SEMESTRÁLNÍ PRÁCE Z PT

(standardní zadání)

Vypracovali:

Xuan Toan Dinh (A19B0027P): dinhgos@students.zcu.cz

Václav Šíma (A19B0203P): simava@students.zcu.cz

Zadání

- Vytvořit funkční program a dokumentaci
- Načíst vstupní data do vhodných datových struktur programu
- Vytvoření prostředí pro snadnou obsluhu programu
- Umožnit sledování aktuální stavu přepravy
- Provést simulace pro zadaný počet dnů a statistky uložit do souborů
- Vytvořit generátor vlastních dat
- Vytvořit Javadoc dokumentaci
- Vytvořit dále rozšiřitelný kód (kontrola pomocí PMD)

Obsah

Analýza problému 4
Načtení dat 4
Simulace 4
Implementace 5
UML diagram 6
Uživatelský dokumentace 7
Závěr 8

Analýza problému

Načtení dat

První problém, se kterým se během zpracování semestrální práce budeme potýkat, bude načtení vstupních dat a jejich navržení vhodných datových struktur pro jejich reprezentaci. Pro tuto úlohu se nám jeví nejvýhodnější využít matici sousednosti, kvůli rychlejšímu vyhledávání cest a taktéž formát textového dokumentu, ze kterého se budou data načítat je pro toto řešení optimální.

Simulace

Samotný chod simulace lze realizovat třemi způsoby. Rozvést kompletní produkci továren, rozvést pouze tolik zboží, kolik splní poptávku pro všechny supermarkety a možnost prediktivního plánování. V naší implementaci využijeme možnost číslo 2, kde supermarkety nakoupí pouze tolik zboží, aby naplnily poptávku s volitelnou možností pro prediktivní plánování.

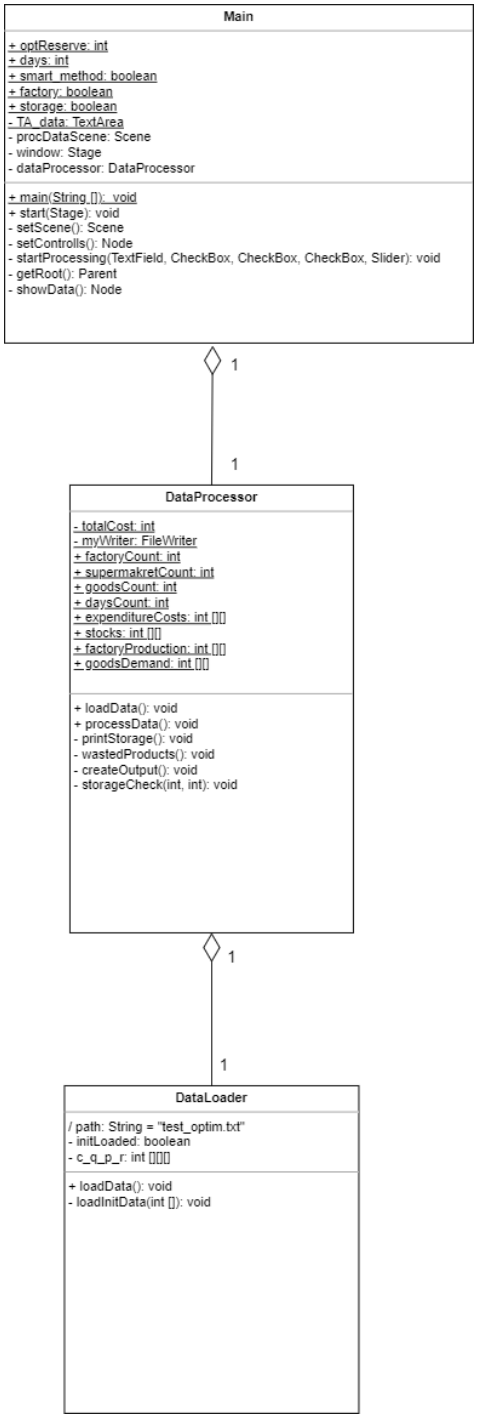
Implementace

Při spuštění programu se nejdříve načtou data ze vstupního souboru. Cesta k vstupnímu souboru se musí uvést ve třídě DataLoader do instance path. Pokud byl soubor zadán špatně nebo ji program nemůže přečíst, tak vypíše program chybovou hlášku a ihned se ukončí. Jinak pokud byl vstupní soubor zadán správně, tak její data načte do polí náklady, zásoby, produkce, poptávka a také si uloží informace o počtu továren, supermarketů, zboží a dní poté, co si uživatel zvolí nastavení pro simulaci (viz. Uživatelská příručka).

