Maturitní témata 2020/21

Otázka č. 28 – Počítačové sítě a programování

OOP:

- specifické programovací paradigma (základní programovací styl, liší se v pojmech a abstrakcích), které ho odlišilo od původního imperativního (procedurálnímu), zdrojový kód je v OOP přidružen k datům (metody jsou zapouzdřeny v objektech), což umožňuje snadnější přenos kódu mezi různými projekty (abstrakce a zapouzdření), propojení umožnilo zavést dědičnost ale kvůli zjednodušení si vyžádalo zavedení polymorfismu

Třída:

- vzor, podle kterého se objekty vytváří, definuje jejich vlastnosti a schopnosti (chování a reakce na okolí), představuje skupinu objektů které nesou stejné vlastnosti

- třída je předpis pro budoucí objekt (abstrakce) ale samotnou realizací pro řešení analytického problému je objekt jako instance třídy

- objekt přebírá vlastnosti třídy

- třída má deklaraci, výčet atributů a metody

- dědičnost - jedna z vlastností třídy, sdílení kódu, vztah mezi třídami niko mezi jejich objekty, třída ze které se dědí je předek (nadtřída), třída která dědí je potomek (podtřída)

Instance třídy:

- konkrétní datový objekt v paměti odvozený z nějakého vzoru (třídy) používaný v OOP jazycích (Java, C++, C#, atd.), instance třídy (objekt) představuje základní stavební prvek OOP

- objekt který se vytvoří podle třídy se nazývá instance, instance mají stejné rozhraní jako třída podle kterého se vytváří ale navzájem se liší svými daty (atributy)

- např. třída člověk a od ní si vytvoříme instance Karel a Josef, obě instance mají jistě ty samé metody a atributy, jako třída (např. jméno a věk) a metody (jdi_do_prace, pozdrav) ale hodnoty v nich se liší (první instance má atribut jméno Karel a věk 22, druhá Josef a 45)

- komunikace mezi objekty probíhá pomocí předávání zpráv díky čemuž je syntaxe přehledná, zpráva obvykle vypadá takto: příjemce.jméno_metody(parametry), např. karel.pozdrav(sousedka) by mohl způsobit že instance karel pozdraví instanci sousedka

Modifikátory přístupu:

- upravují přístup k metodám a atributům třídy, říkají kdo (které třídy) mohou volat metody (přistupovat k atributům třídy)

- většina běžně používaných objektově orientovaných programovacích jazyků má tři modifikátory přístupu:

- public - lze je volat odkudkoli, libovolné třídy mohou přistupovat k tomuto atributu/metodě
- protected - lze volat pouze z metod stejné či odvozené třídy, některé programovací jazyky nabízejí speciální modifikátor přístupu protected, který označuje metody a vlastnosti které jsou dostupné pouze třídě a jejím potomkům kteří z ní dědí
- private - lze je volat pouze z metod téže třídy, žádná jiná třída nemůže přistupovat

- pokud neuvedeme žádný modifikátor mají přístup jen třídy ve stejném balíčku (package), je to tedy významově podobné jako private ale třídy stejného balíčku mají výjimku (Java)

- pro třídy v jednoduchých projektech většinou používáme s modifikátorem public, výjimečně bez modifikátoru když se jedná o pomocnou třídu kterou nemá být vidět mimo náš balíček

- s atributy bychom měli pracovat jako by byly private, v OOP by třída měla vždy své atributy upravovat sama (zapouzdření)

- pro metody zde využijeme všechny varianty modifikátorů, je třeba rozhodnout jaký přístup pro danou metodu chceme, private pro pomocné metody které nemají být přístupné z jiných tříd, protected pomocné metody u kterých počítáme s tím že by je mohly používat i odvozené třídy, public metody rozhraní kterými komunikujeme s dalšími třídami

```
class Employee:
    def __init__(self,name):
        self.name = name
    def display (self):
        print('The name of employee is:',self.name)


first = Employee('Rushabh')
second = Employee('Dhaval')


second.display()
first.display()
```