**Údajové štruktúry – semestrálna práca**

2016

**Informačný systém pre sklad minerálnych vôd**

**Tomáš Illo**

**Zadanie**

Cieľom semestrálnej práce je vytvoriť informačný systém pre veľkosklad minerálnych vôd. Veľkosklad má dodávateľov, od ktorých odoberá minerálne vody, a zákazníkov (predajne), do ktorých ich dodáva. Predajne sú usporiadané do k zón, pričom platí, že predajne v zóne 1 sú najbližšie k veľkoskladu, zatiaľ čo predajne v zóne k sú od neho najďalej.

Systém musí umožniť

* **Pridanie nového dodávateľa**. Dodávateľ je charakterizovaný obchodným názvom a adresou sídla. Obchodný názov je unikátny.
* **Pridanie nového zákazníka**, ktorý je charakterizovaný obchodným názvom a adresou centrály. Každý zákazník môže mať viacero predajní, ktoré sa odlišujú svojou adresou.
* **Pridanie novej predajne konkrétneho zákazníka**. Pri pridaní predajne je nutné zadať zónu, v ktorej sa nachádza.
* **Pridanie nového typu minerálnej vody**. Typ minerálnej vody je jednoznačne identifikovateľný na základe EAN kódu. Okrem neho má nasledujúce atribúty: názov a dodávateľ.
* **Zaevidovanie novej dodávky minerálnych vôd**, ktorá príde od dodávateľa. Dodávka má nasledujúce parametre: typ minerálnej vody, množstvo (počet kusov), dátum plnenia.
* **Zaevidovanie požiadavky od zákazníka**. Táto obsahuje typ minerálnych vôd, príslušné množstvá, predajňu a dátum, kedy musí byť tovar doručený na predajňu. V prípade, že dátum doručenia je menší ako „aktuálny dátum + 2 dni“, požiadavka sa nezaeviduje.
* **Kontrola požiadaviek od zákazníkov a množstva vôd na sklade**. Po vykonaní tejto operácie sa vypíše zoznam tých minerálnych vôd, ktorých množstvo na sklade je menšie ako je množstvo požadované zákazníkmi. Údaje musia byť vypísané v abecednom poradí podľa názvu dodávateľa a názvu minerálnej vody. Vo výpise musí byť aj chýbajúce množstvo jednotlivých minerálnych vôd.
* **Vyskladnenie.** V tomto prípade je nutné pripraviť kontajner minerálnych vôd, ktoré musia byť doručené nasledujúci deň. Doručovanie prebieha pomocou auta, ktoré rozváža vody k jednotlivým zákazníkom v poradí podľa ich zonácie, t.j. na dne kontajnera musia byť vody ktoré, je potrebné doručiť do predajní v zóne k, nad nimi sa nachádzajú vody, ktoré sa doručia do predajní v zóne k -1,..., na vrchu sú vody, ktoré sa doručia do predajní v zóne 1. Pri vyskladnení je ďalej nutné zohľadniť dátum plnenia, t.j. vyskladňujú sa minerálne vody s najskorším dátumom plnenia. V prípade požiadavky, ktorá sa nedá zrealizovať (t.j. požaduje sa viac minerálnej vody, ako je jej množstvo na sklade), sa táto označí za nezrealizovanú. Vyskladnené minerálne vody je nutné odpísať zo skladu.
* **Odovzdanie tovaru zákazníkovi**. Z kontajnera, ktorý sa nachádza v aute, sa odoberie najvrchnejšia položka. Týmto dôjde k zrealizovaniu konkrétnej požiadavky, t.j. v tomto okamžiku sa požiadavka označí za zrealizovanú.
* **Vypísanie aktuálneho množstva jednotlivých typov minerálnych vôd na sklade**. Údaje musia byť vypísané v abecednom poradí podľa názvu dodávateľa a názvu minerálnej vody.
* **Vyhľadanie odberateľa**, ktorý za dané časové obdobie:
  + odobral najväčšie množstvo konkrétneho typu minerálnej vody,
  + odobral najväčšie množstvo minerálnych vôd od konkrétneho dodávateľa.
* Vypísanie všetkých požiadaviek od všetkých odberateľov, ktoré sa nepodarilo zrealizovať.
* **Vyhľadanie dodávateľa**, po ktorého minerálnych vodách bol za dané časové obdobie najväčší dopyt.
* **Uložiť do a načítať zo súboru** aktuálny stav evidencie.

# Rozbor

Cieľom semestrálnej práce je vytvoriť program, ktorý bude schopný reagovať na každú z vyššie spomenutých požiadaviek.

