# 10. Principy OOP (dědičnost, rozhraní, abstraktní třída)

## Interface

Interface (rozhraní), je speciální druh třídy, která obsahuje soupis metod.

#### public interface DataType {

#### public String getValue();

#### public String getDataType();

#### public void nullValue();

#### public void reset();

#### }

Interface je předpis pro třídy, který jim říká, jaké metody musí implementovat. Když poté třída implementuje dané rozhraní, tak se zavazuje, že tyto metody implementuje (doplní do nich zdrojový kód a tím jim dá funkčnost).

V rozhraní se nesmí deklarovat proměnné a žádná deklarovaná metoda nesmí obsahovat zdrojový kód. Pokud bude nějaký zdrojový kód obsahovat, kompilátor vyhodí chybu.

#### public class Trida implements DataType {

#### public String getValue() {

#### source code;

#### }

#### public String getDataType() {

#### source code;

#### }

#### public void nullValue() {

#### source code;

#### }

#### public void reset() {

#### source code;

#### }

#### }

Jedna třída může implementovat i více rozhraní, což je velká výhoda oproti dědičnosti.

V programu nelze vytvářet instance rozhraní.

## Abstract class

Třída je abstraktní třídou právě tehdy, když obsahuje alespoň jednu abstraktní metodu. Metodu, která neobsahuje žádnou implementaci podobně jako u rozhraní.

Hlavní rozdíl mezi abstraktní třídou a rozhraním je ten, že v abstraktní třídě se můžou deklarovat proměnné a některé metody mohou obsahovat implementaci.

#### public abstract class DataType {

#### private int i;

#### public abstract String getValue();

#### public abstract String getDataType();

#### public abstract void nullValue();

#### public void reset(){

#### source code;

#### }

#### }

Abstraktní metody musí po modifikátoru přístupu obsahovat klíčové slovo abstract. Pokud metoda toto slovo neobsahuje, musí mít nějakou implementaci.

Pokud se má podědit abstraktní třída, provádí se to úplně stejně jako, když se dědí obyčejná třída s tím rozdílem, že se musí implementovat všechny abstraktní metody. Třída může dědit pouze jednu abstraktní třídu.

#### public class Trida extends DataType {

#### public String getValue() {

#### source code;

#### }

#### public String getDataType() {

#### source code;

#### }

#### public void nullValue() {

#### source code;

#### }

#### /\*

#### není třeba implementovat

#### public void reset() {

#### source code;

#### }

#### \*/

#### }

Instanci abstraktní třídy nelze vytvořit.