1. *Egyszerű osztályok I.*

Az objektumelvű modellezés önálló egységként azonosítja a megoldandó problémának egy-egy részét: az ahhoz tartozó adatokat, és az ezekkel kapcsolatos tevékenységeket. Az azonos tulajdonságú objektumokat a típusuk jellemezi: ez megmutatja az objektum lehetséges állapotait, és az azzal végezhető műveleteket (metódusokat).
A típus specifikációja sokszor csak indirekt módon olvasható ki a típus megvalósításából: egy állapotot az objektum adattagjai által felvett típusinvariánst kielégítő értékegyüttes reprezentál; egy műveletet az adattagokkal dolgozó típusinvariáns feltételt tartó program implementál.

1. Egy szappanadagolót a tartályában tárolt szappan aktuális mennyisége, az adagolófejének megnyomásakor kiadott mennyiség, és a tartály kapacitása jellemez (mindegyik milliliterben). Egy adagolót meg lehet nyomni, fel lehet tölteni, valamint meg lehet nézni, hogy üres-e.

Típusdefiníció: **Adagoló**

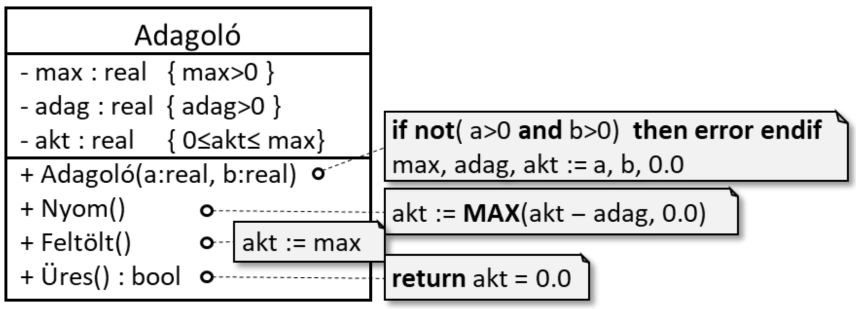
lehetséges adagolók (az adagolók állapotai)	a) a := Nyom(a) // a:Adagoló	a, o := Nyom(a) o : $\mathbb{R}$
	b) a := Feltölt(a) // a:Adagoló	a, o := Feltölt(a, i) i, o : $\mathbb{R}$
	c) l := Üres(a) // l:$\mathbb{L}$	
Inv: max : $\mathbb{R}$ max>0 adag: $\mathbb{R}$ adag>0 akt : $\mathbb{R}$ 0≤akt≤max	a) akt := **MAX**(akt–adag, 0)	akt, o := **MAX**(akt–adag, 0), **MIN**(adag, akt)
	b) akt := max	t := **MIN**(max–akt, i); akt, o := akt+t, i–t
	c) l := akt=0	l := akt=0

Egy adagoló állapotát három nemnegatív valós szám (egy ’a’ adagoló objektum esetén az a.akt, a.max, és a.adag) reprezentál, amelyekre teljesül, hogy 0≤akt≤max, max>0, és adag>0. A műveleteket ezen adattagokkal dolgozó programok implementálják.

Egy művelet megadható elő- utófeltételes specifikációval is. Például a Nyom() művelet esetén:

A = (akt : $\mathbb{R}$, max : $\mathbb{R}$, adag : $\mathbb{R}$)
Ef = ($akt=akt_0 \wedge max=max_0 \wedge adag=adag_0 \wedge Inv$)
Uf = (akt=**MAX**(akt–adag, 0) $\wedge max=max_0 \wedge adag=adag_0 \wedge Inv$)

Osztály: (ez a fenti típusdefiníciós táblázat objektum-orientált leképezése)



Egy objektumot a konstruktorral hozunk létre, amely az invariánsnak megfelelően inicializálja az adattagokat. Ha a kezdőértékeket paraméterezéssel adjuk meg, ellenőrizni kell azok helyességét. (Az ”error” megszakítja a konstruktor futását, az objektum nem jön létre.)

