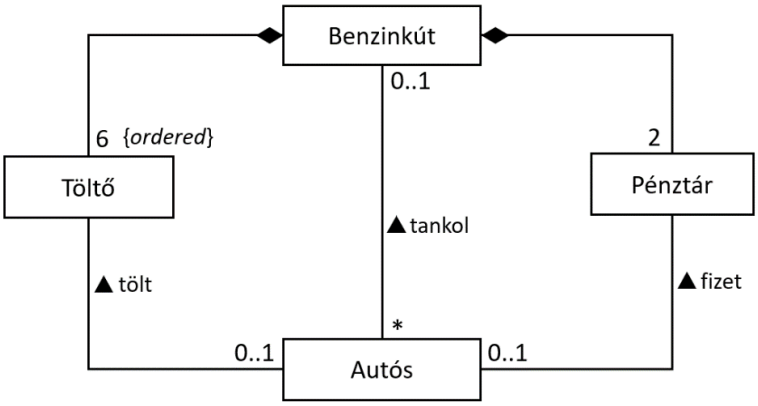
*9. Az 1-n kapcsolatok gyűjteményeinek feldolgozása*

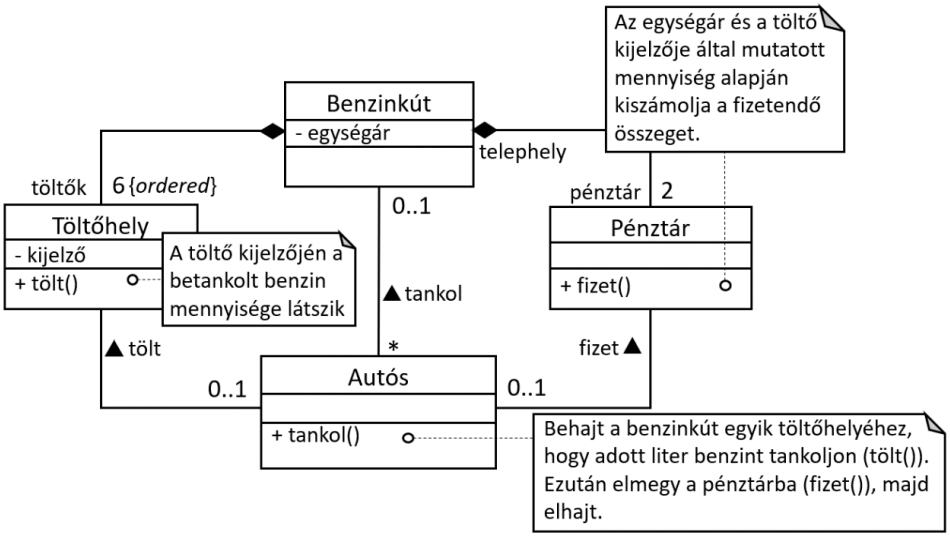
Témakör: Most a problémák osztálydiagramját a metódusaik részletes definíciójával együtt kell megadni. Egy metódus sokszor egy '1 - *' kapcsolat mentén kialakult gyűjteményt dolgoz fel.

1. Egy benzinkúton pontosan 6 töltőhely és 2 pénztár található. Egy autós eldönti hányas töltőhelyre akar beállni (ez a benzinkútnak küldött üzenet, amely ellenőrizni, hogy van-e adott sorszámú töltőhely), miután beállt adott mennyiségű benzint tölt a tankjába, majd a pénztárnál fizet (a pénztár kérdezi le az autós által megadott töltőhelytől a fogyasztott mennyiséget, kiszámítja az árat, lenullázza a töltőhely mérőóráját), és végül távozik. Egy autós csak egy töltőhelyet és egy pénztárt használ. Szimuláljuk azt a helyzetet, amikor egy autós tankol! (Ezt a feladatot később az előadáson újra megoldom több autós párhuzamos viselkedésével, valamint várakozással a töltőhelyek előtt.)

