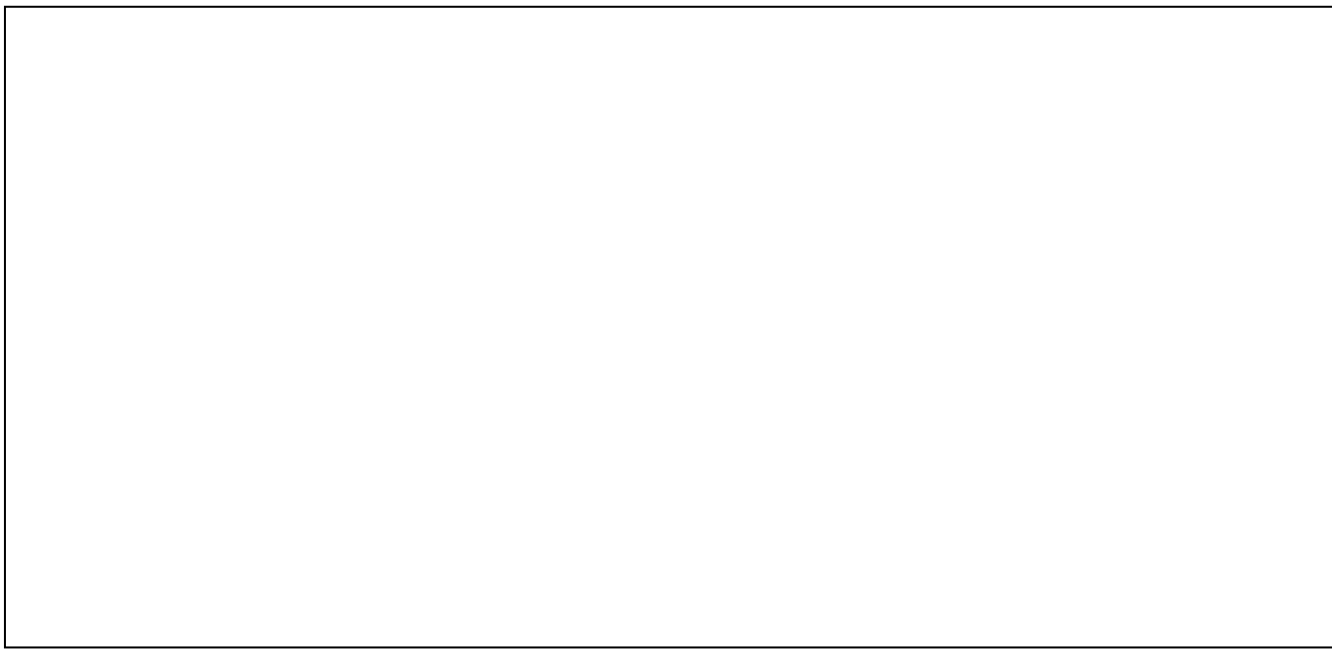
- `podstawa` – liczba naturalna

Wynik:

Na standardowym wyjściu prezentowana jest silnia zadanej liczby.

Twoje zadania

1. Utwórz funkcję `silnia`
2. Zwróć obliczoną silnię



Ćwiczenie 2

Firma *Hula Hop Inc.* jest największym w Europie środkowo-wschodniej producentem obręczy hula-hop. Firma oferuje usługę premium – produkcje zindywidualizowanego hula-hop z wypustkami masującymi. Klient może zamówić hula-hop o promieniu `r` i liczbie wypustek `n`. Ze względu na ograniczenia technologiczne `n` musi być w przedziale $< 5, 50 >$, z kolei `r` w przedziale $< 40, 120 >$. Niedawno firma zaopatrzyła się w nową maszynę, która nie potrafi automatycznie obliczać współrzędnych początków i końców wypustek. Oczekuje wprowadzenia pary liczb: współrzędnej początku i końca wypustek na obręczy koła.

