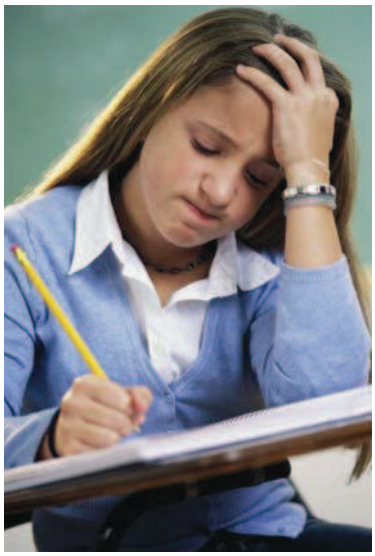
Počítače a programování 2

PPA2



problém, úkol, ...

Algorithms + Data Structures = Programs
Niklaus Wirth – autor jazyka Pascal

program + počítač

Programovací jazyk

Fortran, Algol60, Pascal, C, C++, Java, C#

Datové typy

Příkazy

Program pro tentýž problém můžeme zapsat ve více jazycích!

Jazyk je (pouze) vyjadřovací prostředek!

Datový typ

je množina hodnot a soubor operací nad nimi

čísla, logické hodnoty, znaky, řetězce, …

Příkazy

řídí postup výpočtu

přiřazení
alternativy (if, ...)
opakování / cyklu (while, ...)

Mohu jakýkoliv algoritmus zapsat uvedenými příkazy ?

Počítač

procesor

vykonává instrukce nad daty
v registrech *8, 16, 32, 64, … bitů*

instrukce

přesunu *move*
aritmeticko-logické
řídící *skok – JMP (jump), …*

(hlavní) paměť
obsahuje instrukce + data programu

Jak se příkazy programu transformují na instrukce?

Nemáme instrukce *přiřazení, if, while, ...* !

přiřazení

instrukce MOV

a = a + 1;

```
MOV    R1, A   // přesun hodnoty do registru
INC    R1      // inkrementace
MOV    A, R1   // přesun do paměti
```

příkaz if

instrukce porovnání a podmíněného skoku

```
if (i < 10)
   ... ;
```

```
        CMP    I, 10     // porovnej - compare
        JGE    L         // je-li i větší nebo rovno 10
                         // skoč na L
        ...              // vykonej co je na …
L:
```

příkaz opakování

instrukce porovnání a podmíněného skoku

```
while (i < 10)
   ... ;

L1:      CMP I,10    // porovnej - compare
         JGE  L2     // je-li i větší nebo rovno 10
                     // skoč na L2
         ...         // vykonej co je na …
         JMP  L1
L2:
```

```
program opakuj10krat;
var i:integer;
begin
  i:=0;
  while i<10 do i:=i+1
end.
```

```
                        // i:=0;
      XOR     R1, R1    // operací XOR vynuluj registr R1
      MOV     I, R1     // na paměťové místo pro proměnnou I
                            // přesuň hodnotu v R1

                        // while i<10 do i:=i+1
L1:   CMP     I, 10     // porovnej hodnotu v I a 10
      JGE     L2        // byla-li hodnota I větší nebo rovná 10
                            // skoč na navěstí L2
      MOV     R1, I     // do registru R1 přesuň hodnotu
                            // z paměťového místa pro I
      INC     R1        // zvětši hodnotu registru R1 o 1
      MOV     I, R1     // na paměťové místo pro proměnnou I
                            // přesuň hodnotu v R1
      JMP     L1        // skoč na instrukci s návěstím L1
L2:
```

Jiný stroj – jiné registry, jiné instrukce

jiný překladač, pro tentýž jazyk

Nápad:

- **definujme „univerzální“ instrukce, do kterých budeme překládat náš program – virtuální stroj**
- **pro každý reálný stroj stačí napsat „jenom“ vykonání (interpretaci) těchto instrukcí jeho instrukcemi**

Pozn.: bytecode, JVM

Jaké registry ?

