

**Semestrální práce**

Předmět:MTI/ALG1 – Algoritmizace a programování 1

Studijní obor: B0613A140005 - Informační technologie

Autor práce: Tomáš Kulhavý

Vedoucí práce: Ing. Jana Vitvarová, Ph.D.

**Transformace zrcadlení matic**

**Obsah**

[1. Specifikace požadavků 3](#__RefHeading___Toc138_318290650)

[2. Návrh řešení 4](#__RefHeading___Toc161_318290650)

[1 Zrcadlení podle hlavní diagonály 4](#__RefHeading___Toc128_2474860443)

[2 Zrcadlení podle vedlejší diagonály 4](#__RefHeading___Toc130_2474860443)

[3 Zrcadlení podle horizontální osy 5](#__RefHeading___Toc132_2474860443)

[4 Zrcadlení podle vertikální osy 5](#__RefHeading___Toc134_2474860443)

[3. Protokol z testování 6](#__RefHeading___Toc163_318290650)

[Přílohy 9](#__RefHeading___Toc169_318290650)

# Specifikace požadavků

Program by měl umět načíst dvě čtvercové matice stejné velikosti a měl by zjistit zda druhá matice vznikne jednoduchou transformací zrcadlení podle hlavní, vedlejší diagonály nebo horizontální, vertikální osy matice.

Program má umožnit při jednom spuštění zpracování libovolného počtu zadání. Pro každou úlohu program načte celé číslo jako rozměr zpracovávaných matic, poté budou načítány jednotlivé matice po řádcích. Program vyzve uživatele k zadání celého čísla. Všechny vstupy až na zadávání čísel do matice jsou ošetřeny, aby bylo možné napsat pouze celé číslo.

Pro každé zadání program vypíše informaci, zda druhá ze zadaných matic vznikne z první některou z uvedených transformací. Program výsledek svého šetření vypisuje do konzole jako jednu z následujících krátkých textových zpráv: „Zrcadlení dle hlavní diagonály“, „Zrcadlení dle vedlejší diagonály“, „Zrcadlení dle horizontální osy“, „Zrcadlení dle vedlejší osy“ nebo „Nenalezena transformace“. Po načtení záporného nebo nulového čísla místo rozměru matice program skončí svoji činnost.

Pro dvě zadané matice může program najít i více transformací, které vypíše na výstup do konzole.

# Návrh řešení

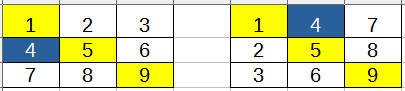
Program je rozdělen na hlavní třídu Main.java. Ta slouží k rozcestníku programu. Uživateli vypíše do konzole menu pomocí cyklu do-while. Uživatel má na výběr vánoční úlohu, semestrální práci nebo možnost z programu odejít.

Dále program obsahuje třídu MatrixUI.java. Ta komunikuje s uživatelem a slouží k načítání rozměru matice a k plnění samotné matice. Rozměry matice a prvky v ní musí být pouze celá čísla. Matice je plněna pomocí cyklu for, která ji plní postupně po řádcích.

K vyhodnocení transformace matice je zde třída MatrixTools.java. Každá metoda přijímá dvě matice celých čísel a vracejí boolean. Pokud je matice symetrická podle daných požadavků vrátí „true“, jestliže ne tak vrátí „false“.

## Zrcadlení podle hlavní diagonály

Algoritmus pro zjištění transformace zrcadlení podle hlavní diagonály funguje přes cykly for. Postupně z načtených matic si vybere řádky např. u matice 3x3 druhý řádek z první matice a první řádek z druhé matice. A pomocí druhého cyklu for si vybere první číslo z druhého řádku první matice a druhé číslo z prvního řádku druhé matice. Pokud se tyto prvky rovnají, metoda pokračuje dál a na konci vrátí „true“. Jestliže se tyto čísla nerovnají metoda vrátí „false“ a metoda se ukončí.