Stałą π możesz pobrać z wywołania `math.pi`.

Specyfikacja problemu:

Dane:

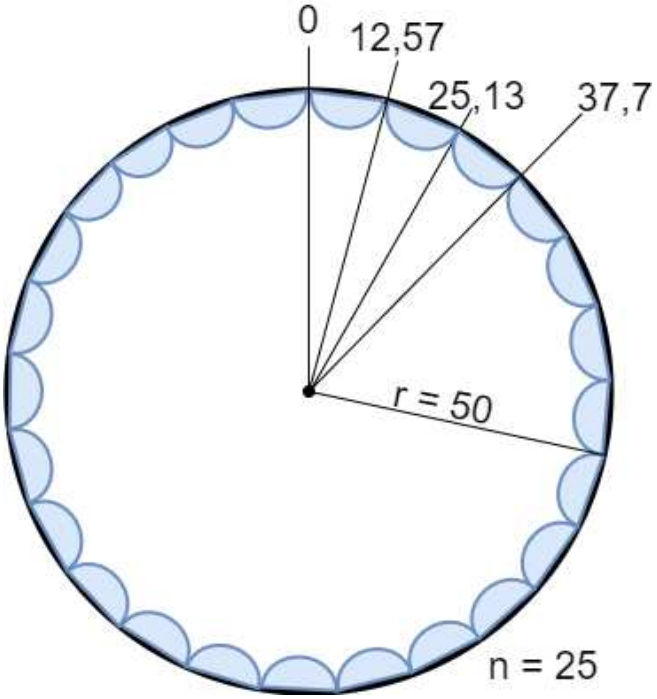
- `n` – liczba wypustek w hula-hop; liczba naturalna z przedziału $< 5, 50 >$
- `r` – promień hula-hop; liczba naturalna z przedziału $< 40, 120 >$

Wynik:

Na standardowym wyjściu program prezentuje współrzędne kolejnych wypustek na okręgu, zgodnie ze specyfikacją maszyny.

Przykład:

Rozważmy jedno z zamówień klienta. Zostało ono przedstawione na rysunku: `n` wynosi 25, z kolei `r` równa się 50. Dla czytelności wartości na rysunku zostały zaokrąglone z dokładnością do dwóch miejsc po przecinku.



Maszyna poprawnie wykona hula-hop, gdy otrzyma następujące dane:

```
0 12.566370614359172
12.566370614359172 25.132741228718345
25.132741228718345 37.69911184307752
37.69911184307752 50.26548245743669
50.26548245743669 62.83185307179586
62.83185307179586 75.39822368615503
75.39822368615503 87.96459430051421
87.96459430051421 100.53096491487338
100.53096491487338 113.09733552923255
113.09733552923255 125.66370614359172
125.66370614359172 138.23007675795088
138.23007675795088 150.79644737231007
150.79644737231007 163.36281798666926
163.36281798666926 175.92918860102841
175.92918860102841 188.49555921538757
188.49555921538757 201.06192982974676
201.06192982974676 213.62830044410595
213.62830044410595 226.1946710584651
226.1946710584651 238.76104167282426
238.76104167282426 251.32741228718345
251.32741228718345 263.89378290154264
263.89378290154264 276.46015351590177
276.46015351590177 289.02652413026095
```

```
289.02652413026095 301.59289474462014
301.59289474462014 314.1592653589793
```

Przygotuj program, który przygotuje parametry dla maszyny dla następujących parametrów: `n = 35`, `r = 80`.

Twoje zadania

1. Program ma wypisywać kolejne pary liczb, będące współrzędnymi pojedynczej wypustki na obręczy hula-hop.

Ćwiczenie 3

Jan Bitek jest początkującym biegaczem. Znalazł w internecie rewolucyjną metodę na rozgrzewkę. Polega ona na tym, że każdy kolejny krok jest dłuższy od poprzedniego o `roznica_kroku`, aż do osiągnięcia `max_dlugosc_kroku`. Wówczas każdy kolejny krok ma długość `max_dlugosc_kroku`. Rozgrzewka kończy się, gdy Jan przebiegnie dystans `koniec_dystans_rozgrzewki`.

