

Najważniejsze Funkcje i Algorytmy

Python do matury

[darmowy fragment]

Podstawy

1.1 Wypisywanie i wczytywanie tekstu

Aby wypisać tekst lub wartość zmiennej na ekranie stosujemy funkcję *print()*.

```
print("hej!")
```

Aby wczytać dane do programu używamy funkcji *input()*. W poniższym przykładzie tekst podany przez użytkownika zostanie zapisany w zmiennej *x*, a następnie zostanie wyświetlony na ekranie.

```
x = input()  
print(x)
```

1.2 Wczytywanie danych z pliku

Podstawową umiejętnością, bez której nie ukończysz żadnego zadania z programowania, jest wczytywanie danych z pliku tekstowego. Możesz to zrobić na dwa sposoby:

a) otwierając plik, a następnie wykonując operacje na każdej linijce

```
with open("plik.txt") as f:  
    for line in f:  
        print(line.strip()) # <-- tutaj przykładowo wypisujemy każdą linie na ekranie
```

b) otwierając plik, a następnie zapisując wszystkie linijki tekstu do listy

```
lines = []  
with open("plik.txt") as f:  
    lines = f.readlines()  
lines = [x.strip() for x in lines]
```

Zauważ, że w obydwu przypadkach stosujemy także funkcję *strip()*, aby pozbyć się wszystkich zbędnych spacji oraz enterów na końcu linii, które mogłyby utrudniać dalsze operacje na danych.

Operatory

2.1 Operatory matematyczne

W tabelce poniżej znajdują się wszystkie operatory matematyczne przydatne podczas matury.

Operator	Działanie	Przykład	Skrócona forma
+	Dodawanie	$c = a + b$	$d += e$
-	Odejmowanie	$c = a - b$	$d -= e$
*	Mnożenie	$c = a * b$	$d *= e$
/	Dzielenie	$c = a / b$	$d /= e$
%	Reszta z dzielenia	$c = a \% b$	$d \%= e$
**	Potęgowanie	$c = a ** b$	$d **= e$
//	Dzielenie z pominięciem w wyniku liczb po przecinku	$c = a // b$	$d //= e$

Zwróć uwagę, że potęgowanie wykonujemy przy użyciu symboli ******, a nie symbolu **^**.

2.2 Operatory logiczne

W tabelce poniżej znajdują się wszystkie operatory logiczne przydatne podczas matury.

Operator	Działanie	Przykład	Wynik
==	Równa się	$5 == 6$	Fałsz
!=	Nie równa się	$8 != 9$	Prawda
>	Jest większe niż	$7 > 9$	Fałsz
<	Jest mniejsze niż	$2 < 5$	Prawda
>=	Jest większe lub równe	$8 >= 2$	Prawda
<=	Jest mniejsze lub równe	$7 <= 7$	Prawda
in	Sprawdza, czy wartość znajduje się w liście	$2 \text{ in } [1,2,3,4]$	Prawda
and	Zwraca prawdę, jeśli oba warunki są prawdziwe	$5 > 6 \text{ and } 2 > 0$	Fałsz
or	Zwraca prawdę, jeśli co najmniej jeden warunek jest prawdziwy	$5 > 6 \text{ or } 2 > 0$	Prawda
not	Zaprzecza warunkowi (zmienia prawdę w fałsz, a fałsz w prawdę)	$\text{not } 10 < 5$	Prawda

Listy

3.1 Operacje na listach

Założymy, że deklarujemy poniższą tablicę:

```
lista = [3, 2, 5, 7]
```

Podstawową czynnością jest tworzenie pętli przechodzącej przez wszystkie wartości. Poniższy fragment kodu wypisuje po kolej kolejne liczby z listy.

```
for i in lista:  
    print(i)
```