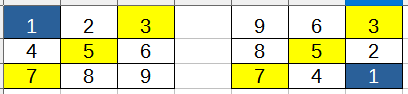
Obrázek 1: Zrcadlení podle hlavní diagonály (porovnávání členů)

* První cyklus for jde po délku řádku první matice
* Druhý cyklus for jde po velikost prvního cyklu
* Poté porovnává matici A[sloupec][řádek] a matici B[řádek][sloupec] – tudíž porovnáváme prohozené pozice řádků a sloupců

## Zrcadlení podle vedlejší diagonály

Algoritmus pro zjištění transformace zrcadlení podle vedlejší diagonály funguje podobně jako algoritmus podle hlavní diagonály. Porovnává první řádek první matice a poslední řádek druhé matice. Z první matice vybere první číslo a z druhé matice vybere poslední číslo.

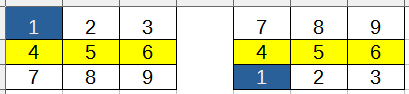
* První cyklus for jde po délku řádku první matice – 1
* Druhý cyklus for jde po velikost sloupce (index sloupce je z prvního cyklu) a odečítá se jednička a index sloupce z prvního cyklu
* Poté porovnává matici A[sloupec][řádek] a matici B[velikost řádku – 1 – hodnota druhého cyklu][velikost sloupce – 1 – hodnota prvního cyklu (index sloupce)] – tudíž porovnáváme prohozené pozice řádků a sloupců

Obrázek 2: Zrcadlení podle vedlejší diagonály (porovnávání členů)

## Zrcadlení podle horizontální osy

Algoritmus pro zjištění transformace zrcadlení podle horizontální osy také funguje přes cykly for. Postupně z načtených matic si vybere řádky např. u matice 3x3 první řádek z první matice a poslední řádek z druhé matice. A pomocí druhého cyklu for si vybere první číslo z prvního řádku první matice a první číslo číslo z posledního řádku druhé matice. Pokud se tyto prvky rovnají, metoda pokračuje dál a na konci vrátí „true“. Jestliže se tyto čísla nerovnají metoda vrátí „false“ a metoda se ukončí.

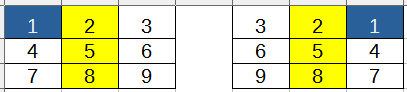
* První cyklus for jde po délku řádku první matice
* Druhý cyklus for jde po velikost sloupce (index sloupce je hodnota z prvního cyklu) děleno dvěma – dělí se dvěma, jelikož jdeme po poloviční velikost sloupce
* Poté porovnává matici A[sloupec][řádek] a matici B[sloupec][velikost sloupce – 1 – aktuální hodnota druhého cyklu (index řádku)]

Obrázek 3: Zrcadlení podle horizontální osy (porovnávání členů)

## Zrcadlení podle vertikální osy

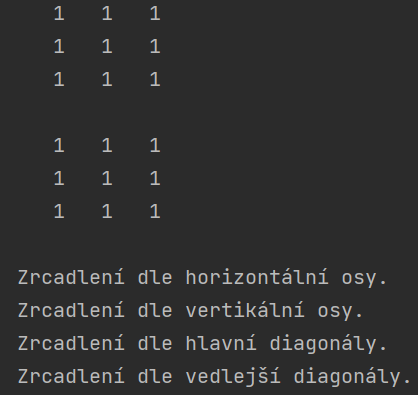
Algoritmus pro zjištění transformace zrcadlení podle vertikální osy také funguje podobně jako algoritmus podle horizontální osy. Ale porovnává první sloupec první matice a poslední sloupec druhé matice. Z první a druhé matice vybere první číslo ze sloupců a porovná je. Pokud se rovnají vrátí „true“, pokud ne tak vrátí „false“.