Počas riešenia semestrálnej práce nastalo niekoľko problémov pre splnenie všetkých požiadaviek zadania. Najväčším problémom bolo vyskladnenie. Odpisovanie jednotlivých položiek zo skladu. Ďalším z problémov, bolo vyhľadanie odberateľa ktorý spĺňal konkrétne podmienky za dané časové obdobie.

Vstup od užívateľa je riešený pomocou príkazov, ktoré sú popísané nižšie. To by malo zabezpečiť dynamický vstup od užívateľa s čo najmenšou závislosťou na predchádzajúcom príkaze . Tiež je možné dáta a príkazy načítať zo súboru a zároveň uložiť do súboru vo formáte, pripravenom na následné načítanie.

# Návrh aplikácie

* **ArrayList**

Údajovú štruktúru ArrayList som použil vo väčšine zoznamov, kde som potreboval vkladať na koniec zoznamu a prehľadávať zoznam bez jeho zmeny.

Výpočtová zložitosť

Vhodnosť z pohľadu výpočtovej zložitosti

konštruktor -> O(1)

add -> O(1)

insert -> O(N)

delete -> O(N)

deleteFromIndex -> O(N)

indexOf -> O(N)

get -> O(1)

size -> O(1)

clear -> O(N)

iterator -> O(N)

* **LinkedList**

LinkedList som použil pre zoznam dodávok. V tomto prípade je potrebné mazať prvky z rôznych pozícii a na to sa táto údajová štruktúra perfektne hodí.

Výpočtová zložitosť

Vhodnosť z pohľadu výpočtovej zložitosti

konštruktor -> O(1)

add -> O(1)

insert -> O(N)

deleteFirst -> O(1)

deleteLast -> O(N)

deleteFromIndex -> O(N)

indexOf -> O(N)

getFirst -> O(1)

getLast -> O(N)

getAt -> O(N)

size -> O(1)

clear -> O(1)

iterator -> O(N)

* **Prioritný front**

Prioritný front používam tam, kde potrebujem niečo zoradiť do správneho poradia podľa časti, alebo prvku samotného. Napríklad pri vyskladnení, kde potrebujem objednávky zoradiť podľa zóny predajne do ktorej ich treba doručiť. Ďalej túto údajovú štruktúru používam na zoradenie podľa dátumu.

Výpočtová zložitosť

Vhodnosť z pohľadu výpočtovej zložitosti

konštruktor -> O(1)

push -> O(log2(N))

pop -> O(1)

peek -> O(1)

isEmpty -> O(1)

size -> O(1)

clear -> O(1)

* **Stack**

Stack bol pre mňa najvhodnejší pre kontajner, auto na prevoz. Do auta som uložil objednávky podľa zóny predajne a pri každej predajni stačilo zobrať najvrchnejšiu položku, teda objednávku s najnižšou zónou predajne.

Výpočtová zložitosť

Vhodnosť z pohľadu výpočtovej zložitosti

konštruktor -> O(1)

push -> O(1)

peek -> O(1)

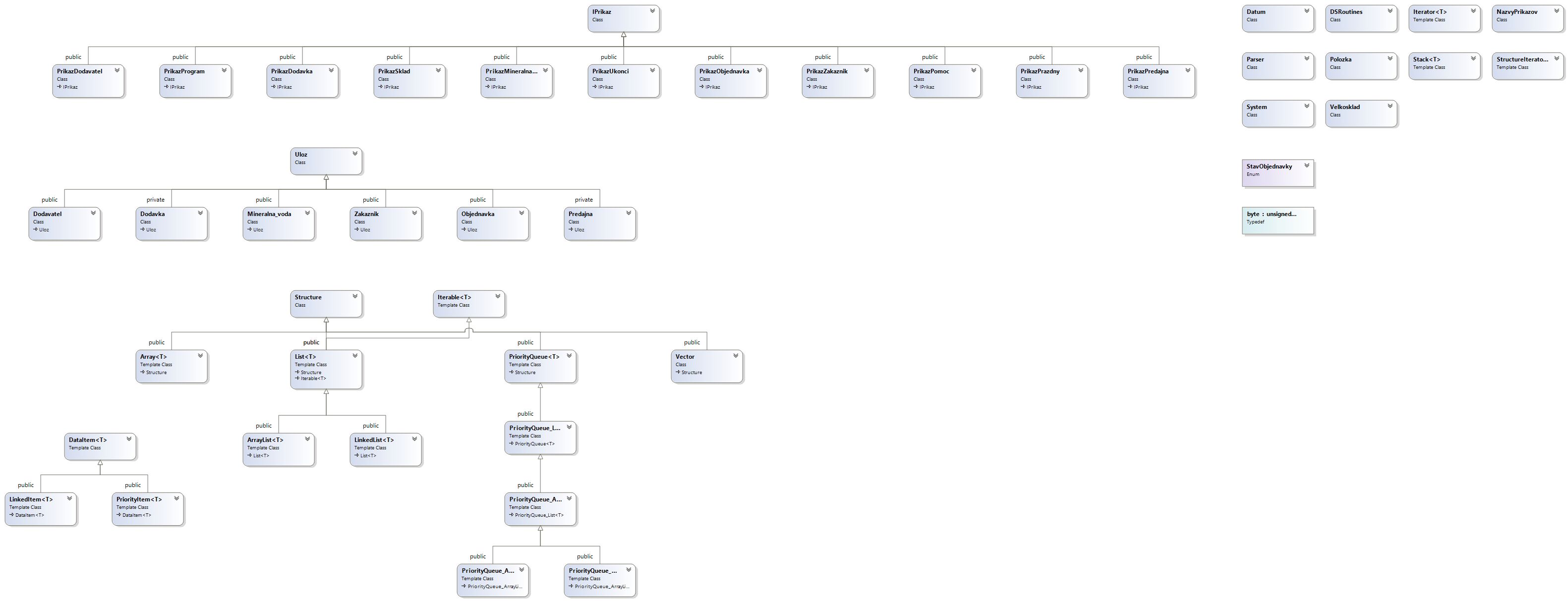
isEmpty -> O(1)

