TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Testy pro určení míry shody dvou matic reálných čísel

Zápočtová práce

Autor: Kevin Daněk

Předmět: MTI/ALG1 – Algoritmizace a programování 1

Rok: 2021/2022

Liberec 2021

1. Úvod

Tento dokument popisuje rozbor, návrh a zpracování úlohy zadané v rámci zápočtové práce pro předmět *Algoritmizace a programování 1*.

2. Specifikace zadání

Výsledkem práce má být program, který uživateli dovolí načíst dvě obecně obdélníkové matice (dále jenom „matice“) a dovolí porovnat jejich vlastnosti a míry shody v sadě předdefinovaných testů.

Uživatelské rozhraní

Program a uživatel komunikují skrz standardní vstup a výstup. Program na standardní výstup vypisuje menu očíslovaných možností, ze kterého uživatel vybírá možnosti zadáním příslušného čísla.

Přesnost porovnávání

Přesnost při porovnávání matic je určena jako o dvanáct řádů menší největší absolutní hodnota v maticích.

Dílčí testy

Testy jsou prováděny na zadaných maticích a chování jednotlivých implementací je popsán dále v dokumentu. Mezi implementované testy patří:

- Shoda rozměry
- Shoda hodnotami
- Shoda znaménky prvků
- Shoda počtem nul
- Shoda součty řádků
- Shoda maximálními absolutními hodnotami
- Shoda transpozicí

3. Návrh a popis řešení

V souladu s obecnými požadavky je kód rozdělen do dvou hlavních tříd – jedna pro metody manipulující s maticemi a jedna pro komunikaci s uživatelem.

Knihovní třída *MatrixTools* obsahuje metody, které slouží pro načtení, výpis a práci s maticemi. Časová složitost algoritmů v knihovní třídě roste lineárně s počtem prvků matic.

Čtení a výpis matic

Čtení matic probíhá ve dvou fázích – čtení rozměrů a čtení hodnot.

Čtení rozměrů probíhá v cyklu s podmínkou za tělem cyklu, aby program nepokračoval na čtení hodnot, dokud zadané rozměry nedávají v kontextu programu smysl, tj. jsou nezáporné.

Čtení hodnot probíhá standardně, kdy počet zadaných hodnot je součin rozměrů matice. Čtení probíhá pomocí instance objektu *Scanner*, který nemá předdefinovanou lokalizaci, je tedy nutné dbát na správné užití desetinné tečky nebo čárky.

Výpis matic je ve formátu nadpisu a dílčí matice. Matice jsou formátovány pomocí metody *printf* na standardní výstup.

Transpozice matice

Transponování matice pomocí metody *transpose* funguje tak, že se vytvoří prázdná matice se stejnými rozměry. Následně se iteruje přes prvky matice z argumentu a přiřazují se na „opačné“ souřadnice prvku v prázdné matici, to znamená, že prvek z matice v argumentu na řádku m a sloupci n se přiřadí do prázdné matice jako prvek na řádku n a v sloupci m.

Porovnávání reálných čísel

Pro porovnávání reálných čísel byla zavedena metoda *compare*, která závisí nejenom na porovnávaných číslech z argumentu, ale také na přesnosti porovnávání, která je uchována jako statický atribut třídy *MatrixTools*.

Tato přesnost se aktualizuje po každém načtení matice a je vypočítána jako součin největší absolutní hodnoty v matici a jmenné konstanty s hodnotou $1E-12$, přičemž se pracuje s tou největší z obou matic.

Samotné porovnání probíhá pomocí *prahové metody* – pokud rozdíl porovnávaných čísel je menší než práh přesnosti, je rozdíl natolik zanedbatelný, že můžeme usoudit, že se čísla rovnají.

Shoda rozměry

Metoda pro zjištění, zda se matice shodují svými rozměry, se skládá z vyhodnocení složené podmínky, která kontroluje, zda se rovná počet řádků obou matic a zároveň zda se rovná počet sloupců obou matic.