* První cyklus for jde po délku řádku první matice děleno dvěma – jelikož jdeme pouze do půlky velikosti řádku
* Druhý cyklus for jde po velikost sloupce (index sloupce je hodnota z prvního cyklu)
* Poté porovnává matici A[sloupec][řádek] a matici B[velikost řádku – 1 – hodnota prvního cyklu (index sloupce)][řádek] – tudíž porovnáváme prohozené pozice řádků a sloupců

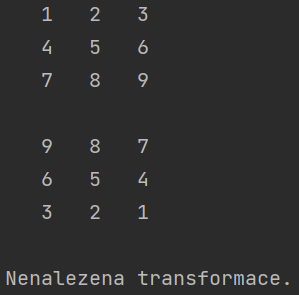
Obrázek 4: Zrcadlení podle vertikální osy (porovnávání členů)

# Protokol z testování

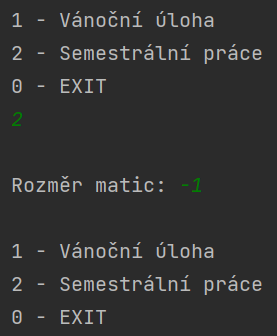
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Číslo testu | Typ testu | Očekávaný výsledek | Skutečný výsledek | Prošel |
| 1 | běžná hodnota | Všechny možné stavy | Všechny možné stavy | ano |

Obrázek 5: Test č. 1 – Výpis všech možných stavů

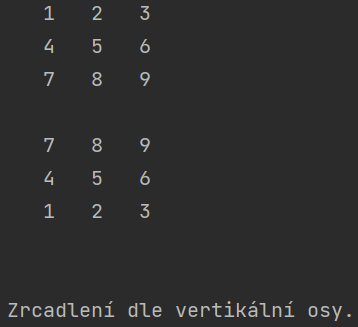
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Číslo testu | Typ testu | Očekávaný výsledek | Skutečný výsledek | Prošel |
| 2 | běžná hodnota | Nenalezena transformace | Nenalezena transformace | ano |

Obrázek 6: Test č. 2 - Žádná transformace

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Číslo testu | Typ testu | Očekávaný výsledek | Skutečný výsledek | Prošel |
| 3 | běžná hodnota | Výstup z úlohy matic | Výstup z úlohy matic | ano |

Obrázek 7: Test č. 3 - Výstup z úlohy matic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Číslo testu | Typ testu | Očekávaný výsledek | Skutečný výsledek | Prošel |
| 4 | běžná hodnota | zrcadlení dle vertikální osy | zrcadlení dle vertikální osy | ano |

Obrázek 8: Test č. 4 - Zrcadlení dle vertikální osy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Číslo testu  Obrázek 9: Test č. 5 - Nevalidní vstup | Typ testu | Očekávaný výsledek | Skutečný výsledek | Prošel |
| 5 | Nevalidní vstup | chyba | chyba | ano |

# Přílohy

Seznam obrázků

[Obrázek 1: Zrcadlení podle hlavní diagonály (porovnávání členů) 4](#Obr%C3%A1zek!0|sequence)

[Obrázek 2: Zrcadlení podle vedlejší diagonály (porovnávání členů) 4](#Obr%C3%A1zek!1|sequence)

[Obrázek 3: Zrcadlení podle horizontální osy (porovnávání členů) 5](#Obr%C3%A1zek!2|sequence)

[Obrázek 4: Zrcadlení podle vertikální osy (porovnávání členů) 5](#Obr%C3%A1zek!3|sequence)

[Obrázek 5: Test č. 1 – Výpis všech možných stavů 6](#Obr%C3%A1zek!4|sequence)

[Obrázek 6: Test č. 2 - Žádná transformace 6](#Obr%C3%A1zek!5|sequence)

[Obrázek 7: Test č. 3 - Výstup z úlohy matic 7](#Obr%C3%A1zek!6|sequence)

[Obrázek 8: Test č. 4 - Zrcadlení dle vertikální osy 7](#Obr%C3%A1zek!7|sequence)

[Obrázek 9: Test č. 5 - Nevalidní vstup 8](#Obr%C3%A1zek!9|sequence)