

Semestrální práce – úloha 23

Technická dokumentace

zpracoval Miroslav Kramár

# OBSAH

* Specifikace požadavků
  + Zadání
  + Ukázka komunikace programu s uživatelem
  + Komentář ke specifikaci požadavků
  + Předpoklady pro fungování programu
* Návrh řešení
  + Popis a řešení problému
  + Postup řešení
* Protokol z testování
* Výsledky akceptačních testů

# SPECIFIKACE POŽADAVKU

Zadání

*Zapište program, který načte matici a bude provádět zadané transformace zadané matice o 0, -90 a 90 stupňů. Transformace budou zadávány hodnotami 0, -1, 1.*

*Specifikace vstupu*

*Program má umožnit při jednom spuštění zpracování libovolného počtu zadání. Pro každou úlohu program nejdříve načte dvě celá čísla jako počet řádků a počet sloupců matice a poté vlastní matici. Dále bude program postupně načítat čísla 0, 1 nebo -1 a pro každé zadané číslo provede příslušnou transformaci. Zadáním jiného čísla končí zpracování aktuální úlohy. Po načtení záporného nebo nulového čísla namísto počtu řádků matice, nechť program skončí svoji činnost.*

Ukázka komunikace programu s uživatelem

*Pocet radku*

*3*

*Pocet sloupcu*

*3*

*Matice*

*1 2 3*

*4 5 6*

*7 8 9*

*Transformace*

*1*

*Rotovaná matice*

*3 6 9*

*2 5 8*

*1 4 7*

*Transformace*

*1*

*Rotovaná matice*

*9 8 7*

*6 5 4*

*3 2 1*

*Transformace*

*0*

*Rotovaná matice*

*9 8 7*

*6 5 4*

*3 2 1*

*Transformace*

*2*

*Pocet radku*

*-1*

Komentář ke specifikaci požadavku

V ukázce interakce uživatele s programem jsou uživatelem zadávané jednotlivé prvky matice pouze celá čísla a nikde není naznačeno, že by program měl být schopen jako prvky matice akceptovat i jiné číselné datové typy. Vzhledem k tomuto faktu jsem program zpracoval tak, že akceptuje a pracuje pouze s celými čísly zadanými uživatelem.

Předpoklady pro fungování programu

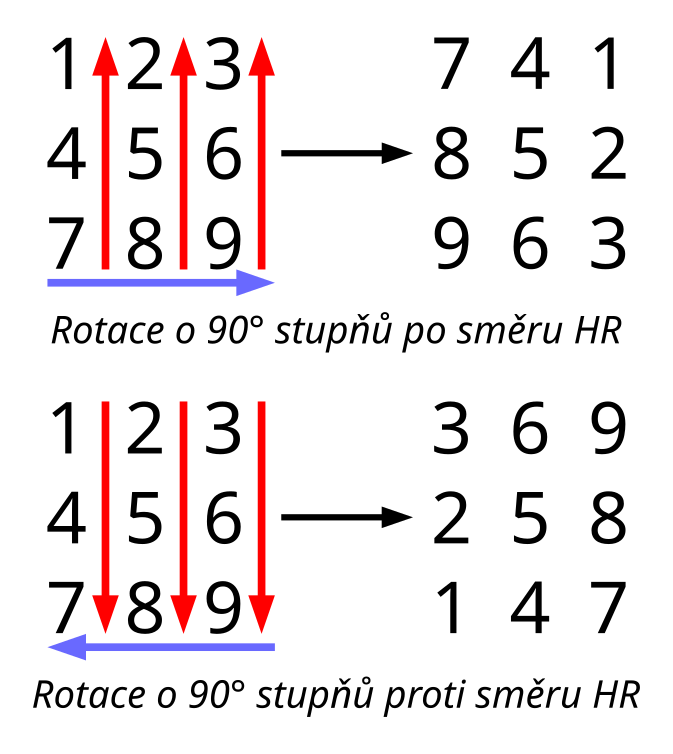
Předpokládám, že pokud v programu uživatele vyzvu k zadání specifického datového typu, tak ho uživatel skutečně zadá. V opačném případě program nevyhnutelně selže.

V případě, že uživatel zadá očekávaný parametr následovaný neočekávaným/nežádoucím parametrem, tak program neselže, jelikož program po každém načtení vyčistí zbytek řádku vstupu.