Shodné hodnoty

Metoda pro zjištění, zda mají matice shodné hodnoty, pracuje s předpokladem, že nezáleží na pozici hodnot, ani na jejich počtu výskytu. Pokud mají matice stejné unikátní prvky, test vyjde pravdivý.

Pro uchování unikátních prvků se vybízí použití množiny – protože ale jeden z pokynů bylo nepoužívat prostředky pro *„výpis, uspořádání, vyhledávání a porovnávání“*, obsahuje kód vlastní implementaci datové struktury *Set*, která se chová jako dynamické pole s možností uchovat pouze unikátní hodnoty.

Samotný test se skládá z pomocné metody pro naplnění množiny unikátních hodnot pro matici, která je volána pro obě matice v argumentu, a následné porovnání délky a obsahu množin.

Shodná znaménka prvků

Metoda pro testování shodných znamének prvků pracuje s předpokladem, že matice mají stejné rozměry.

Mezi potenciální výsledky patří kladné znaménko, záporné znaménko nebo nula. Pro lepší čitelnost jsem namísto složených podmínek použil funkci *signum* ze třídy *Math*. Pokud se výsledky funkcí *signum* prvků nerovnají, mají rozdílná znaménka a test může skončit s výsledkem *nepravda*.

Shodný počet nul

Metoda pro testování stejného počtu nulových prvků v maticích používá pomocnou metodu, která iteruje přes prvky matice a spočítá počet nulových prvků – výsledky pro obě matice jsou následně porovnány.

Shodné součty řádků

Metoda pro testování, zda jsou součty prvků v řádcích pro obě matice stejné, pracuje s předpokladem, že matice mají stejné rozměry. Zároveň také platí, že pořadí součtů řádků musí být pro obě matice stejné.

Metoda využívá pomocné funkce, která vrací pole součtů řádků pro danou matici. Tyto součty jsou následně prvek po prvku porovnány, zdali jsou stejné.

Shodné maximální absolutní hodnoty

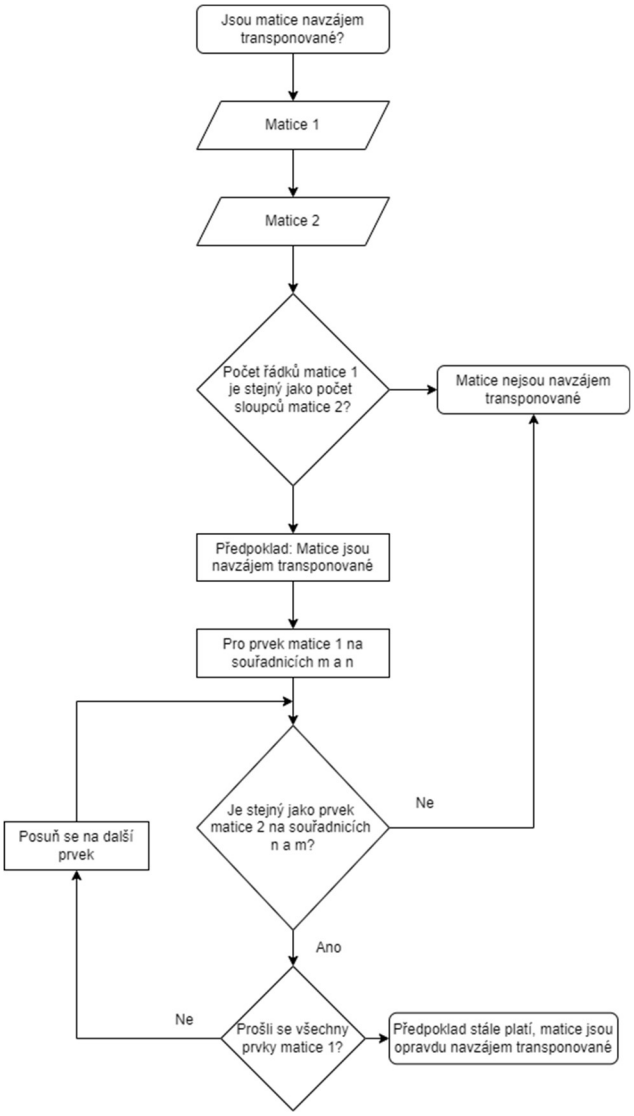
Metoda pro zjištění, zdali mají dvě matice stejné maximální absolutní hodnoty, projde každou matici zvlášť pomocí pomocné funkce. V těle této funkce začíná maximum na záporném nekonečnu definovaném v objektu *Double*. Výsledky pro obě matice se následně navzájem porovnají, a výsledek je vrácen.

