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

Студент группы 8О-306 Кузьмичев Александр, № по списку 12.

Контакты: alcuzmichev@yandex.ru  
Работа выполнена: 17.04.2021  
Преподаватель: Иванов Дмитрий Анатольевич, доц. каф. 806  
Отчет сдан:   
Итоговая оценка:   
Подпись преподавателя:

### 1. Тема работы

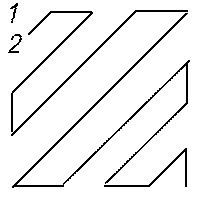
## Последовательности, массивы и управляющие конструкции Коммон Лисп

### 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 № 3.45)

### Запрограммировать на языке Коммон Лисп функцию, принимающую в качестве единственного аргумента целое число *n* - порядок матрицы. Функция должна создавать и возвращать двумерный массив, представляющий целочисленную квадратную матрицу порядка *n*, элементами которой являются числа 1, 2, ... n2, расположенные по схеме, показанной на рисунке.



### 4. Оборудование студента

Оборудование ПЭВМ студента.

### 5. Программное обеспечение

Программное обеспечение ЭВМ студента.

### 6. Идея, метод, алгоритм

Создадим функцию, которая возвращает список индексов конкретной диагонали. Затем пройдем по диагонали в нужном порядке и заполним.

### 7. Сценарий выполнения работы

### 8. Распечатка программы и её результаты

**Программа**

(defun get-diag(n diag-num)

(let ((diag-len (if (> diag-num n) (- (\* 2 n) diag-num) diag-num))

(diag-start (if (> diag-num n) (list (- diag-num n ) (- n 1)) (list 0 (- diag-num 1))))

(result-list NIL))

(loop for i upfrom 0 to (- diag-len 1)

do

(setq result-list (cons (list (+ (first diag-start) i) (- (second diag-start) i)) result-list))

)

result-list

)

)

(defun fill-diag(matrix diag cur)

(cond

(diag

(setf (aref matrix (first (first diag)) (second (first diag))) cur)

(fill-diag matrix (rest diag) (+ cur 1)))))

(defun diag-matrix(n)

(let ((matrix (make-array (list n n)))

(diag-count (- (\* 2 n) 1))

(reverse? nil)

(current-value 1)

)

(loop for i upfrom 1 to diag-count

do

(setq cur-diag (if reverse? (get-diag n i) (reverse (get-diag n i))))

(setq reverse? (not reverse?))

(fill-diag matrix cur-diag current-value)

(setq current-value (+ current-value (length cur-diag)))

)

matrix)

)

(diag-matrix 3)

(diag-matrix 4)

(diag-matrix 5)

**Результаты**

(print(diag-matrix 7)) =>

#2A((1 3 4 10 11 21 22)

(2 5 9 12 20 23 34)

(6 8 13 19 24 33 35)

(7 14 18 25 32 36 43)

(15 17 26 31 37 42 44)

(16 27 30 38 41 45 48)

(28 29 39 40 46 47 49))

(print(diag-matrix 9)) =>

#2A((1 3 4 10 11 21 22 36 37)

(2 5 9 12 20 23 35 38 53)

(6 8 13 19 24 34 39 52 54)

(7 14 18 25 33 40 51 55 66)

(15 17 26 32 41 50 56 65 67)

(16 27 31 42 49 57 64 68 75)

(28 30 43 48 58 63 69 74 76)

(29 44 47 59 62 70 73 77 80)

(45 46 60 61 71 72 78 79 81))

### 9. Дневник отладки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Дата, время | Событие | Действие по исправлению | Примечание |
| 1 |  |  |  |  |

### 10. Замечания автора по существу работы

Выполняя третью лабораторную работу по функциональному программированию, я в первый раз попробовал поработать с матрицами с помощью языка Common Lisp. Я по-прежнему считаю наиболее удобным языком для таких задач Python, но Common Lisp приятно удивил меня.

### 11. Выводы

В третьей лабораторной работе я поработал с матрицами и индексами, используя возможности Common Lisp. Мне пришлось разбить решение на два этапа (функции). На первом этапе я определял индексы для конкретной диагонали. На втором заполнял диагональ. Данная лабораторная работа помогла мне лучше почувствовать язык, его особенности и подводные камни.