Adagoló a = **new** Adagoló(100.0, 5.0)

Ezután az ’a’ változónévvel hivatkozhatunk az objektumra, meghívhatjuk a metódusait: a.Nyom(), a.Feltölt(), l:=a.Kifogyott(), de az adattagjait nem lehet elérni, hiszen mind privát.

2. Hány UFO található egy űrállomás védelmi körzetében, vagy egyszerűbben: adott pontok közül hány esik bele egy adott gömbbe?

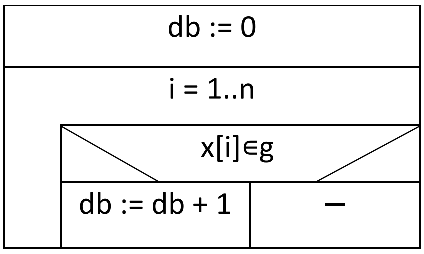
Specifikáció:

A = (x:UFO^n, g:Űrhajó, db:$\mathbb{N}$)
A = (x:Pont^n, g:Gömb, db:$\mathbb{N}$)
Ef = (x=x' $\wedge$ g=g')
Uf = (Ef $\wedge$ db = $\sum_{\substack{i=1..n \\ x[i]\in g}} 1$)

Számlálás

m .. n ~ 1 .. n
felt(i) ~ x[i]∈g

Algoritmus:



(A kódolásnál a felsorolós foreach ciklus is használható.)

Típusdefiníciók:

Gömb

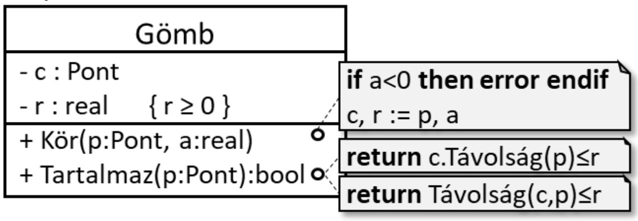
gömbök	l := p∈g // g:Gömb, p:Pont, l:$\mathbb{L}$
c : Pont r : $\mathbb{R}$ Inv: r ≥ 0	l := $\lvert\overline{c,p}\rvert$ ≤ r

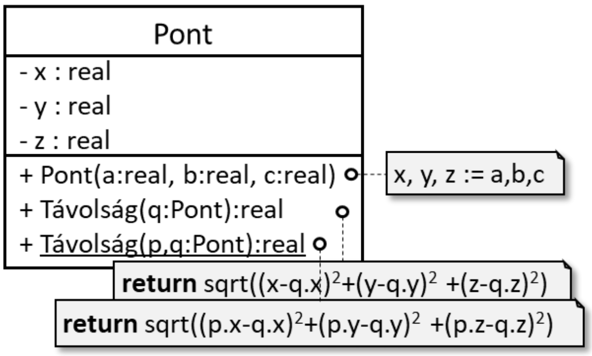
Pont

pontok	d := $\lvert\overline{a,b}\rvert$ // a, b :Pont, d:$\mathbb{R}$
x, y, z :$\mathbb{R}$	d := $\sqrt{(a.x-b.x)^2+(a.y-b.y)^2+(a.z-b.z)^2}$

Az l:=p∈g értékadást l:=Tartalmaz(g, p) vagy az objektum-orientált l:=g.Tartalmaz(p) alakkal is jelölhetjük. A d:=$\lvert\overline{a,b}\rvert$ értékadást d:=Távolság(a,b) vagy d:=a.Távolság(b) alakban is írhatjuk. Ez utóbbi objektum-orientált változat azonban nem tükrözi a távolság szimmetrikus tulajdonságát.

Osztályok:





Egy gömb példányosításához használt konstruktor két paramétert kap (középpont és sugár). Ha a sugár ('a' paraméter) negatív szám, akkor nem hozhatunk létre gömböt (lásd invariáns).