Shoda dle transpozice

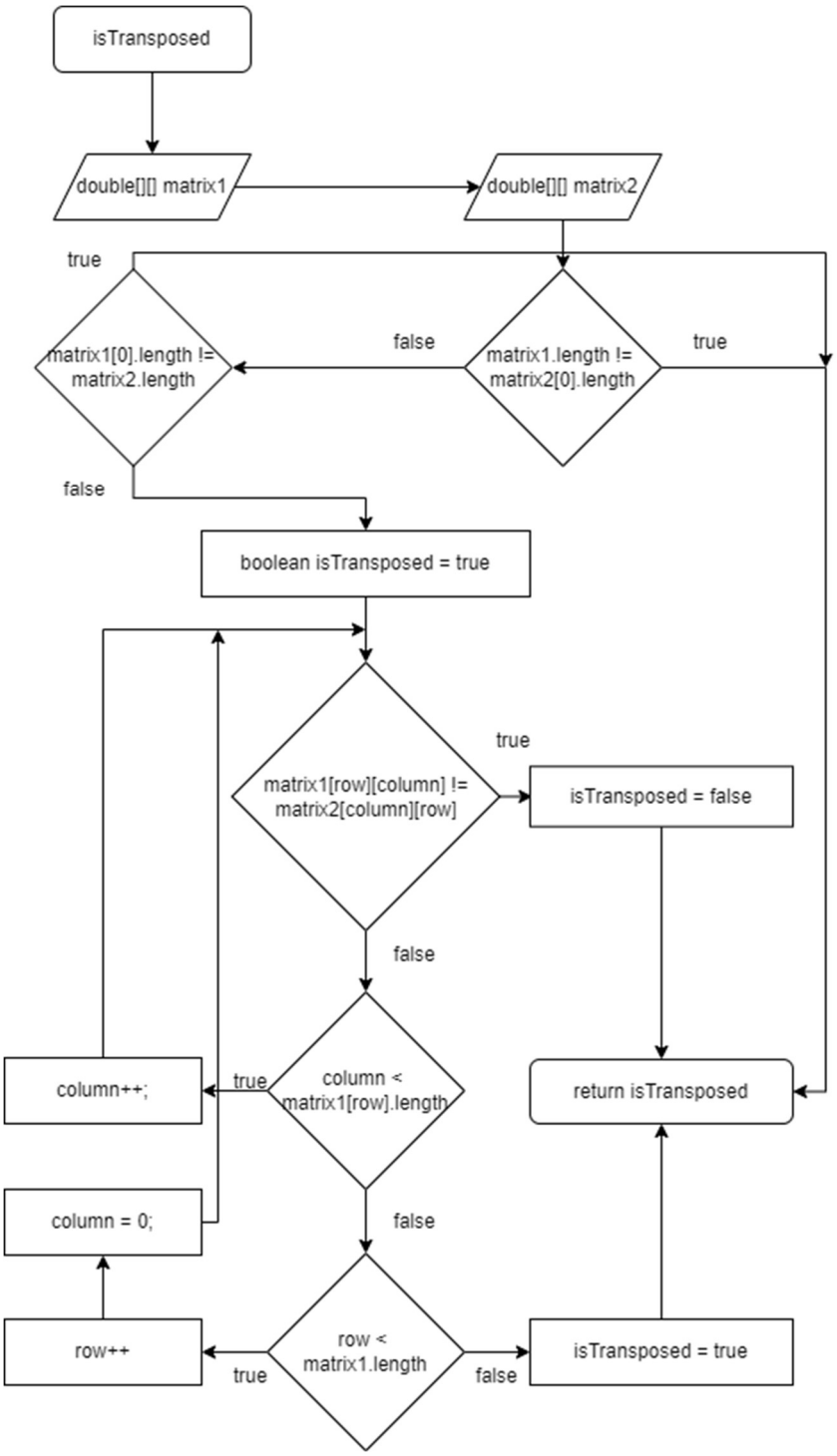
Metoda pro testování, zdali jsou matice navzájem transponované, fungují podobně jako algoritmus pro samotnou transpozici, ale s tím rozdílem, že místo přiřazování do prázdné matice se zde porovnává prvek z první matice na souřadnicích $[x, y]$ s prvkem z druhé matice na souřadnicích $[y, x]$.

