Отчет по лабораторной работе № 3
по курсу «Функциональное программирование»

Студент группы М8О-307-19 МАИ *Инютин Максим Андреевич*, №10 по списку
Контакты: mainyutin@gmail.com
Работа выполнена: 01.05.2022

Преподаватель: Иванов Дмитрий Анатольевич, доц. каф. 806
Отчет сдан:
Итоговая оценка:
Подпись преподавателя:

1. Тема работы

Последовательности, массивы и управляющие конструкции Common Lisp.

2. Цель работы

Научиться создавать векторы и массивы для представления матриц, освоить общие функции работы с последовательностями, инструкции цикла и нелокального выхода.

3. Задание (вариант № 3.47)

Назовём допустимым преобразованием матрицы перестановку двух строк или двух столбцов.

Запрограммировать на языке Коммон Лисп функцию, принимающую в качестве единственного аргумента двумерный массив — действительную матрицу. Функция должна возвращать новую матрицу (исходный массив должен оставаться неизменным), полученную путём допустимых преобразований из исходной матрицы, и такую, чтобы один из элементов матрицы, обладающий наибольшим по модулю значением, располагался в левом верхнем углу матрицы.

Каждый шаг следует выдавать на печать.

4. Оборудование студента

Процессор Intel Core i7-9750H (12) @ 4.5GHz, память: 32 Gb, разрядность системы: 64.

5. Программное обеспечение

ОС Ubuntu 20.04.4 LTS, комилятор GNU CLISP 2.49.92, текстовый редактор Atom 1.58.0

6. Идея, метод, алгоритм

Скопирую матрицу, чтобы не менять исходную. Затем найду индексы одного из максимальных по модулю элементов. Пусть эти индексы i и j. Тогда требуется поменять первую и i-ю строку, первый и j-й столбец. Если $i = 1$, то менять строки не нужно. Аналогично если $j = 1$, менять столбцы не нужно.

Обход всей матрицы для поиска элемента имеет сложность $O(n \cdot m)$. Перестановка двух столбцов или строк имеют линейную сложность, поэтому итоговая сложноть $O(n \cdot m)$.

7. Сценарий выполнения работы

8. Распечатка программы и её результаты

8.1. Исходный код

```
; 3.47

; Returns indexes of element with maximal absolute value
(defun find-max-ind (a)
        (let ((res_i 0) (res_j 0)
        (n (array-dimension a 0)) (m (array-dimension a 1)))
        (loop for i from 0 to (- n 1)
                do (loop for j from 0 to (- m 1)
                        do (if (> (abs (aref a i j)) (abs (aref
   a res_i res_j)))
                                (progn (setq res_i i) (setq
   res_j j)))))
        (values res_i res_j)))

; Swaps rows i1 and i2
(defun swap-row (a i1 i2)
        (princ "Swapping row ")
        (princ (+ i1 1))
        (princ " and ")
        (princ (+ i2 1))
        (terpri)
        (let ((m (array-dimension a 1)))
        (loop for j from 0 to (- m 1)
                do (rotatef (aref a i2 j) (aref a i1 j)))))

; Swaps columns j1 and j2
(defun swap-col (a j1 j2)
```

```
        (princ "Swapping column ")
        (princ (+ j1 1))
        (princ " and ")
        (princ (+ j2 1))
        (terpri)
        (let ((n (array-dimension a 0)))
        (loop for i from 0 to (- n 1)
                do (rotatef (aref a i j1) (aref a i j2)))))

; Returns matrix copy
(defun copy-matrix (a)
        (let* ((n (array-dimension a 0)) (m (array-dimension a
   1))
        (res (make-array (list n m) :initial-element 0.0)))
        (loop for i from 0 to (- n 1)
                do (loop for j from 0 to (- m 1)
                        do (setf (aref res i j) (aref a i j))))
        res))

(defun swap-max-to-top-left (matr)
        (let ((a (copy-matrix matr))
        (i (nth-value 0 (find-max-ind a)))
        (j (nth-value 1 (find-max-ind a))))
        (if (> i 0) (swap-row a 0 i))
        (if (> j 0) (swap-col a 0 j))
        a))
```

