

Taylor

E 3011

Jan Böhm

RECETOX

March 14, 2023

Co nás dnes čeká

1 Noprogramování

- My bad
- for⁴
- Tak trochu jiný while

2 Opakování matematiky

Chyby

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

❶ `SyntaxError: expected ':'`

Chyby

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
- 6 `TypeError: type str doesn't define __round__ method`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
- 6 `TypeError: type str doesn't define __round__ method`
- 7 `ModuleNotFoundError: No module named 'E3011'`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
- 6 `TypeError: type str doesn't define __round__ method`
- 7 `ModuleNotFoundError: No module named 'E3011'`
- 8 `ZeroDivisionError: division by zero`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
- 6 `TypeError: type str doesn't define __round__ method`
- 7 `ModuleNotFoundError: No module named 'E3011'`
- 8 `ZeroDivisionError: division by zero`
- 9 `ValueError: math domain error`

Napište program, který při *pokus* jej spustit vypíše následující chybovou hlášku:

- 1 `SyntaxError: expected ':'`
- 2 `SyntaxError: unterminated string literal (detected at line 3)`
- 3 `SyntaxError: '(' was never closed`
- 4 `IndentationError: expected an indented block after 'for' statement on line 3`
- 5 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
- 6 `TypeError: type str doesn't define __round__ method`
- 7 `ModuleNotFoundError: No module named 'E3011'`
- 8 `ZeroDivisionError: division by zero`
- 9 `ValueError: math domain error`
- 10 `NameError: name 'cos' is not defined`

Cyklus I:

O programu níže rozhodněte:

- Kolik operací + proběhně?
- Co vypíše?

```
1 result = 0
2 for i in range(5):
3     result += i
4
5 print(result)
```

Cyklus II:

O programu níže rozhodněte:

- Kolik operací + proběhně?
- Co vypíše?

```
1 result = 0
2 for i in range(5):
3     result += i
4 for j in range(5):
5     result += j
6
7 print(result)
```

Cyklus III:

O programu níže rozhodněte:

- Kolik operací + proběhně?
- Co vypíše?

```
1 result = 0
2 for i in range(5):
3     result += i
4     for j in range(5):
5         result += j
6
7 print(result)
```

Cyklus IV:

O programu níže rozhodněte:

- Kolik operací + proběhně?
- Co vypíše?

```
1 result = 0
2 for i in range(5):
3     for j in range(5):
4         result += i
5         result += j
6
7 print(result)
```


While jinak

```
1 x = 408849
2 while True:
3     x -= sum([int(b) for b in str(decToBin(x))])
4     if x <= 1:
5         break
```

- Co kód dělá?

While jinak

```
1 x = 408849
2 while True:
3     x -= sum([int(b) for b in str(decToBin(x))])
4     if x <= 1:
5         break
```

- Co kód dělá?
- Zastaví se někdy?

While jinak

```
1 x = 408849
2 while True:
3     x -= sum([int(b) for b in str(decToBin(x))])
4     if x <= 1:
5         break
```

- Co kód dělá?
- Zastaví se někdy?
- Zjistěte, kolik iterací proběhne

While jinak

```
1 x = 408849
2 while True:
3     x -= sum([int(b) for b in str(decToBin(x))])
4     if x <= 1:
5         break
```

- Co kód dělá?
- Zastaví se někdy?
- Zjistěte, kolik iterací proběhne
- Přepište kód bez použití konstrukce `while True -- break`

Co nás dnes čeká

1 Neprogramování

- My bad
- for⁴
- Tak trochu jiný while

2 Opakování matematiky



Taylorův rozvoj

Taylorův rozvoj

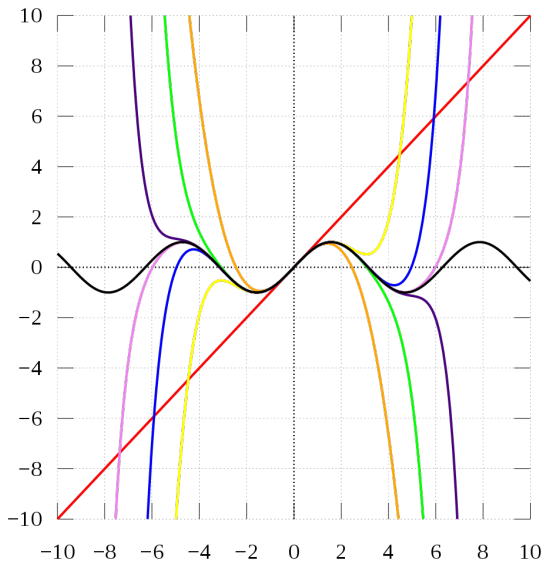
Taylorův polynom slouží k lokálnímu nahrazení funkce "jednodušší" funkcí – polynomem. Dokonce jsme schopni říct, jak takový polynom spočítat: *rozumnou* funkci f můžeme v okolí bodu a aproximovat polynomem

$$f(x) \approx \frac{f(a)}{0!} + \frac{f'(a)}{1!} \cdot (a - x) + \frac{f''(a)}{2!} (a - x)^2 + \dots + \frac{f^{(n)}(a)}{n!} \cdot (a - x)^n$$

neboli

$$f(x) \approx \sum_{k=0}^n \frac{f^{(k)}(a)}{k!} \cdot (a - x)^k$$

Taylorův rozvoj



Aplikace: Eulerovo číslo

Python sám o sobě nemá zabudovanou konstantu e – pokud ji chceme, potřebujeme e povolat z knihovny `math`. Proto si napíšeme vlastní funkci, která bude vracet hodnotu e . Jak na to?