# NÁVRH ŘEŠENÍ

Popis a řešení problému

Nejstěžejnější na celém řešení bylo přijít na správné pořadí, ve kterém přesouvat prvky z původní matice do nové, rotované matice. Řešení tohoto problému je zobrazené na následujícím obrázku.



Prvky z původní matice se do rotované matice přesouvají tak, že postupně čteme jednotlivé prvky ve sloupci směrem, který udává červená šipka. Jednotlivé sloupce poté procházíme ve směru modré šipky.

Postup řešení

Finální řešení se dá formálně rozdělit na dvě části, a to sice na jednotlivé metody pro rotování matic a program starající se o načítání vstupů a jejich zpracování pomocí zmíněných metod.

**Metody:**

* **Načtení matice**
  1. Zjistím velikost zadané matice
     + Pomocí metody length()
  2. Postupně načítám každý prvek v řádku matice
     + Pomocí metody nextInt()
  3. Když dojdu na konec řádku, tak skočím na nový řádek a opakuji
     + Využít dvou vnořených for cyklů
* **Vypsání matice**
  1. Zjistím velikost zadané matice
     + Pomocí metody length()
  2. Postupně vypisuji každý prvek v řádku matice
     + Pomocí metody System.out.println()
  3. Když dojdu na konec řádku, tak začnu nový řádek a opakuji
     + Využít dvou vnořených for cyklů
* **Rotace matice po směru hodinových ručiček**
  1. Zjistím velikost zadané matice
     + Pomocí metody length()
  2. Založím novou matici, do které budu přesouvat prvky ze zadané matice. Tato nová matice bude mít oproti zadané matici prohozenou velikost řádků a sloupců, tj. pokud zadaná matice má 2 řádky a 3 sloupce, tak nová matice bude mít řádky 3 a sloupce 2.
  3. Postupuji zadanou maticí tak, že každý sloupec čtu odzdola nahoru, přičemž sloupci matice procházím zleva doprava. Novou, rotovanou matici zaplňuji standardně, tj. do řádků zapisuji do řádků zapisuji zleva doprava, přičemž řádky píši odshora dolu.
     + Využít dvou vnořených for cyklů. Ve vnějším cyklu bude i nabývat hodnot od 0 do počtu řádků zdrojové matice. Ve vnitřním cyklu bude j nabývat hodnot od počtu sloupců zdrojové matice minus 1 a bude klesat až na hodnotu 0.
     + Založit 2 pomocné proměnné pro procházení nové matice, např. iTarget a jTarget.
       - iTarget bude deklarován před vnořenými cykly a bude nastaven na hodnotu 0. Hodnota iTarget se bude zvětšovat o 1 vždy na konci vnějšího cyklu.
       - jTarget bude deklarován na začátku vnějšího cyklu a jeho hodnota bude nastavena na 0. Hodnota jTarget se bude zvětšovat o 1 vždy na konci vnitřního cyklu.
     + Uvnitř vnořených cyklů se bude hodnota prvku ze zdrojové matice na pozici **[j] [i]** přiřazovat do nové matice na pozici **[iTarget] [jTarget]**.
     + Na konci je potřeba matici vrátit
* **Rotace matice proti směru hodinových ručiček**
  1. Zjistím velikost zadané matice
     + Pomocí metody length()
  2. Založím novou matici, do které budu přesouvat prvky ze zadané matice. Tato nová matice bude mít oproti zadané matici prohozenou velikost řádků a sloupců.
  3. Postupuji zadanou maticí tak, že každý sloupec čtu odshora dolu, přičemž sloupci matice procházím zprava doleva. Novou, rotovanou matici zaplňuji standardně.
     + Využít dvou vnořených for cyklů. Ve vnějším cyklu bude i nabývat hodnot od 0 do počtu sloupců zdrojové matice. Ve vnitřním cyklu bude j nabývat hodnot od počtu řádků zdrojové matice minus 1 a bude klesat až na hodnotu 0.
     + Založit 2 pomocné proměnné pro procházení nové matice, např. iTarget a jTarget.
       - iTarget bude deklarován před vnořenými cykly a bude nastaven na hodnotu 0. Hodnota iTarget se bude zvětšovat o 1 vždy na konci vnějšího cyklu.
       - jTarget bude deklarován na začátku vnějšího cyklu a jeho hodnota bude nastavena na 0. Hodnota jTarget se bude zvětšovat o 1 vždy na konci vnitřního cyklu.
     + Uvnitř vnořených cyklů se bude hodnota prvku ze zdrojové matice na pozici **[i] [j]** přiřazovat do nové matice na pozici **[jTarget] [iTarget]**.
     + Na konci je potřeba matici vrátit