žádné – zásobníkový stroj

operace se vykonávají nad zásobníkovou pamětí

data jsou přístupná v opačném pořadí než byla zapsána

`5*(6+4)`

zapiš 5	**5**	
zapiš 6	**5 6**	
zapiš 4	**5 6 4**	
sečti	**5 10**	**// sčítance jsou 4 a 6**
		// sečti a zapiš
vynásob	**50**	**// součnitelé jsou 5 a 10**
		// vynásob a zapiš

```
class Opakuj10Krat {
  public static void main(String[] args) {
    int i=0;
    while (i<10)
       i++;
  }
}
```

```
                    // i:=0;
 0   iconst_0       // vlož do zásobníku konstantu 0 typu int
 1   istore_1       // ulož hodnotu typu int ze zásobníku do
                       // proměnné 1 (i)
 2   iload_1        // vlož do zásobníku hodnotu typu int
                       // z proměnné 1 (i)

                    // while (i<10) i++;
 3   bipush 10      // vlož do zásobníku hodnotu 10 typu int
 5   if_icmpge 14   // porovnej poslední dvě hodnoty vložené
                       // do zásobníku
                       // a je-li i větší nebo rovno 10 skoč na
                       // index 14
 8   iinc 1 1       // inkrementuj proměnnou 1 (i)
11   goto 2         // skoč na index 2
14   return         // návrat
```

problém

↓ ***vymyslet***

algoritmus + data

↓ ***zvolit jazyk a zapsat***

program

↓ ***přeložit***

počítač

Co je programování ?

datový typ v jazycích

hodnoty a operace odvozené z jednoduchých *matematických abstrakcí* – celé číslo, řetězec, vektor, matice

***život* je bohatší**

celky jsou opsány:

- **několika hodnotami**
- **akcemi, které nad nimi pracují – obecně**
- **více operací – procedura, funkce, metoda**

Pozn.: Volba hodnot a akcí je „programování“ a závisí na řešeném problému

celek: **auto**

hodnoty: **obsah nádrže, spotřeba, …**
akce: **ujetá vzdálenost, …**

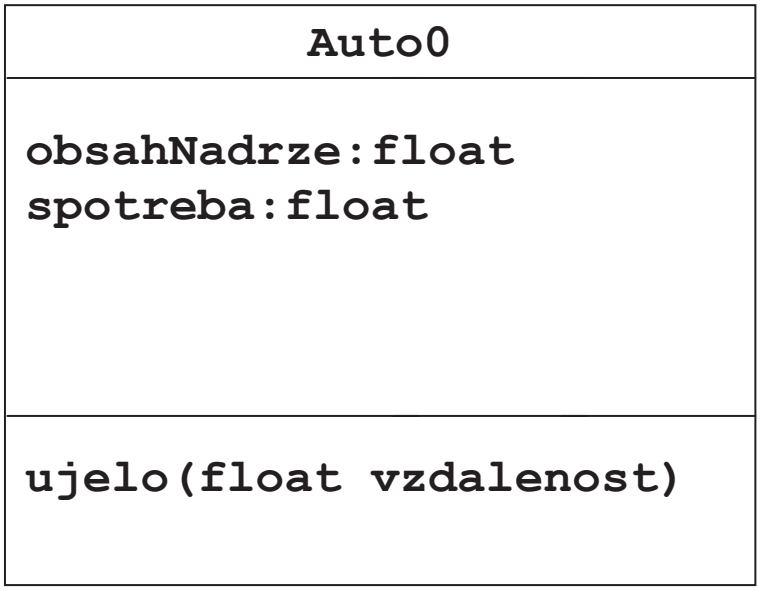
hodnoty a akce jsou definovány třídou

hodnoty - proměnné
akce - metody

```
class Auto0 {
  float obsahNadrze;
  float spotreba;

  void ujelo(float vzdalenost) {
    obsahNadrze = obsahNadrze - vzdalenost /
                    100.0F * spotreba;
  }
}
```

graficky:



opis auta – třída Auto0

konkrétním jednotlivým celkům, říkáme instance třídy / objekty

instanci třídy vytvoří operátor new

```
new Auto0()
```

vrátí referenci / odkaz na objekt

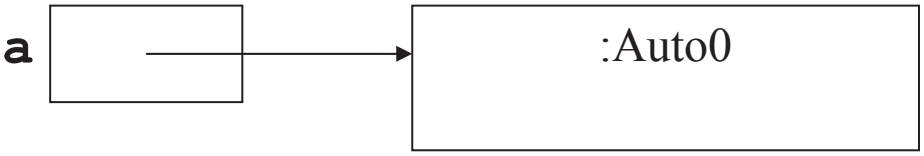
uložíme ji na (referenční) proměnnou na objekty třídy `Auto0`

```
Auto0 a;
a = new Auto0();
```

graficky

```
Auto0 a;
```


```
a = new Auto0();
```



také, jako obvykle

```
Auto0 a = new Auto0();
```

klientská třída / program

práce s instancí(emi) jiné třídy (tříd)