Aplikace: Eulerovo číslo

Python sám o sobě nemá zabudovanou konstantu e – pokud ji chceme, potřebujeme e povolat z knihovny `math`. Proto si napíšeme vlastní funkci, která bude vracet hodnotu e . Jak na to?

Využijeme Taylorův rozvoj funkce $f(x) = e^x$. Funkci rozvineme v bodě 0 a pak položíme $x = 1$.

Aplikace: Eulerovo číslo

Python sám o sobě nemá zabudovanou konstantu e – pokud ji chceme, potřebujeme e povolat z knihovny `math`. Proto si napíšeme vlastní funkci, která bude vracet hodnotu e . Jak na to?

Využijeme Taylorův rozvoj funkce $f(x) = e^x$. Funkci rozvineme v bodě 0 a pak položíme $x = 1$.

Derivace se nám budou hezky počítat:

$$f(x) = f'(x) = f''(x) = \dots = e^x|_{x=0} = 1$$

Aplikace: Eulerovo číslo

Python sám o sobě nemá zabudovanou konstantu e – pokud ji chceme, potřebujeme e povolat z knihovny `math`. Proto si napíšeme vlastní funkci, která bude vracet hodnotu e . Jak na to?

Využijeme Taylorův rozvoj funkce $f(x) = e^x$. Funkci rozvineme v bodě 0 a pak položíme $x = 1$.

Derivace se nám budou hezky počítat:

$$f(x) = f'(x) = f''(x) = \dots = e^x|_{x=0} = 1$$

Rozvoj funkce e^x okolo 0 můžeme jednoduše vyjádřit jako

$$e^x \approx \sum_{k=0}^n \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Aplikace: Eulerovo číslo

Python sám o sobě nemá zabudovanou konstantu e – pokud ji chceme, potřebujeme e povolat z knihovny `math`. Proto si napíšeme vlastní funkci, která bude vracet hodnotu e . Jak na to?

Využijeme Taylorův rozvoj funkce $f(x) = e^x$. Funkci rozvineme v bodě 0 a pak položíme $x = 1$.

Derivace se nám budou hezky počítat:

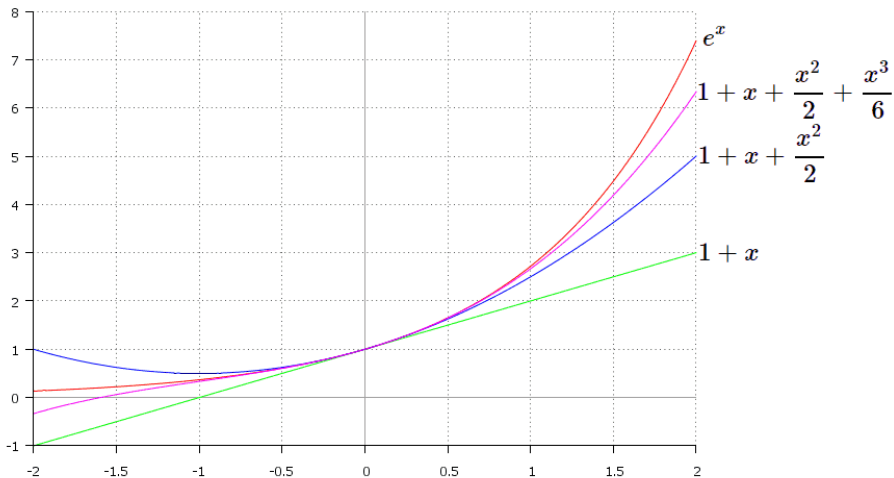
$$f(x) = f'(x) = f''(x) = \dots = e^x|_{x=0} = 1$$

Rozvoj funkce e^x okolo 0 můžeme jednoduše vyjádřit jako

$$e^x \approx \sum_{k=0}^n \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Speciálně pro $x = 1$ je $e^1 = e$ naše hledané číslo:

$$e \approx \sum_{k=0}^n \frac{1}{k!}$$



Stop

Ještě zbývá vyřešit, kdy se sčítáním přestat - jak zvolit n .

Stop

Ještě zbývá vyřešit, kdy se sčítáním přestat - jak zvolit n . Jednoduchý přístup - jakmile začnou přírůstky být příliš malé, přestaneme.

Algorithm 2: How to implement function that returns e

Input: $tol = 0.001$

$ind \leftarrow 0$

$e \leftarrow 0$

$newTerm \leftarrow \frac{1}{ind!}$

while $newTerm \geq tol$ **do**

$result \leftarrow newTerm$

$ind \leftarrow ind + 1$

$newTerm \leftarrow \frac{1}{ind!}$

return $result$