8.2. Результаты работы

```
[1]> (load "lab3.lisp")
;; Loading file lab3.lisp ...
;; Loaded file lab3.lisp
#P"/home/engineerxl/Study/6
   term/functional-programming/lab3/lab3.lisp"
[2]> (setq a (make-array '(1 1) :initial-contents '((1))))
#2A((1))
[3]> (swap-max-to-top-left a)
#2A((1))
[4]> (setq a (make-array '(1 2) :initial-contents '((1 2))))
#2A((1 2))
[5]> (swap-max-to-top-left a)
Swapping column 1 and 2
#2A((2 1))
[6]> a
#2A((1 2))
[7]> (setq a (make-array '(1 2) :initial-contents '((1 -2))))
#2A((1 -2))
[8]> (swap-max-to-top-left a)
Swapping column 1 and 2
#2A((-2 1))
[9]> a
#2A((1 -2))
[10]> (setq a (make-array '(2 2) :initial-contents '((1 2) (1
   -3))))
#2A((1 2) (1 -3))
[11]> (swap-max-to-top-left a)
Swapping row 1 and 2
Swapping column 1 and 2
#2A((-3 1) (2 1))
[12]> a
#2A((1 2) (1 -3))
[13]> (setq a (make-array '(2 2) :initial-contents '((1 2) (-3
   1))))
#2A((1 2) (-3 1))
[14]> (swap-max-to-top-left a)
Swapping rows 1 and 2
#2A((-3 1) (1 2))
[15]> (setq a (make-array '(3 4) :initial-contents '((1.0 -2 3.0
   -4) (5.0 -6 7.0 -8) (9.0 -10 11.0 -12))))
#2A((1.0 -2 3.0 -4) (5.0 -6 7.0 -8) (9.0 -10 11.0 -12))
```

```
[16]> (swap-max-to-top-left a)
Swapping rows 1 and 3
Swapping columns 1 and 4
#2A((-12 -10 11.0 9.0) (-8 -6 7.0 5.0) (-4 -2 3.0 1.0))
[17]> (setq a (make-array '(3 4) :initial-contents '((1.0 -2 3.0
   -4) (5.0 -6 1234567 -8) (9.0 -10 11.0 -12))))
#2A((1.0 -2 3.0 -4) (5.0 -6 1234567 -8) (9.0 -10 11.0 -12))
[18]> (swap-max-to-top-left a)
Swapping rows 1 and 2
Swapping columns 1 and 3
#2A((1234567 -6 5.0 -8) (3.0 -2 1.0 -4) (11.0 -10 9.0 -12))
[19]> (setq a (make-array '(3 4) :initial-contents '((1.0 -2 3.0
   -4) (5.0 -6 -1234567 -8) (9.0 -10 11.0 -12))))
#2A((1.0 -2 3.0 -4) (5.0 -6 -1234567 -8) (9.0 -10 11.0 -12))
[20]> (swap-max-to-top-left a)
Swapping rows 1 and 2
Swapping columns 1 and 3
#2A((-1234567 -6 5.0 -8) (3.0 -2 1.0 -4) (11.0 -10 9.0 -12))
[21]> (setq a (make-array '(3 4) :initial-contents '((1 2 3 4)
   (12 6 7 8) (9 10 12 12))))
#2A((1 2 3 4) (12 6 7 8) (9 10 12 12))
[22]> (swap-max-to-top-left a)
Swapping row 1 and 2
#2A((12 6 7 8) (1 2 3 4) (9 10 12 12))
[23]> (setq a (make-array '(3 4) :initial-contents '((1 2 3 4)
   (5 6 12 12) (9 10 12 12))))
#2A((1 2 3 4) (5 6 12 12) (9 10 12 12))
[25]> (swap-max-to-top-left a)
Swapping row 1 and 2
Swapping column 1 and 3
#2A((12 6 5 12) (3 2 1 4) (12 10 9 12))
[26]> a
#2A((1 2 3 4) (5 6 12 12) (9 10 12 12))
```

9. Дневник отладки

Дата	Событие	Действие по исправлению	Примечание
10.05.2022	Функция изменяет значение матрицы, если она называется a	Использование let для создания локальной переменной	
10.05.2022	Неоправданное использование глобальных перемен	Замена setq на let	

10. Замечания автора по существу работы

Сложность полученного решения $O(n \cdot m)$. Пространственная сложность тоже $O(n \cdot m)$, поэтому добиться лучшей асимптотики нельзя.

При выполнении работы я заметил, что русские буквы плохо работают в Common Lisp, из-за этого я сделал вывод программы на английском.

11. Выводы

Я познакомился с векторами и массивами в Common Lisp. Язык поддерживает императивную парадигму в отличии от, например, Prolog, на котором работа с матрицами очень неприятная и сложная.