referenční proměnná.*xxx*

xxx proměnná
metoda

```
class MojeAuto {

  public static void main(String[] args) {

    Auto0 a;
    a = new Auto0();
    System.out.println("Spotreba a je
                      "+a.spotreba+" l/100km");
    System.out.println("Obsah nadrze a je
                         "+a.obsahNadrze+" l");

    a.spotreba = 5.0F;
    a.obsahNadrze = 20.0F;
    System.out.println("Spotreba a je
                      "+a.spotreba+" l/100km");
    System.out.println("Obsah nadrze a je
                         "+a.obsahNadrze+" l");

    a.ujelo(200.0F);
    System.out.println("Obsah nadrze a je
                         "+a.obsahNadrze+" l");
  }
}
```

imperativně (procedurálně) – NE !!!

data a metody (procedury) jsou odděleny

```
class Auto0 {
  float obsahNadrze;
  float spotreba;
}

class MojeAuto {

  static void ujelo(Auto0 a, float vzdalenost){
    a.obsahNadrze = a.obsahNadrze -
                  vzdalenost/100.0F*a.spotreba;
  }

  public static void main(String[] args) {
    Auto0 a = new Auto0();
    System.out.println("Spotreba a je "
                       +a.spotreba+" l/100km");
    System.out.println("Obsah nadrze a je "
                          +a.obsahNadrze+" l");

    a.spotreba = 5.0F;
    a.obsahNadrze = 20.0F;
    System.out.println("Spotreba a je "
                       +a.spotreba+" l/100km");
    System.out.println("Obsah nadrze a je "
                          +a.obsahNadrze+" l");

    ujelo(a,200.0F);
    System.out.println("Obsah nadrze a je "
                          +a.obsahNadrze+" l");
  }
}
```

konstruktor

metody, které mají stejný název jako třída, jsou konstruktory
umožňují inicializovat datové členy, členské proměnné

konstruktor je možné přetížit, což umožňuje inicializovat objekt různými způsoby

přetížené metody mají stejné jméno, ale liší se počtem nebo typem nebo pořadím formálních parametrů

Pozn.1: V předcházejícím příkladě se použil tzv. *implicitní* konstruktor `Auto0()`

