Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет: “Информационные технологии и прикладная математика”
Кафедра: 806 “Вычислительная математика и программирование”

Отчет по лабораторной работе №3

по курсу «Функциональное программирование»

Студент: Полей-Добронравова Амелия Вадимовна

Группа: М8О-307Б, № по списку 16.

Преподаватель: Иванов Д. А., доц. каф. 806

Дата: 28.04.2021

Итоговая оценка:

Подпись преподавателя:

Москва, 2021

1. Тема работы

Последовательности, массивы и управляющие конструкции Коммон Лисп.

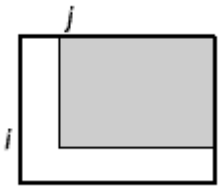
2. Цель работы

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

3. Задание (вариант № 3.24)

Запрограммировать на языке Коммон Лисп функцию, принимающую в качестве единственного аргумента двумерный массив, представляющий действительную матрицу A.

Функция должна возвращать новую матрицу B того же размера, каждый элемент которой b_{ij} равен наибольшему из элементов матрицы A, расположенных в области, определяемой индексами i и j и заштрихованной на рисунке.



4. Оборудование студента

MacBook Pro (15-inch, Mid 2012), процессор 2,3 GHz Intel Core i7, память 8 ГБ.

5. Программное обеспечение

macOS Mojave 10.14.6

6. Идея, метод, алгоритм

Нужно создать локальную переменную матрицы B, это делаю с помощью функции `make_matrix` принимающей в качестве аргумента матрицу, из которой возьмет размеры для новой матрицы.

Далее циклом пройдемся по всем элементам матрицы B, вызывая для их заполнения функцию `(find_max a i j)` принимающую на вход матрицу для поиска максимума и границы для поиска - i и j, находящую максимум в

необходимом прямоугольнике, и далее с помощью setf заменяем значение `(aref b i j)` на найденное.

7. Сценарий выполнения работы

Изучить методы работы с векторами и массивами в Common Lisp и написать программу. Протестировать её и отладить.

8. Распечатка программы и её результаты

Программа

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


(defun make_matrix (array &key
                    (element-type (array-element-type array))
                    (fill-pointer (and (array-has-fill-pointer-p array)
                                       (fill-pointer array)))
                    (adjustable (adjustable-array-p array)))
  (let* ((dimensions (array-dimensions array))
         (new-array (make-array dimensions
                                :element-type element-type
                                :adjustable adjustable
                                :fill-pointer fill-pointer)))
    new-array))


(defun find_max(a ii jj)
      (let ((max (aref a 0 jj)))
            (loop for i upfrom 0 below (+ 1 ii) do
            (loop with m = (array-dimension a 1)
                  for j upfrom jj below m do
```

```
                        (if (< max (aref a i j))
                        (setf max (aref a i j)))))
    max))

(defun max_matrix (a)
      (let ((b (make_matrix a)))
            (loop with n = (array-dimension a 0)
      for i upfrom 0 below n do
            (loop with m = (array-dimension a 1)
            for j upfrom 0 below m do
            (setf (aref b i j) (find_max a i j))))
    b))

(defvar a (make-array '(5 4)
            :initial-contents
            '((3 4 6 5)
              (1 3 3 2)
              (4 8 0 2)
              (3 2 9 3)
              (4 6 2 1))))
(print-matrix a)
(print-matrix (max_matrix a))
(print-matrix a)
```

Результаты

```
CL-USER 7 > (print-matrix (max_matrix a))

#2A((6 6 6 5)
    (6 6 6 5)
    (8 8 6 5)
    (9 9 9 5)
    (9 9 9 5))

CL-USER 8 > (print-matrix a)

#2A((3 4 6 5)
    (1 3 3 2)
    (4 8 0 2)
    (3 2 9 3)
    (4 6 2 1))
```

Красным цветом результат работы функции для нижней матрицы A

9. Дневник отладки

№ Дата, время Событие Действие по исправлению Примечание

10. Замечания автора по существу работы

У меня основные проблемы возникли не с самими матрицами, а скорее с синтаксисом циклов, и пониманием, что слово let в Common Lisp не копирует значение, а создаёт новое имя для него внутри программы.

11. Выводы

Мне показалось, что создание массивов в Common Lisp не удобное.
Во-первых потому, что при попытке сделать
(b (make_matrix ((array-dimension a 0) (array-dimension a 1)))) у меня вылетала ошибка, и вообще создание матриц с заранее неизвестным числовым значением довольно мудрено записывается.
В остальном способ работы очень напоминает другие языки и парадигмы программирования.