4. Vývojový diagram vybrané úlohy

Pro vývojový diagram byl zadán algoritmus pro ověření, zda jsou dvě matice navzájem transponované, tj. zda je jedna transpozicí druhé a naopak. Připravil jsem dva vývojové diagramy – jeden popisující obecný algoritmus, nezávisle na implementaci a prostředí, a druhý, který je blíže ke skutečné implementaci nalezené ve zdrojovém kódu.



Obrázek 1 - Obecný algoritmus pro zjištění, zda jsou matice navzájem transponované



Obrázek 2 - Algoritmus pro zjištění, zda jsou matice navzájem transponované, který je blíže k implementaci v kódu

5. Testování programu

Test 1

Mějte dvě matice o rozměrech 3x3

1	2	3
4	5	6
7	8	9

9	8	7
6	5	4
3	2	1

Test	Očekávané chování	Chování programu
Shodné rozměry	Ano	Ano
Shodné hodnoty	Ano	Ano
Shodná struktura	Ano	Ano
Shodný počet nul	Ano	Ano
Shodné součty řádků	Ne	Ne
Shodné maximální absolutní hodnoty	Ano	Ano
Navzájem transponované	Ne	Ne

Test 2

Mějte dvě matici 3x3 a matici 4x2

1	2	3
4	5	6
7	8	9

1	2
2	4
5	6
7	8

Test	Očekávané chování	Chování programu
Shodné rozměry	Ne	Ne
Shodné hodnoty	Ne	Ne
Shodná struktura	Ne	Ne
Shodný počet nul	Ano	Ano
Shodné součty řádků	Ne	Ne
Shodné maximální absolutní hodnoty	Ne	Ne
Navzájem transponované	Ne	Ne

Test 3

Mějte dvě matici 2x5 a matici 5x2

1	0	1	0	-1
0	-1	0	1	0

1	0
0	-1
1	0
0	1
-1	0

Test	Očekávané chování	Chování programu
Shodné rozměry	Ne	Ne
Shodné hodnoty	Ano	Ano
Shodná struktura	Ne	Ne
Shodný počet nul	Ano	Ano
Shodné součty řádků	Ne	Ne
Shodné maximální absolutní hodnoty	Ano	Ano
Navzájem transponované	Ano	Ano

Záporné rozměry matice

Při zadání alespoň jednoho záporného či nulového rozměru při zadávání jedné z matic program zareaguje zprávou, že byly zadány neplatné rozměry a aby znovu zadal platné rozměry.

Spuštění testu bez načtených matic

Pokud uživatel spustí test bez načtení obou matic do paměti programu, tak program zareaguje vyvoláním výjimky, že jedna nebo druhá matice jsou *null.*

Výpis matic bez načtených matic

Pokud uživatel spustí výpis matic bez načtení jakýchkoliv matic, program vyvolá výjimku, že matice jsou typu *null*.