Pozn.2: Má-li třída konstruktor s parametry, není vytvořen implicitní konstruktor

```
class Auto {
  final float kapacitaNadrze;
  float obsahNadrze;
  float spotreba;

  Auto(float kapacitaNadrze, float spotreba) {
    this.kapacitaNadrze = kapacitaNadrze;
    this.spotreba = spotreba;
  }

  Auto(Auto a) {
    kapacitaNadrze = a.kapacitaNadrze;
    spotreba = a.spotreba;
  }

  Auto() {
    kapacitaNadrze = 24.0F;
    spotreba = 8.0F;
  }

  float doplnNadrz() {
    float doplneni;
    doplneni = kapacitaNadrze - obsahNadrze;
    obsahNadrze = kapacitaNadrze;
    return doplneni;
  }

  float dojezd() {
    float dojede = obsahNadrze / spotreba *
                     100.0F;
    return dojede;
  }

  void ujelo(float vzdalenost) {
    obsahNadrze = obsahNadrze - vzdalenost /
                    100.0F * spotreba;
  }
}
```

```
class MojeAuta {

  public static void main(String[] args) {

    Auto a0 = new Auto();
    System.out.println("Kapacita nadrze a0 je"
                       +a0.kapacitaNadrze+" l");
    System.out.println("Spotreba a0 je "
                       +a0.spotreba+" l/100km");
    System.out.println("Obsah nadrze a0 je "
                       +a0.obsahNadrze+" l");

    Auto a1 = new Auto(35.0F, 5.5F);
    System.out.println("Kapacita nadrze a1 je "
                       +a1.kapacitaNadrze+" l");
    System.out.println("Spotreba a1 je "
                       +a1.spotreba+" l/100km");
    System.out.println("Obsah nadrze a1 je "
                       +a1.obsahNadrze+" l");

    /* promenna kapacitaNadrze je final a
       napriklad zvetsit ji nejde
       a1.kapacitaNadrze = 50.0F;
       System.out.println("Kapacita nadrze a1
       je "+a1.kapacitaNadrze+" l");
    */

    a1.spotreba = 6.0F;  // spotreba a1 stoupla
    System.out.println("Spotreba a1 je "
                       +a1.spotreba+" l/100km");

    Auto a2 = new Auto(45.0F, 10.5F);
    System.out.println("Kapacita nadrze a2 je "
                       +a2.kapacitaNadrze+" l");
    System.out.println("Spotreba a2 je "
                       +a2.spotreba+" l/100km");
    System.out.println("Obsah nadrze a2 je "
                          +a2.obsahNadrze+" l");
```

```
/* dalsi moje auto a3 ma nadrz jako a2 a
   spotrebu jako a1 */
Auto a3 = new Auto(a2.kapacitaNadrze,
                   a1.spotreba);
System.out.println("Kapacita nadrze a3 je "
                  +a3.kapacitaNadrze+" l");
System.out.println("Spotreba a3 je "
                  +a3.spotreba+" l/100km");


/* moje auto a4 je stejne jako a2 */
Auto a4 = new Auto(a2);
System.out.println("Kapacita nadrze a4 je "
                  +a4.kapacitaNadrze+" l");
System.out.println("Spotreba a4 je "
                  +a4.spotreba+" l/100km");

// autu, kteremu rikam a1, nekdy rikam a11
Auto a11 = a1;
System.out.println("Kapacita nadrze a11 je"
                 +a11.kapacitaNadrze+" l");
System.out.println("Spotreba a11 je "
                 +a11.spotreba+" l/100km");

/* zase mu vzrostla spotreba */
a1.spotreba = 7.0F;
System.out.println("Kapacita nadrze a1 je "
                  +a1.kapacitaNadrze+" l");
System.out.println("Spotreba a1 je "
                  +a1.spotreba+" l/100km");
System.out.println("Kapacita nadrze a11 je"
                  +a11.kapacitaNadrze+"l");
System.out.println("Spotreba a11 je "
                  +a11.spotreba+"l/100km");
```

```
    /* autu a1 doplnime nadrz */
    System.out.println("Nacerpali jsme "
                       +a1.doplnNadrz()+ " l");

    /* zjistime dojezd a1 */
    System.out.println("Dojezd je "
                       +a1.dojezd()+" km");

    /* auto a1 ujelo 200 km */
    a1.ujelo(200.0F);

    /* zjistime obsah nadrze */
    System.out.println("Obsah nadrze je "
                       +a1.obsahNadrze+" l");

    /* a muzeme jet kousek dal, doplnit nadrz,
       zmenit spotrebu, zjistit dojezd,... */

    /* prodal jsem auto a4 */
    a4 = null;

    /* prodal jsem auto a1 */
    a1 = null;
    a11 = null;
  }
}
```

objekt parametrem metody objektu téže třídy

```
class Auto {
  final float kapacitaNadrze;
  float obsahNadrze;
  float spotreba;

  Auto(float kapacitaNadrze, float spotreba) {
    this.kapacitaNadrze = kapacitaNadrze;
    this.spotreba = spotreba;
  }

  float doplnNadrz() {
    float doplneni;
    doplneni = kapacitaNadrze - obsahNadrze;
    obsahNadrze = kapacitaNadrze;
    return doplneni;
  }

  float dojezd() {
    float dojede = obsahNadrze / spotreba *
                                          100.0F;
    return dojede;
  }

  void ujelo(float vzdalenost) {
    obsahNadrze = obsahNadrze -
              vzdalenost/100.0F*spotreba;
  }

  float rozdilSpotreby (Auto a) {
    return this.spotreba - a.spotreba;
  }
}
```

```
class MojeAuta {

  public static void main(String[] args) {

    Auto a0 = new Auto(40.0F, 6.5F);
    System.out.println("Kapacita nadrze a0 je
           "+a0.kapacitaNadrze+" l");
    System.out.println("Spotreba a0 je
           "+a0.spotreba+" l/100km");
    System.out.println("Obsah nadrze a0 je
           "+a0.obsahNadrze+" l");

    Auto a1 = new Auto(35.0F, 5.5F);
    System.out.println("Kapacita nadrze a1 je
           "+a1.kapacitaNadrze+" l");
    System.out.println("Spotreba a1 je
           "+a1.spotreba+" l/100km");
    System.out.println("Obsah nadrze a1 je
           "+a1.obsahNadrze+" l");

    float rozdil = a0.rozdilSpotreby(a1);
    System.out.println("Rozdil spotreby a0 a a1
           je "+rozdil+" l");
  }
```

Doplňte si auto o počet míst a počet pasažérů, metody nástup a výstup !