size -> O(1)

clear -> O(1)

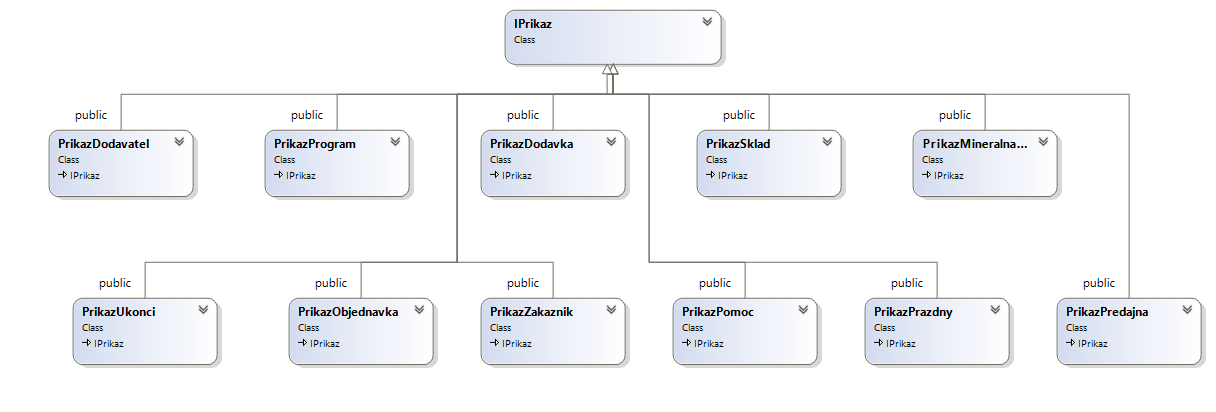
**Diagram tried**

( Všeobecný diagram )

