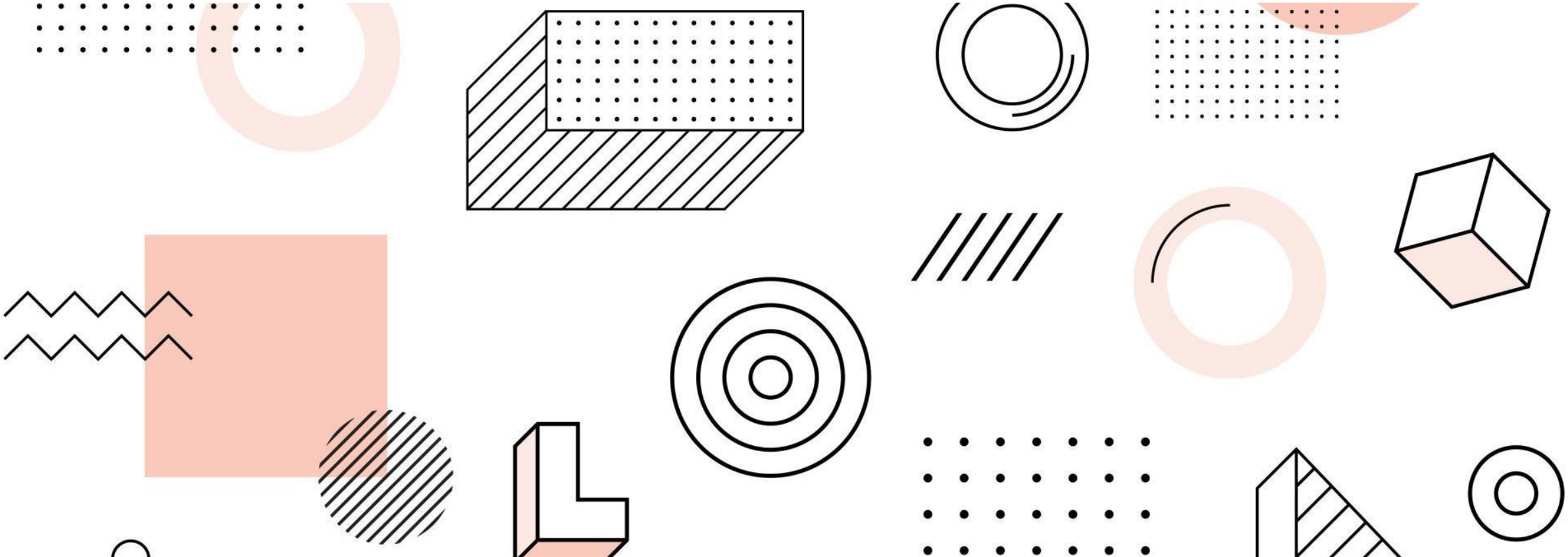
# PYTHON PRO INŽENÝRSKÉ VÝPOČTY

Týden 7. Objekty.



# Tak začneme!

Python je postaven na principech objektově orientovaného programování (OOP). Objektově orientované programování je programovací paradigma, které se zaměřuje na uspořádání kódu do objektů, které zapouzdřují data (atributy) a chování (metody). Zde je několik klíčových charakteristik jazyka Python jako objektově orientovaného jazyka:

- **Objekty**: Vše v Pythonu je objekt, ať už se jedná o jednoduché datové typy, jako jsou celá čísla a řetězce, nebo složitější entity, jako jsou funkce a třídy.

- **Třídy**: Python poskytuje syntaxi pro definování tříd, které slouží jako šablony pro vytváření objektů. Třídy zapouzdřují atributy (data) a metody (funkce), které definují chování objektů. Tím atributy a metody objektu jsou spojeny dohromady a skryty před okolním světem. To umožňuje lepší organizaci kódu a chrání integritu dat objektu.

- **Dědičnost**: Mechanismus pro vytváření nových tříd (podtříd) na základě existujících tříd (nadtříd). Dědičnost umožňuje podtřídám dědit atributy a metody od svých nadtříd, čímž podporuje opakované použití kódu a vytváří hierarchické vztahy.

- **Polymorfismus**: Polymorfismus umožňuje, aby se s objekty různých tříd zacházelo jako s objekty společné nadtřídy. Umožňuje flexibilitu při zacházení s objekty, které sdílejí společné rozhraní nebo chování, což umožňuje opakované použití kódu a modularitu.

# VÝCHOZÍ TŘÍDY

Na začátku kurzu jsme se seznámili s výchozími třídami Pythonu, které umožňuji práci s různými typy dat:

- Celá čísla *(int)*
- Čísla s pohyblivou řádovou čárkou *(float)*
- Řetězec *(string)*
- Seznam (*list*)
- Tuple
- Set
- Bool
- NoneType

```
# Importing the math module
import math


# Using functions from the math module
radius = 5
area = math.pi * math.pow(radius, 2)
print("Area of the circle:", area)
```

# TŘÍDY A OBJEKTY

Objekty se vytvářejí z tříd, které definují data a metody (funkce), jež jsou s objektem spojeny.

```
#Define a class
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def calculate_area(self):
        area = self.length * self.width
        return area


#Create an object
rect = Rectangle(4, 5)


#Access object attributes
print(rect.length)


#Call object method
area = rect.calculate_area()
```

# DĚDIČNOST

Dědičnost je mechanismus pro vytváření nových tříd (podtříd) na základě existujících tříd (nadtříd). Dědičnost umožňuje podtřídám dědit atributy a metody od svých nadtříd, což podporuje opakované použití kódu a vytváří hierarchické vztahy.

```
#Define a class
class Square(Rectangle):
    def __init__(self, length):
        self.length = length
        self.width = length

#Create an object
sq = Square(4)


#Access object attributes
print(sq.length)


#Call object method
area = sq.calculate_area()
```