Először csak a szerkezetét tervezzük meg az osztálydiagramnak:

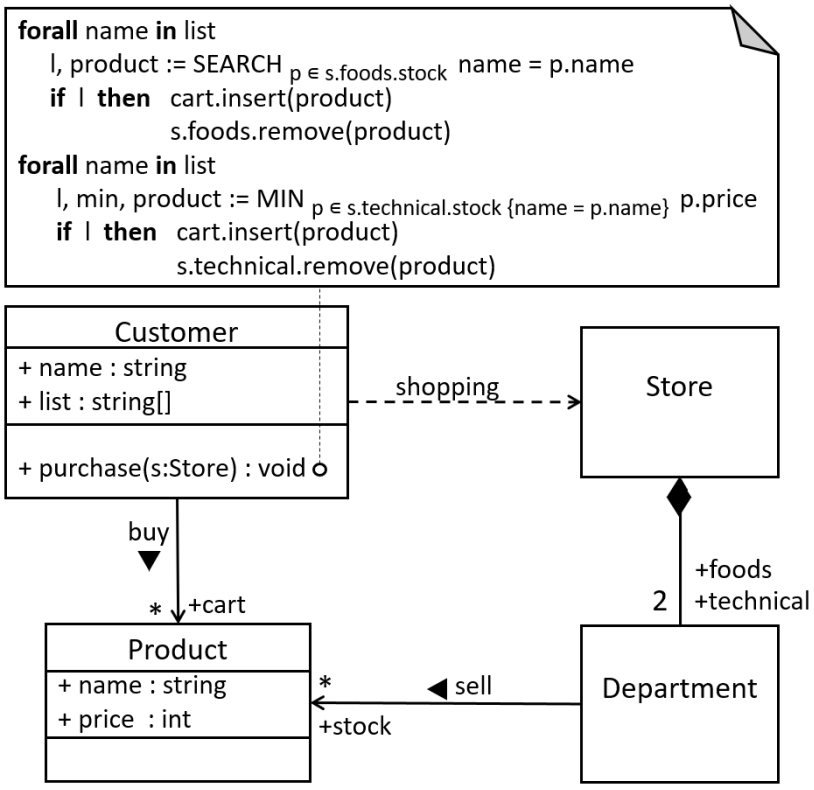



A tankol() metódus átgondolása során további metódusokat és adattagokat veszünk fel.




Dönteni kell még (de ez már a házi feladat) a szerepnevek láthatóságáról, tulajdonosáról, definiálni kell gettereket és settereket, illetve bevezethetünk a jobb áttekinthetőség érdekében segédmetódusokat is (például amelyik adott sorszám alapján hivatkozást ad egy töltőhelyhez, hogy meg lehessen hívni annak tölt() metódusát; illetve kellenek getterek és setterek privát adattagokhoz.)

2. Egy kisvárosi üzlet élelmiszer részlegből és műszaki részlegből áll. A vásárlók egy bevásárlólistával jönnek, amely azon termékek neveit tartalmazza, amit megvennének. Az üzletben a listájukon szereplő termékeket keresik: először az élelmiszer részlegen nézik végig a teljes bevásárlólistát, és a megtalált termékeket magukhoz veszik (beteszik a kosarukba), majd a műszaki részlegen ezt megismétlik, de megfontoltabban: ha egy (a bevásárlólistán szereplő) áruból több is van a részlegen, akkor a legolcsóbbat választják. Szimuláljuk azt a helyzetet, ahogy egy vásárló megveszi az általa szükségesnek vélt termékeket!