****

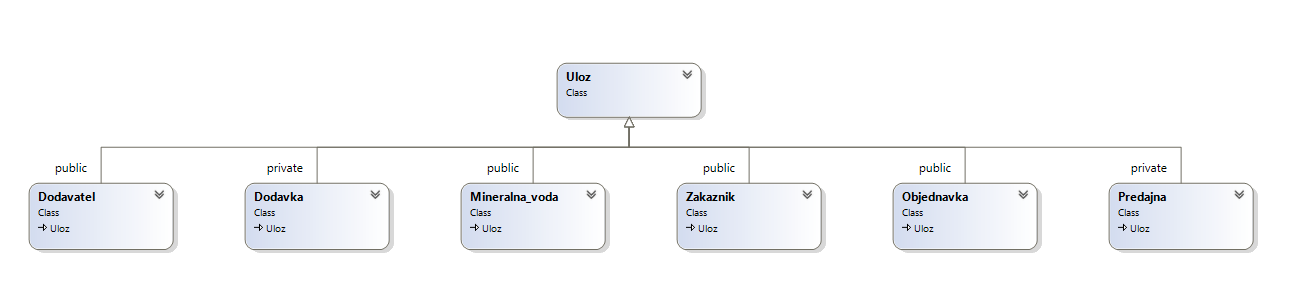
**Diagram tried**

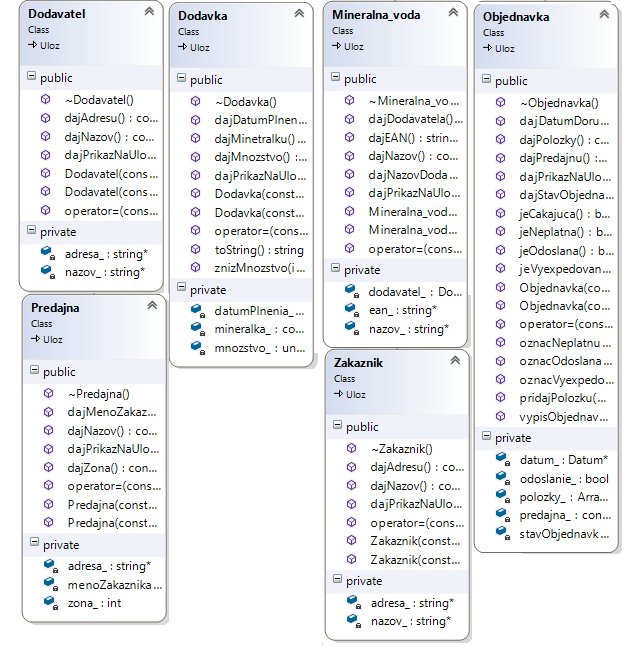
( Diagram triedy príkazov)



**Diagram tried aplikácie**

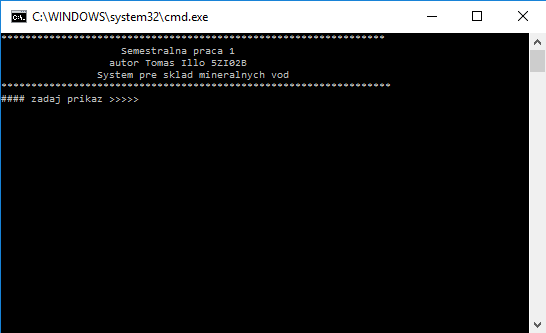
( Triedy ktoré obsahujú hlavnú logiku aplikácie )





**Používateľská príručka**

Aplikácia je ovládaná pomocou príkazov. Nižšie sú popísané jednotlivé príkazy.

****

**Spätná väzba aplikácie**

* Ak sa po zadaní príkazu vypísaný riadok začína znakom ‘$’
* Pri nevykonaní príkazu vypísaný riadok začína znakom ‘~‘
* Pomoc k príkazom začína ‘\*\*’
* Ak aplikácia očakáva vstup, riadok začína ‘####’

**Ovládanie aplikácie**

* Príkaz dodávateľ

dodavatel pridaj |meno| |adresa|

Pridá nového dodávateľa.

dodavatel najdiMax |odkedy| |dokedy|

Vypíše dodávateľa, po ktorého minerálnych vodách bol za dané časové obdobie najväčší dopyt.

* Príkaz dodávka

dodavka pridaj |typMineralnejVody| |mnozstvo| |datumPlnenia|

Pridá novú dodávku.

* Príkaz zákazník

zakaznik pridaj |meno| |adresa|

Pridá nového zákazníka.

zakaznik najdiMaxVoda |mineralnaVoda| |odkedy| |dokedy|

Vypíše zákazníka, ktorý za dané časové obdobie odobral najväčšie množstvo konkrétneho typu minerálnej vody.

zakaznik najdiMaxDodavatel |dodavatel| |odkedy| |dokedy|

Vypíše zákazníka, ktorý za dané časové obdobie odobral najväčšieod konkrétneho dodávateľa.

* Príkaz sklad

sklad vyskladni

Vyskladní tovar, ktorý má byť odovzdaný zákazníkovi nasledujúci deň.

sklad odovzdaj

Odovzdá tovar zákazníkovi.

sklad vypis

Vypíše obsah skladu.

* Príkaz minerálna\_voda

mineralnaVoda pridaj |nazov| |ean| |dodavatel|

Pridá novú minerálnu vodu.

* Príkaz predajna

predajna pridaj |menoZakaznika| |adresa| |zona|

Pridá novú predajnu.

* Príkaz program

program nacitaj

Načíta dáta zo súboru.

program uloz

Uloží dáta aplikácie do súboru.

* Príkaz pomoc

pomoc

Vypíše pomocné informácie k ovládaniu.

pomoc |prikaz|

Vypíše pomocné informácie ku konkrétnemu príkazu.

* Príkaz ukonci

Ukonci

Ukončí aplikáciu.