

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

Студент группы 80-306 Пивницкий Даниэль Сергеевич, № по списку 13.

Контакты: pivnitskiydaniel@gmail.com

Работа выполнена: 06.06.2022

Преподаватель: Иванов Дмитрий Анатольевич, доц. каф. 806

Отчет сдан:

Итоговая оценка:

Подпись преподавателя:

1. Тема работы

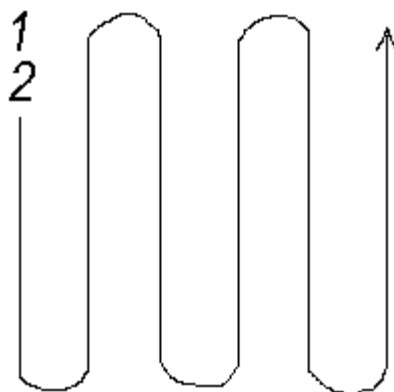
Последовательности, массивы и управляющие конструкции Коммон Лисп.

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 № 3.42)

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 $1, 2, \dots, n^2$, расположенные по схеме, показанной на рисунке.



4. Оборудование студента

Ноутбук Lenovo Legion 5 Pro, процессор Ryzen 5 5600h, память 16ГБ, 64-разрядная система.

5. Программное обеспечение

OS Windows 10, программа LispWorks Personal Edition 7.1.2

6. Идея, метод, алгоритм

Проход по матрице по столбцам вниз-вверх в цикле.

7. Сценарий выполнения работы

1. Изучение функций для работы с матрицами
2. Изучение функций для работы с циклами
3. Написание программы для заполнения матрицы

8. Распечатка программы и её результаты

Программа

```
(defun matrix-tl-br (n)
  (let ((matrix (make-array (list n n)))
        (count (ceiling n 2)) ; количество проходов вверх-вниз
        (cur_num 1))

    (dotimes (i count)
      ; заполняем столбец сверху вниз
      (loop for j from 0 upto (- n 1)
        do (setf (aref matrix j (* i 2)) cur_num)
            (setf cur_num (+ cur_num 1)))

      ; заполняем следующий столбец снизу вверх
      ; только если это не последний столбец матрицы нечетного
      порядка
      (unless (and (= (mod n 2) 1) (= i (- count 1)))
        (loop for j from (- n 1) downto 0
          do (setf (aref matrix j (+ (* i 2) 1)) cur_num)
              (setf cur_num (+ cur_num 1))))))

    matrix))

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

Результаты

```
CL-USER 1 > (defun matrix-tl-br (n)
  (let ((matrix (make-array (list n n)))
        (count (ceiling n 2)) ; количество проходов вверх-вниз
        (cur_num 1))

    (dotimes (i count)
      ; заполняем столбец сверху вниз
      (loop for j from 0 upto (- n 1)
        do (setf (aref matrix j (* i 2)) cur_num)
```

```

        (setf cur_num (+ cur_num 1)))

; заполняем следующий столбец снизу вверх
; только если это не последний столбец матрицы нечетного
порядка
(unless (and (= (mod n 2) 1) (= i (- count 1))))
(loop for j from (- n 1) downto 0
  do (setf (aref matrix j (+ (* i 2) 1)) cur_num)
      (setf cur_num (+ cur_num 1)))))

matrix))
MATRIX-TL-BR

CL-USER 2 >

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

CL-USER 3 > (print-matrix (matrix-tl-br 1))

#2A((1))

CL-USER 4 > (print-matrix (matrix-tl-br 2))

#2A((1 4)
     (2 3))

CL-USER 5 > (print-matrix (matrix-tl-br 3))

#2A((1 6 7)
     (2 5 8)
     (3 4 9))

CL-USER 6 > (print-matrix (matrix-tl-br 4))

#2A((1 8 9 16)
     (2 7 10 15)
     (3 6 11 14)
     (4 5 12 13))

CL-USER 7 > (print-matrix (matrix-tl-br 5))

#2A((1 10 11 20 21)
     (2 9 12 19 22)
     (3 8 13 18 23)
     (4 7 14 17 24)
     (5 6 15 16 25))

CL-USER 8 > (print-matrix (matrix-tl-br 10))

```

```
#2A ((1 20 21 40 41 60 61 80 81 100)
      (2 19 22 39 42 59 62 79 82 99)
      (3 18 23 38 43 58 63 78 83 98)
      (4 17 24 37 44 57 64 77 84 97)
      (5 16 25 36 45 56 65 76 85 96)
      (6 15 26 35 46 55 66 75 86 95)
      (7 14 27 34 47 54 67 74 87 94)
      (8 13 28 33 48 53 68 73 88 93)
      (9 12 29 32 49 52 69 72 89 92)
      (10 11 30 31 50 51 70 71 90 91))
```

```
CL-USER 9 > (print-matrix (matrix-tl-br 0))
```

```
#2A()
```

9. Дневник отладки

№	Дата, время	Событие	Действие по исправлению	Примечание
1				

10. Замечания автора по существу работы

Алгоритм для решения этой задачи составляется довольно тривиально. Основное время ушло на изучение синтаксиса языка Коммон Лисп для работы с матрицами и циклами.

Очень сильно помогла функция для печати матрицы, которая была дана в курсе.

11. Выводы

Выполняя данную лабораторную работу, я освоил функции для работы с двумерными массивами на языке Коммон Лисп. Я выполнил несложную задачу, но при этом попрактиковался в написании циклов и в работе с массивами.

Коммон Лисп предоставляет широкий спектр функций для работы с многомерными массивами.