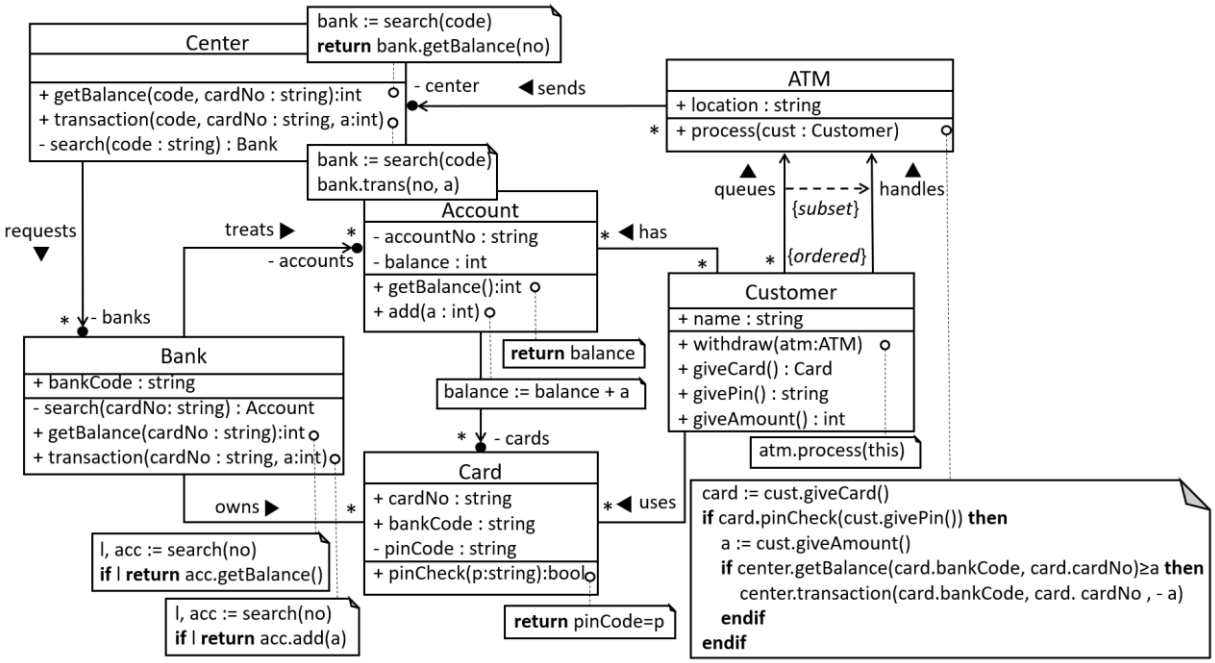



A feladatleírásban szereplő vásárlói magatartást a Customer purchase() metódusa írja le: egy lineáris keresést és egy feltételes minimumkeresést tartalmaz.

Feltesszük, hogy minden gyűjtemény (collection(Item)) – ebben a feladatban ilyen a vásárló kosara (cart), és ezért Item = Product, valamint a két részleg készlete (foods.stock és technical.stock) – felsorolható, továbbá rendelkezik egy beilleszt (insert()), egy eltávolít (remove()), és egy mindent töröl (clear()) metódussal.

(Vigyázat! Háromféle „name” változó van: a Customer adattagja, a Product adattagja, és a purchase()-ben használt forall ciklusok lokális változói.)

3. Egy ATM automatánál sorban állnak az ügyfelek, hogy pénzt vehessenek fel. Az ügyfelek rendelkeznek bankkártyákkal, amikhez tartozik egy PIN kód, valamint egy bankszámla. Az ügyfelek sorban vehetnek fel egy adott összeget az automatából a bankkártyájuk, valamint a kódjuk megadásával. Ha a kód megegyezik a kártya kódjával, akkor az automata kiadja az összeget, feltéve, hogy az összeget levonva az egyenlegből az továbbra is pozitív marad. Ennek megállapításához az automata egy központon keresztül a kártya adatainak megadásával lekérheti az ügyfél egyenlegét, illetve elküldhet a banknak egy jelentést a lebonyolított tranzakcióról, amely alapján a bank leveszi az összeget az ügyfél számlájáról. Szimuláljuk azt a helyzetet, amikor egy ügyfél pént vesz fel egy automatából!