Na listach możemy wykonywać m. in. poniższe operacje:

a) dodawanie wartości na koniec listy:

```
lista.append(7) # <-- w rezultacie otrzymamy [3, 2, 5, 7, 7]
```

b) dołączanie do końca listy wartości z drugiej listy:

```
lista.extend([1, 2, 3]) # <-- otrzymamy [3, 2, 5, 7, 1, 2, 3]
```

c) usuwanie wartości z listy:

```
lista.remove(2) # <-- otrzymamy [3, 5, 7]
```

d) wstawianie wartości (w przykładzie wstawiamy liczbę 4 na miejsce o indeksie 3:)

```
lista.insert(3, 4) # <-- otrzymamy [3, 2, 5, 4, 7]
```

e) sortowanie listy:

```
lista.sort() # <-- sortujemy listę rosnąco. Jeśli dodamy w nawiasie reverse=True,  
lista zostanie posortowana malejąco. Otrzymujemy [2, 3, 5, 7]
```

f) Czyszczenie listy (usuwanie wszystkich wartości):

```
lista.clear()
```

Tekst

4.1 Operacje na tekście

Założymy, że deklarujemy poniższą tablicę:

```
tekst = 'ala ma kota'
```

Na danych tekstowych (typu *string*) możemy wykonywać m. in. poniższe operacje:

a) pozbycie się zbędnych spacji i enterów z przodu i tyłu tekstu:

```
tekst.strip()
```

b) zmiana fragmentu tekstu na inny:

```
tekst.replace('Ala', 'Ola') # <-- w wyniku otrzymamy 'Ola ma kota'
```

c) rozdzielanie tekstu, w miejscu danego znaku (np. spacji):

```
tekst.split(' ') # <-- w wyniku otrzymamy tablice ['Ala', 'ma', 'kota']
```

d) zmiana wielkości liter:

```
tekst.lower() # <-- otrzymamy 'ala ma kota'  
tekst.upper() # <-- otrzymamy 'ALA MA KOTA'
```

e) sprawdzanie, czy tekst zaczyna lub kończy się podanym ciągiem znaków:

```
tekst.startswith('n') # <-- otrzymamy wartość False  
tekst.endswith('a') # <-- otrzymamy wartość True
```

Warto zapamiętać!

Dane typu tekstowego traktowane są jako wielkie listy znaków. Oznacza to, że możemy na przykład tworzyć pętle przechodzące przez wszystkie znaki tekstu, a także możemy sortować litery, tworzące dany tekst.

```
wyraz = 'domek'  
print(sorted(wyraz)) # <-- Otrzymamy tablicę ['d', 'e', 'k', 'm', 'o']
```

Najważniejsze algorytmy

5.1 Algorytm na szukanie NWW i NWD

Najprostszym sposobem na znalezienie NWD jest skorzystanie z poniższej funkcji, której łatwo nauczyć się na pamięć:

```
def nwd(a, b): return nwd(b, a%b) if b else a
```

Aby otrzymać NWW wystarczy pomnożyć liczbę $a * b$, a następnie podzielić je przez ich NWD. Zależność tą przedstawia wzór:

$$NWW(a, b) = \frac{a \cdot b}{NWD(a, b)}$$

5.2 Sprawdzanie czy liczba jest pierwsza

Poniżej przedstawiona jest funkcja, która zwróci wartość **True**, jeśli liczba jest pierwsza lub wartość **False**, jeśli liczba nie jest pierwsza:

```
def czy_pierwsza(n):
    for i in range(2,int(n/2)):
        if (n%i) == 0:
            return False
    return True
```

5.3 Rozkład na czynniki pierwsze

Poniższa funkcja zwraca tablicę z czynnikami tworzącymi daną liczbę:

```
def rozklad(n):
    czynniki = []
    k = 2
    while n != 1:
        while n % k == 0:
            n /= k
            czynniki.append(k)
        k += 1
    return czynniki
```

Na koniec...

Ten PDF jest okrojoną wersją pełnej ściągi, którą wysyłam každemu uczestnikowi mojego kursu "Pythona do Matury". Jeśli chcesz nauczyć się Pythona, poznać ciekawe tricki i sposoby na unikanie najczęstszych błędów lub przypomnieć sobie wszystko, co musisz wiedzieć, aby bezstresowo napisać zadanie z programowania podczas egzaminu, skorzystaj z linku, który zostawiam poniżej. Prowadzi on do strony z kursem. Ponadto, na dole strony znajdziesz również kod zniżkowy -25%.

Miłej nauki!

Febru

Kurs znajdziesz na
febru.me/kurs-pythona

Kod zniżkowy uprawniający do zniżki -25% przy zakupie kursu:

pythonpdf25