Při zahájení simulace se vytvoří textový soubor output.txt, kam se budou zapisovat statistiky simulace. Simulace má tři hlavní for cykly, který nejdřív projede všechny supermarkety potom všechny zboží a nakonec všechny dny. Nejprve se v cyklu vypíše stav zásob, potom se rozhodne v metodě poptavkaMinusZasoby(), jestli může poptávku splnit jenom pomocí zásob nebo ne. Pokud ne tak se zavolá metoda dovezZbozi(), která se postará o dovezení zboží. Pokud se nepodaří dovést zboží, tak vypíše chybovou hlášku.

Potom co všechny supermarkety mají splněnou poptávku, tak si dokoupí zboží do zásob pomocí metody zkontrolujZasoby(), která zkontroluje, jestli mají dostatek zboží na skladě. Pokud nemají maximální kapacitu zásob tak se zavolá metoda dovezZbozi(), která se postará o dovezení zboží. Na konci se vypíše, kolik zboží bylo vyprodukováno zbytečně, celková cena a celková doba simulace.

UML diagram



1. Obrázek – UML diagram

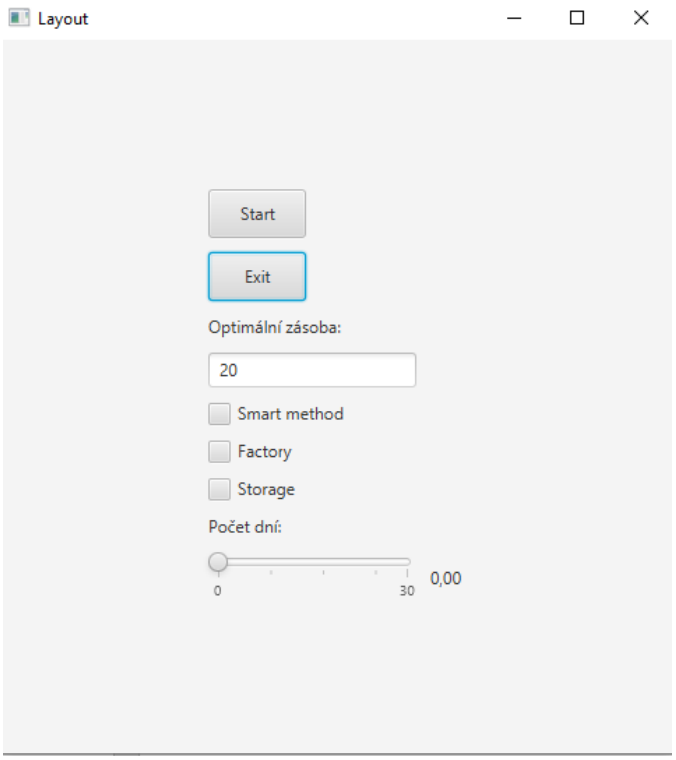
Uživatelský dokumentace

Po spuštění programu se uživateli zobrazí okno pro nastavení simulace. Tlačítko Start slouží k zahájení simulace s daným nastavením. Tlačítko Exit jednoduše ukončí program. Pod tlačítkem Exit je textové pole, kde uživatel může zadat optimální zásobu.

Optimální zásoba je celé číslo, které určuje, kolik zboží daného druhu si chce uživatel koupit do zásoby (křečkovat). Pokud zadá uživatel neplatné číslo, tak se optimální zásoba nastaví na standartní hodnotu, což je 2147483647.

Dále může uživatel zaškrtnout tři zaškrtávací políčka. Políčko Smart method je pro to, aby program koupil zásoby. Zaškrtávací políčko Factory na konci programu vypíše, co kam a kdy produkovaly a kolik zboží vyprodukovaly zbytečně. Poslední zaškrtávací políčko Storage vypíše každodenní rozpis skladových zásob. Na konci je slider, který určuje na kolik dní má simulace běžet.

Program následně vypíše statistiky do okna a také je uloží do textového souboru output.txt.



2. Obrázek – Layout aplikace

Závěr

Během semestrální práce se nám podařila implementovat většina požadavků ze zadání. Požadavky, které se nám nepodařilo zrealizovat, jsou vytvoření vlastního generátoru dat a umožnění sledování za běhu simulace.

Další bod, který by pravděpodobně šel vylepšit je formát výstupu, který náš program generuje.

Na druhou stranu se nám podařilo zdárně implementovat zbylé pod úlohy včetně grafického rozhraní pro obsluhu a náš kód je kvalitně zdokumentován pomocí Javadocu, zkontrolován pomocí PMD a průběh simulace je rychlý.