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

Студент группы М8О-307Б-18 МАИ *Скворцов Кирилл Алексеевич*, №20
Контакты: kilyla2@yandex.ru
Работа выполнена: 25.04.2021

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

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

3. Задание (вариант №3.22)

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

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

4. Оборудование студента

Ноутбук Xiaomi mi Pro 15.6, процессор Intel Core i7-8550U CPU 1.80GHz, память: 8Gb, разрядность системы: 64.

5. Программное обеспечение

ОС Windows 10, онлайн компилятор для common-lisp, текстовый редактор VSCode (использовал т.к. там есть встроенный синтаксический валидатор).

6. Идея, метод, алгоритм

Создать матрицу такой же размерности, что и во входных данных. Каждый элемент матрицы - макимальный элемент в строке и столбце исходной матрицы. Пусть i - индекс строки, j - индекс столбца, тогда в новой матрице в позиции (i, j) будет стоять максимальный элемент из i-ой строки или j-го столбца. Для получения максимального элемента используется функция, получающая из исходной матрицы список всех элементов i-ой строки и j-го столбца исходной матрицы. Затем из этого списка извлекается максимум.

7. Сценарий выполнения работы

8. Распечатка программы и её результаты

8.1. Исходный код

```
(defun get_new_matrix (array)
  (let ((matrix (make-array (list (array-dimension array 0)
   (array-dimension array 1))))
        (cnt (ceiling (array-dimension array 1) 2))
        (num 1))
    (loop for i from 0 below (array-dimension array 0)
        do (loop for j from 0 below (array-dimension array 1)
            do (setf (aref matrix i j) (apply 'max
   (get_possible_values array i j)))
                  ))
       matrix))


(defun get_possible_values (matr i_index j_index)
    (append (loop for i from 0 below (array-dimension matr 0)
      collect (aref matr i j_index))
     (loop for j from 0 below (array-dimension matr 1)
      collect (aref matr i_index j))))
```

8.2. Результаты работы

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

```
#2A((2) (2) (2))
(print (get_new_matrix #2A((1 2 3 4 5))))
#2A((5 5 5 5 5))
(print (get_new_matrix #2A((1 2 3) (4 5 6) (7 8 9) (10 11 12))))
#2A((10 11 12) (10 11 12) (10 11 12) (12 12 12))
```

9. Дневник отладки

10. Замечания автора по существу работы

Замечаний нет.

11. Выводы

Благодаря данной лабораторной работе я познакомился с массивами в языке Lisp, а также узнал, как выполнять различные операции над ними, как использовать циклы. Массивы являются основополагающей структурой данных в программировании и часто используются, потому что в них удобно хранить данные. Работа с циклами после объектно-ориентированных языков оказалась довольно простой и интуитивно понятной.