Jan chciałby wiedzieć jaki dystans przebiegł w tzw. kroku pomiarowym: 10, 100 i 1000. Jeżeli Jan zakończył rozgrzewkę przed którymś z kroków pomiarowych, wówczas powinniśmy wypisać przy tym kroku informacje o całkowitym pokonanym dystansie. Wszystkie jednostki są ustandaryzowane.

Specyfikacja problemu:

Dane:

- `roznica_kroku` – liczba naturalna dodatnia
- `max_dlugosc_kroku` – liczba naturalna dodatnia
- `koniec_dystans_rozgrzewki` – liczba naturalna dodatnia

Wynik:

Na standardowym wyjściu program prezentuje w kolejnych liniach pokonany przez Jana dystans w 10, 100 i 1000 kroku.

Przykład:

- Początkowa długość kroku $= 1$
- `roznica_kroku` $= 1$
- `max_dlugosc_kroku` $= 100$
- `koniec_dystans_rozgrzewki` $= 100000$

Poprawną odpowiedzią jest:

```
55
5050
95050
```

Sprawdź dystans w krokach pomiarowych dla następujących parametrów:

- początkowa długość kroku $= 1$
- `roznica_kroku` $= 2$
- `max_dlugosc_kroku` $= 200$
- koniec_dystans_rozgrzewki $= 70000$

Twoje zadania

1. Program ma wypisywać pokonany przez Jana dystans w 10, 100 i 1000 kroku.

Dla nauczyciela

Autor: Zespół autorski Contentplus.pl sp. z o.o.
Przedmiot: Informatyka

Temat: Algorytmy iteracyjne w języku Python

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

> 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.

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:
>
> 3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

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):

- Przeanalizujesz algorytm znajdowania minimum i maksimum w zbiorze.
- Prześledzisz różnice pomiędzy pętlami `for` i `while`.
- Rozwiążesz proste problemy przy pomocy algorytmu iteracyjnego.

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;
- oprogramowanie dla języka Python 3 (lub nowszej wersji), w tym PyCharm lub IDLE.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał „Algorytmy iteracyjne w języku Python". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla i odczytuje temat lekcji oraz cele zajęć. Prosi uczniów o sformułowanie kryteriów sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Uczniowie tworzą pytania dotyczące tematu zajęć, na które odpowiedzą w trakcie lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel wyświetla zawartość sekcji „Przeczytaj”. Na forum klasy uczniowie analizują przedstawione w niej rozwiązanie zadania, w którym omówiono krok po kroku algorytm znajdowania minimum w zbiorze liczb.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Schemat interaktywny”. Uczniowie wspólnie zapoznają się z treścią polecenia 1. Następnie w parach przygotowują rozwiązania, które w kolejnym kroku porównują z zaprezentowanym w filmie. Na forum klasy uczniowie omawiają swoje spostrzeżenia i wątpliwości.
3. **Ćwiczenie umiejętności.** Prowadzący zapowiada uczniom, że będą rozwiązywać ćwiczenie nr 1 z sekcji „Sprawdź się”. Każdy z uczniów robi to samodzielnie. Po ustalonym czasie następuje porównanie napisanych kodów podczas wspólnego omówienia rozwiązań.

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 wykonują polecenie nr 3 z sekcji „Schemat interaktywny”. Ich zadaniem jest napisanie programu realizującego algorytm gry w FizzBuzz. Następnie porównują swoje rozwiązanie z przedstawionym w filmie.

Materiały pomocnicze:

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

Wskazówki metodyczne:

- Uczniowie mogą wykorzystać treści w sekcjach: „Schemat interaktywny”, „Przeczytaj”, „Sprawdź się” jako materiał do lekcji powtórkowej.