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

Студент группы 8О-308 МАИ *Марков Александр*, №15 по списку
Контакты: markov.lifeacc@gmail.com
Работа выполнена: 10.04.2021

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 №3.14)

Запрограммировать на языке Коммон Лисп функцию, принимающую два аргумента:

- A - двумерный массив, представляющий действительную матрицу произвольного размера,
- r - действительное число.

Функция должна найти наименьший элемент матрицы и вернуть новую матрицу того же размера, получающуюся из данной заменой всех вхождений наименьшего элемента на r.

Исходный массив должен оставаться неизменным.

## 4. Оборудование студента

Процессор AMD Ryzen 5 4600H 3.00 GHz, память: 16Gb, разрядность системы: 64.

## 5. Программное обеспечение

ОС Windows 10, среда LispWorks Personal Edition 7.1.2

# 6. Идея, метод, алгоритм

Задача была разбита на две подзадачи:

- Поиск минимального элемента. Для этого была реализована функция **find-min**;
- Создание нового двумерного массива и изменение всех входений наименьшего элемента. Для этого была реализована функция **change-arr**.

Функция **find-min** принимает двумерный массив, возвращает минимальный элемент и работает следующим образом:

- С помощью вложенных циклов рассматривается каждый элемент двумерного массива.
- Если рассматриваемый элемент меньше, чем текущий наименьший, то обновляем значение наименьшего элемента.

Функий **change-arr** принимает двумерный массив $arr$ и действительное число $r$, возвращает новый двумерный массив, в котором все вхождения наименьшего элемента массива $arr$ заменены на $r$. Работает следующим образом:

- Создается новый двумерный массив $new - arr$, имеющий размер такой же, что и у $arr$;
- С помощью функции **find-min** находится наименьший элемент массива $arr$;
- С помощью вложенных циклов рассматривается каждый элемент двумерного массива. Если этот элемент равен наименьшему, то на это же место в $new - arr$ ставится $r$. Иначе - элемент $arr$.

# 7. Сценарий выполнения работы

# 8. Распечатка программы и её результаты

## 8.1. Исходный код

```
(defun print-matrix (matrix &optional (chars 3) stream)
  (let ((*print-right-margin* (+ 6 (* (+ 1 chars)
   (array-dimension matrix 1)))))
    (pprint matrix stream)
    (values)
  )
)
```

```
(defun find-min (arr)
  (let ((m (array-dimension arr 0))
        (n (array-dimension arr 1))
        (min-el (aref arr 0 0))
       )
    (loop for i upfrom 0 to (- m 1)
      do
      (loop for j upfrom 0 to (- n 1)
        do (when (< (aref arr i j) min-el)
             (setf min-el (aref arr i j))
           )
      )
    )
    min-el
  )
)

(defun change-arr (arr r)
  (let* ((m (array-dimension arr 0))
         (n (array-dimension arr 1))
         (new-arr (make-array (list m n) :initial-element 0.0))
         (min-el (find-min arr))
        )
    (loop for i upfrom 0 to (- m 1)
      do
      (loop for j upfrom 0 to (- n 1)
        do (when (= (aref arr i j) min-el)
             (setf (aref new-arr i j) r)
           )
           (when (/= (aref arr i j) min-el)
             (setf (aref new-arr i j) (aref arr i j))
           )
      )
    )
    new-arr
  )
)
```

## 8.2. Результаты работы

```
CL-USER 17 > (print-matrix test1 6)
```

```
#2A((6.53 4321.4 431.44)
    (8.0 6.53 77.78)
    (123.12 88.132 6.53))

CL-USER 18 > (defvar result1 (change-arr test1 0.0))
RESULT1

CL-USER 19 > (print-matrix result1 6)

#2A((0.0 4321.4 431.44)
    (8.0 0.0 77.78)
    (123.12 88.132 0.0))

CL-USER 20 > (print-matrix test1 6)

#2A((6.53 4321.4 431.44)
    (8.0 6.53 77.78)
    (123.12 88.132 6.53))

CL-USER 21 > (print-matrix test2 6)

#2A((134.2 575.2 4321.4 55.55)
    (77.66 66.77 777.0 412.412)
    (431.44 12.33 33.12 1.0))

CL-USER 22 > (defvar result2 (change-arr test2 0.0))
RESULT2

CL-USER 23 > (print-matrix result2 7)

#2A((134.2 575.2 4321.4 55.55)
    (77.66 66.77 777.0 412.412)
    (431.44 12.33 33.12 0.0))

CL-USER 24 > (print-matrix test2 7)

#2A((134.2 575.2 4321.4 55.55)
    (77.66 66.77 777.0 412.412)
    (431.44 12.33 33.12 1.0))

CL-USER 25 > (print-matrix test3 4)
```

```
#2A((1.0 1.0 44.4)
    (5.0 21.2 41.4))

CL-USER 26 > (defvar result3 (change-arr test3 99.9))
RESULT3

CL-USER 27 > (print-matrix result3 4)

#2A((99.9 99.9 44.4)
    (5.0 21.2 41.4))

CL-USER 28 > (print-matrix test3 4)

#2A((1.0 1.0 44.4)
    (5.0 21.2 41.4))
```

# 9. Дневник отладки

| Дата | Событие | Действие по исправлению | Примечание |
|---|---|---|---|
| | | | |

# 10. Замечания автора по существу работы

Алгоритмически задача очень простая, но было достаточно интересно реализовать ее на Common Lisp.

# 11. Выводы

Массивы являются основополагающей структурой данных в программировании и часто используются, поскольку в них удобно хранить данные. При выполнение данной лабораторной работы я познакомился с массивами в языке Common Lisp, а также попрактиковался с циклами.