# Pole polí

## E 3011

Jan Böhm

RECETOX

April 10, 2024

# Co nás dnes čeká

# Pole polí

## Pole polí

Pole může obsahovat jako prvky další pole.

```
food = [["apple", "banana", "cherry"],
        "meat",
        ["beer", "water", "wine"],
        ["beans", "flour", "pasta", "rice"]]

print(food[0])
print(food[-3])
print(food[0][0])
print(food[1][0])
print(food[2][-2])
```

# Manipulace s polem polí

```
food = [["apple", "banana", "cherry"],
        "meat",
        ["beer", "water", "wine"],
        ["beans", "flour", "pasta", "rice"]]

food.append("cheese")
print(food)
food[0].append(["peach","pear"])
print(food)
food[1] = ["fish"]
print(food)
food[2].insert(2, "whiskey")
print(food)
```

# Manipulace s polem polí

```
food = [["apple", "banana", "cherry"],
        "meat",
        ["beer", "water", "wine"],
        ["beans", "flour", "pasta", "rice"]]

del food[1]
print(food)
del food[2][-1]
print(food)
food[1].remove("water")
print(food)
f = food.pop()
print(food)
print(f)
```

# Matice

## Definice matice

Za matici považujeme pole polí čísel obdélníkového tvaru, kde vnořená pole reprezentují řádky matice.

```
P = [[0, 1, 2],
     [5, -3, 0],
     [1, 2, 0],
     [1/2, 0, -3]]

Q = [[0], [1], [-1]]
R = [[0, 1, 3, 2]]
```

# Co nás dnes čeká

1. Pole polí
2. Pomocné funkce

# Pomocné funkce I

Stáhněte si soubor `matrixSupportFunctions.py`. Naleznete v něm kostry kódů pro tvorbu pomocných funkcí.

## `zeros(nrow, ncol)`

Vytvořte funkci `zeros(nrow, ncol)`, která vrací matici plnou nul s `nrow` řádky a `ncol` sloupci.

```
1 def zeros(nrow, ncol):
2     R = [[0 for ???] for ???] # write your code instead of ???
3     return R
4
5 print(zeros(3,4))
6
7 >>> [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

# Pomocné funkce II

## size(M)

Vytvořte funkci `size(M)`, která vrací vektor délky 2 s počtem řádků a počtem sloupců matice.

```
1 def size(M):
2     s = [0, 0] # init vector of length 2
3     s[0] = len(???) # write your code instead of ???
4     s[1] = len(???) # write your code instead of ???
5     return s
6
7 print(size(zeros(3,4)))
8
9 >>> [3, 4]
```

# Pomocné funkce III

## + × checks

- Vytvořte funkci `canAdd(A, B)`, která vrací `True`, pokud lze matice `A, B` sečíst a jinak `False`
- Vytvořte funkci `canMultiply(A,B)` která vrací `True`, pokud lze matice `A, B` vynásobit (v tomto pořadí) a jinak `False`

```
1  def canAdd(A,B):
2      sizeA = size(A)
3      sizeB = size(B)
4      return ??? and ??? # write your code instead of ???
5
6  def canMultiply(A,B):
7      sizeA = size(A)
8      sizeB = size(B)
9      return ??? # write your code instead of ???
10
11 print(canAdd(A,B), canMultiply(A,B))
12 >>> False True
```

# Pomocné funkce IV

## výběr řádku nebo sloupce

- Vytvořte funkci `getRow(A, i)`, která vrací `i`-tý řádek matice `A`
- Vytvořte funkci `getCol(A, j)`, která vrací `j`-tý sloupce matice `A`

Indexujeme od 0.

```
1  def getRow(A, i):
2      return ???
3
4  def getCol(A, j):
5      sizeA = size(A)
6      col = []
7      for i in range(???):
8          col.append(???)
9      return col
10
11 print(getRow(A, 0)) >>> [1, 2, 3]
12 print(getCol(A, 1)) >>> [2, -1, 0, 1]
```

# Pomocné funkce V

## Transponování matice

Vytvořte funkci `transpose(A)`, která vrátí transponovanou matici `A` (vyměněné řádky a sloupce).

```
def transpose(A):
    sizeA = size(A)
    # R is matrix of zeros in size of transposed A
    R = zeros(???) # write your code instead of ???
    # fill entries in R with values in A
    for i in range(???): # write your code instead of ???
        for j in range(???): # write your code instead of ???
            R[i][j] = ??? # write your code instead of ???
    return R

print(transpose(A))
>>> [[1, 1, 4, 1], [2, -1, 0, 1], [3, 0, 2, 1]]
```

# Pomocné funkce VI

## vynechání sloupce

Vytvořte funkci `omittCol(A, j)`, která vrátí matici `A` s vynecháním `j`-tým sloupcem.
K čemu by se nám tato pomocná funkce mohla hodit?

```
1  def omittCol(A, j):
2      sizeA = size(A)
3      # init matrix of correct size
4      B = zeros(???, ???) # write your code instead of ???
5      # for each row in A
6      for i in range(???):
7          a = ???
8          del a[j]
9          B[i] = ???
10     return ???
11
12 print(omittCol(A, 1))
13 >>> [[1, 3], [1, 0], [4, 2], [1, 1]]
```