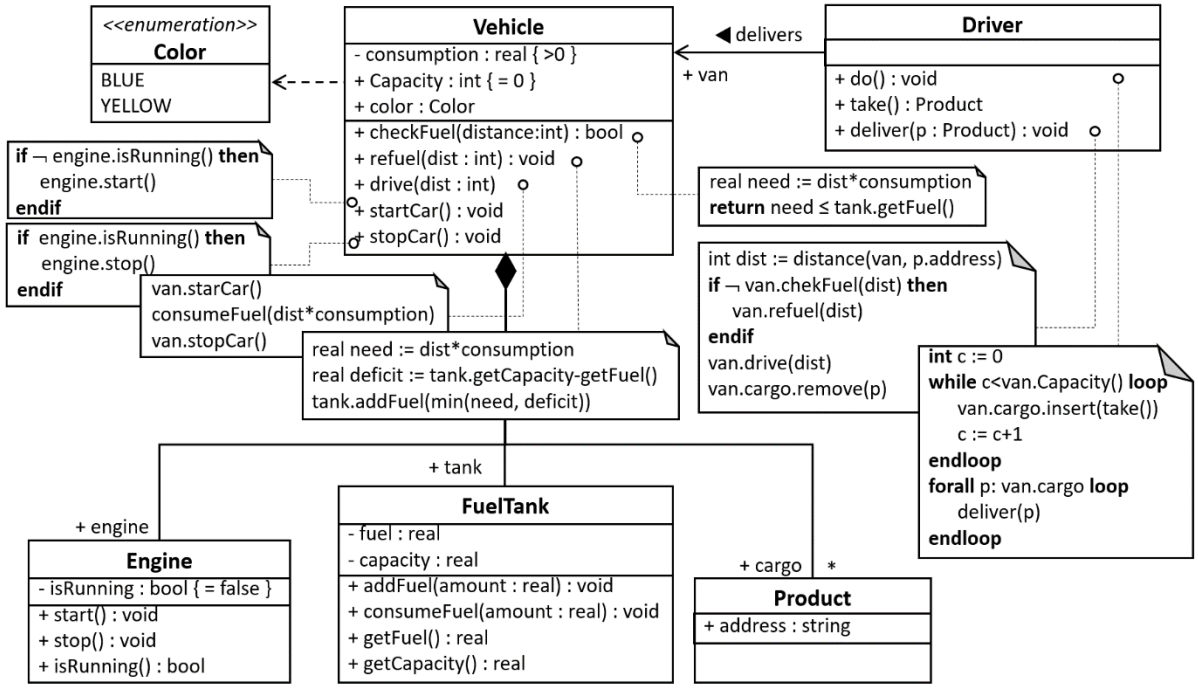

A megoldás az atm.process() hívással indul, amelyet a soronkövetkező ügyfél withdraw() metódusa hív meg. A folyamat az alábbi:

az automata elkéri az ügyféltől a kártyáját
bekéri a pinkódot
ha a megadott pinkód megegyezik a kártya pinkódjával, akkor
    bekéri a felveendő összeget
    ellenőrzi, hogy az ügyfél kérése kielégíthető-e
    ha az igény nem nagyobb a számlaegyenlegnél, akkor
        értesíti a bankot a tranzakciórók, aki levonja az összeget az ügyfél számlájáról.

A pénzfelvétel folyamata a sends, requests, treats, owns asszociációk mentén valósul meg, ezért itt érdemes szerepnevekkel és azok tulajdonosának megjelölésével segíteni a navigációt. Ezen navgációs úton fekvő osztályokba pedig gettereket és egyéb segéd metódusokat érdemes felvenni. A többi asszociáció (has, uses, queues) kevésbé fontos.

Az giveXxx() metódusok a felhasználtól kérnek be adatokat: kártyát, a pin kódot, az összeget. Ehhez maga az ATM nyújt felületet, amelyet nem részletezünk. . A Card pinCheck() metódusa a pinkód ellenőrzését végzi. A Center és a Bank getBalance() és transaction() metódusai a bank felé küldött kéréseket szolgálják ki, de végső soron ezek az Account osztály getBalance() és add() metódusait aktíválják majd

4. Egy csomag kiszállító futár a termékek kiszállítása során bepakol a járművébe annyi terméket, amely belefér (a jármű kapacitását figyelembe véve). A termékekre ráírták a kiszállítás címét, amelyből kiszámolható, hogy ez a cím milyen távolságra van a futár aktuális tartózkodási helyétől (km-ben). A szállítás címei minden esetben benzinkutak (az ott lévő PickPack pontok), így minden állomáson lehet tankolni is. A futárnak a járműbe bepakolt összes terméket ki kell szállítani, aminek menete a következő: először ellenőrzi, hogy a benzinszint a következő kiszállításhoz elegendő-e. Amennyiben nem, előbb tankol; egyébként elindítja a motort, elmegy a címzetthez (aminek eredményeképp csökken a benzinszint a fogyasztás szerint), leállítja a motort, majd kipakolja a terméket. Egy futár járműve kétféle színű lehet: kék és sárga. A járműnek van motorja, rakodótere és egy benzintartálya. A motort el lehet indítani és le lehet állítani. A tartályba a maximális benzinszint figyelembe vételével tankolhatunk. A járműnek ismert a benzinfogyasztása, ami liter/km mértékegységben van adva.




Először az osztályok születnek meg a legszükségesebb adattagokkal és ezek gettereivel, settereivel. Az osztályok asszociációi ekkor még csak kezdetleges formában tűnnek fel a diagramban. Ezt követően a Driver do() metódusából kiindulva kell a futár tevékenységét leírni úgy, hogy ezt szétosztjuk különböző metódusokba. Ennek során dönteni kell, hogy egy metódus milyen lépésekből álljon, ehhez melyik osztályban kell felvenni olyan metódusokatt, amit jól lehetne használni. Ekkor alakulnak ki az asszociációk navigálási irányai, és szerepnevei.