# Class Třída

Když přemýšlíme nad strukturou objektově orientovaného programu, je záhodno si nejprve vyčlenit, ze kterých objektů se bude náš program skládat. Jedna z pomůcek je říci si věty jednoduše popisující programovaný systém. Potom z těchto vět vyčlenit podstatná jména a z těch pak vytvořit naše třídy.

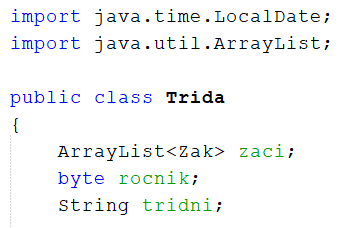
Program Evidence školy má za úkol v jedné škole spravovat jednotlivé třídy. Ve škole pracují učitelé a nepedagogičtí pracovníci. Ve třídách sedí žáci. Žáci jsou zapsáni na určité obory, dle nich pak studují konkrétní předměty. Na základě svých výsledků dostávají známky. Zároveň se jim počítá absence..

Vidíte, že náš program se tímto způsobem docela rychle nafukuje. Máme zde třídy: Skola, Trida, Obor, Predmet, Zak, Pedagogicky\_pracovnik, Nepedagogicky\_pracovnik, Znamkovani, Absence..

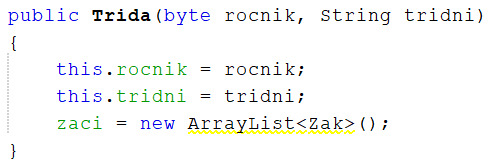
V této učebnici budeme vytvářet pouze jednu dílčí část programu pro evidenci školy. Kdybychom chtěli řešit kompletní program, učební text by se stal dlouhým a také již docela nepřehledným. Základní principy jednoduchých programů však platí i pro ty složitější, proto na konci učebnice byste v případě zájmu měli být schopni celý program dořešit samostatně.

K našemu zjednodušení – předpokládejme, že v celé škole máme pouze jednu třídu. Nikoli ArrayList plný tříd, ale jedinou třídu.

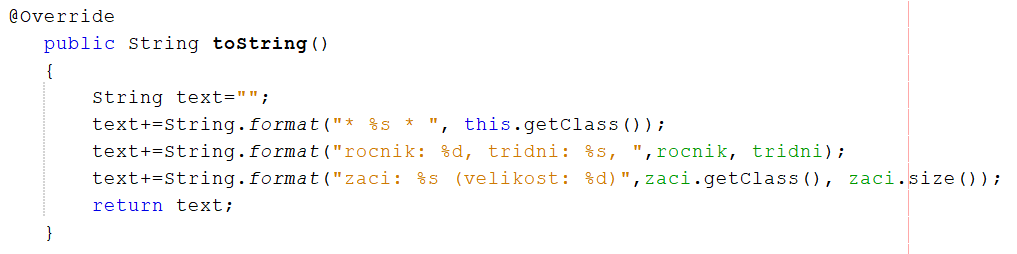
Taková třída obsahuje určitě seznam žáků, dále číslo ročníku a třeba ještě jméno třídního učitele. Jistě Vás napadají další atributy, pro zestručnění programu je však nyní vynecháme.



Třída bude mít na začátku seznam žáků prázdný. Proto postačí jej vytvořit v těle konstruktoru. Ostatní atributy plníme podobně jako ve třídě Zak:

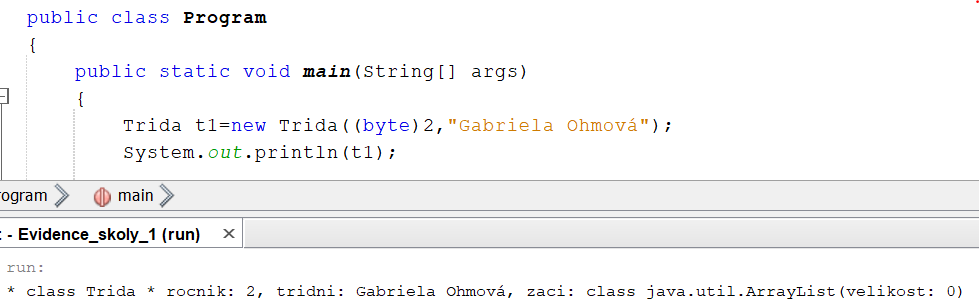


Nyní naprogramujeme metody pro výpis. Začneme známou metodou toString, která bude vypisovat vnitřní stavy atributů třídy Trida. Nezapomeneme použít klíčové slovo override.