Egy adott pontnak egy másiktól való távolságát implementálhatjuk a Pont objektum metódusaként, vagy a Pont osztály osztályszintű metódusaként, amit az diagramban aláhúzás jelez. Ez utóbbit nem egy pontra kell meghívni (d:=p.Távolság(q)), hanem két ponttal: d:=Távolság(p,q). Amíg az első esetben a p pont koordinátáira közvetlenül hivatkozhatunk az x és y segítségével (eseleg **this**.x és **this**.y), addig második esetben csak p.x és p.y formában.

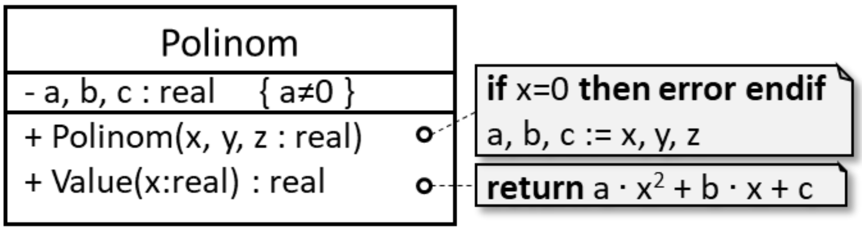
(A tervezés során „felülről-lefelé”, azaz az absztrakttól a konkrét felé haladunk; a kódolásnál viszont „alulról-felfelé”: először az osztályokat, majd a főprogramot készítjük el.)

3. Másodfokú polinommal kapcsolatos feladatok

Típusdefiníció: **Polinom**

másodfokú polinomok // ax^2+bx+c	h := Value(p, x) // p : Polinom, x : $\mathbb{R}$, h : $\mathbb{R}$
a,b,c : $\mathbb{R}$ Inv: $a\neq0$	h := $a\cdot x^2 + b\cdot x + c$

Osztály:



a) Adott egy valós számokat tartalmazó tömb. A tömb mely elemére ad egy adott másodfokú polinom maximális helyettesítési értéket?

Specifikáció:

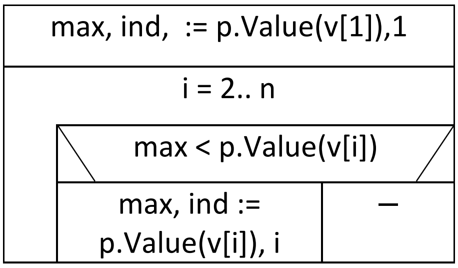
A = (p : Polinom, v : $\mathbb{R}^n$, ind : $\mathbb{N}$, max : $\mathbb{R}$)

Ef = (p=p' $\wedge$ v=v')

Uf = (Ef $\wedge$ (max, ind) =

$\mathrm{MAX}_{i=1..n}$ p.Value(v[i])

Algoritmus:



i: $\mathbb{N}$

b) Adott egy valós számokat tartalmazó tömb. Vajon a tömb minden elemére pozitív-e egy adott másodfokú polinom helyettesítési értéke?

Specifikáció:

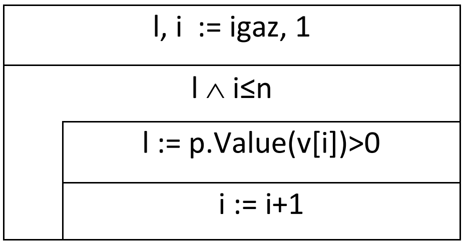
A = (p : Polinom, v : $\mathbb{R}^n$, l : $\mathbb{L}$)

Ef = (p=p' $\wedge$ v=v')

Uf = (Ef $\wedge$

(l, _) = $\forall\mathrm{SEARCH}_{i=1..n}$ p.Value(v[i])>0

Algoritmus:



i: $\mathbb{N}$