**Program pro zpracování vstupů pomocí funkcí**

1. Načtu od uživatele počet řádků a počet sloupců matice
   * Pomocí metody nextInt(); následované samotnou metodou nextLine() pro vyčištění vstupního řádku
2. Založím novou matici o zadané velikosti
3. Načtu jednotlivé prvky matice
   * Pomocí vlastní metody pro načtení matice
4. Načtu od uživatele, jakým směrem chce matici rotovat
   * Pomocí metody nextInt(); následované samotnou metodou nextLine() pro vyčištění vstupního řádku
5. Matici otočím příslušným směrem či ji pouze vypíši
   * Pomocí switche, který na základě přijaté hodnoty volá příslušnou vlastní metodu pro rotování či vypsání matice
6. Opakuji výzvu k zadání směru rotace. Pokud již uživatel matici rotovat nechce, tak opakuji výzvu k zadání nové matice o nové velikosti. Pokud již uživatel nechce zadávat novou matici, tak ukončím program.
   * Pomocí vnořených smyček while. Vnější smyčka while se stará o načítání velikosti matice a jejích prvků, vnitřní smyčka while se poté stará o načítání směru rotace a volání metod pro příslušnou rotaci

# PROTOKOL Z TESTOVÁNÍ

Tabulka vstupních hodnot, které program očekává

|  |  |  |  |
| --- | --- | --- | --- |
| **Akce** | **Očekávané hodnoty** | **Alternativní hodnoty** | **Akce alternativních hodnot** |
| Zadání počtu řádků matice | Celé kladné číslo | 0 a záporná čísla | Ukončení programu |
| Zadání počtu sloupců matice | Celé kladné číslo | 0 a záporná čísla | Ukončení programu |
| Zadání prvků matice | Celá čísla oddělená mezerou | Žádné | Nic |
| Zadání směru rotace | -1, 0, 1 | Jakékoli jiné celé číslo | Ukončení rotování matice, návrat k zadávání prvků matice |

Následuje tabulka provedených testovacích vstupů programu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Číslo testu** | **Typ testu** | **Zadaná hodnota** | **Očekávaný výsledek** | **Skutečný výsledek** | **Prošel (ano/ne)** |
| 1 | Běžná hodnota | Hodnoty zadávané tak, jak program očekává / vyzívá uživatele | Program se provede bez chyb | Program se provedl bez chyb | Ano |
| 2 | Limitní stav | Očekávané hodnoty (různá celá čísla) následované dalším, nežádoucím parametrem | Program se provede bez chyb | Program se provedl bez chyb | Ano |
| 3 | Limitní stav | Zadání více prvků matice, než je očekáváno | Program se provede bez chyb, přebývající prvky budou ignorovány | Program se provedl bez chyb, přebývající prvky byly ignorovány | Ano |
| 4 | Limitní stav | Zadání menšího počtu prvků matice, než je očekáváno | Program počká, dokud uživatel nepředá chybějící počet prvků | Program počkal, dokud uživatel nepředá chybějící počet prvků | Ano |
| 5 | Nevalidní vstup | Jiná než celá čísla, jiné znaky, než čísla | Program selže | Program selhal | Ano |

Komentář k testování programu

Vzhledem k povaze programu, tj. že program nic nevypočítává, pouze přesouvá data z jedné matice do druhé, a jeho relativní jednoduchosti není příliš mnoho prostoru pro neočekávané chování programu způsobené zadáním neočekávaných vstupních hodnot, jelikož program de facto nezná nic, jako je neočekávaná hodnota. Program se rozhoduje na základě toho, jestli obdržel určitou konkrétní hodnotu, ale se samotnou hodnotou neprovádí žádné výpočty.

# VÝSLEDKY AKCEPTAČNÍCH TESTŮ

Následují screenshoty výsledků akceptačních testů

Běžné hodnoty

Obsah obrázku text, stůl

Popis byl vytvořen automaticky

Limitní stav 1

Obsah obrázku text

Popis byl vytvořen automaticky

Limitní stav 2

Obsah obrázku text

Popis byl vytvořen automaticky

Nevalidní vstup

Obsah obrázku text

Popis byl vytvořen automaticky