Místo kompletního seznamu žáků bude metoda vypisovat pouze informace o třídě ArrayList a aktuální počet prvků, tedy počet studentů. Naše třída zatím žádného studenta neobsahuje.

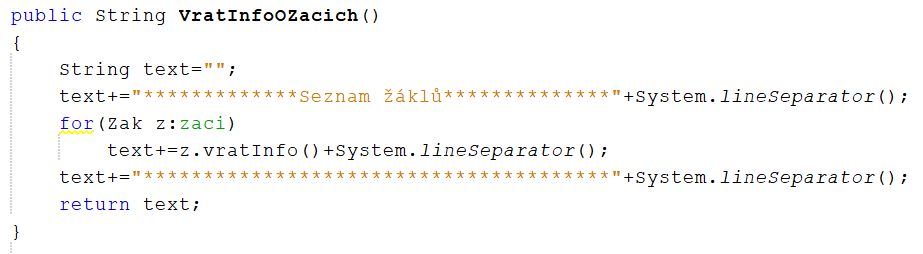
Připomínám, že System.out.println volá metodu toString implicitně. Není tedy potřeba toString  
do kódu psát:



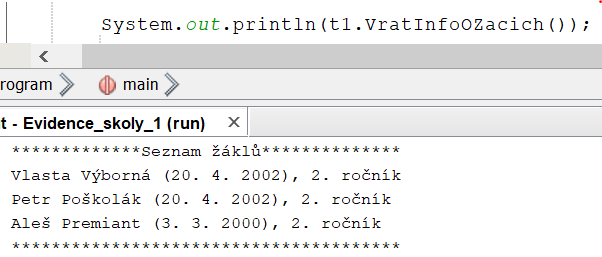
Po další metodě budeme chtít vypsat informace o všech žácích v dané třídě. Pojmenujeme ji VratInfoOZacich. Do metody nic nebude vstupovat, neboť všechny informace pro výpis má třída k dispozici. Výstupem je samozřejmě String.

V metodě procházíme pomocí cyklu **foreach** veškeré žáky v seznamu a voláme na nich jejich vlastní metodu **vratInfo**. Díky provázanosti objektů jsme tedy schopni z jedné třídy volat metodu úplně jiné třídy. Tedy pokud je **public**. Velmi nám to usnadňuje práci, neboť formát pro výpis nemusíme v kódu znovu definovat.

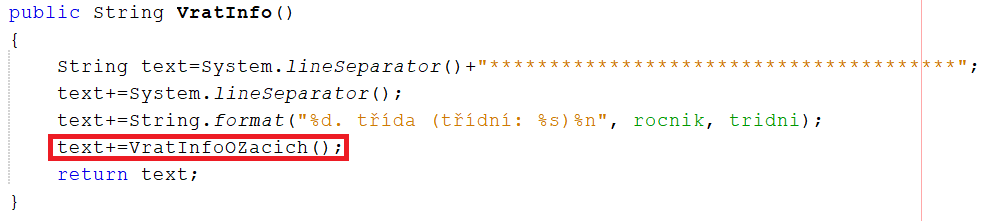
Celý výpis pro přehlednost uzavřeme do textové grafiky:



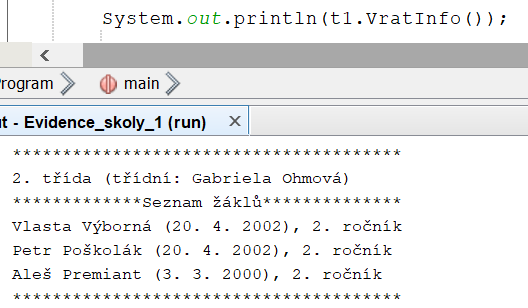
Pro potřeby výpisu jsem do seznamu vložil 3 žáky. Trochu podvádím – metodu pro přidávání žáků budeme řešit teprve v následující kapitole:



Poslední metoda s názvem vratInfo bude vracet kompletní informace o naší třídě, tedy včetně ročníku a třídního. Abychom si programování zjednodušili, zakomponujeme do kódu předchozí metodu VratInfoOZacich.



Výpis je tedy z části shodný s výpisem minulé metody:



V příští kapitole si zapřemýšlíme nad dalšími metodami vhodnými pro práci se třídou. Začneme dříve zmiňovanou metodu pridejNovehoZaka.