Отчет по лабораторной работе № 5 по курсу «Функциональное программирование»

Студент группы 8О-307Б-18 МАИ *Тояков Артем*, №22 по списку
Контакты: temathesuper@mail.ru
Работа выполнена: 14.05.2021

Преподаватель: Иванов Дмитрий Анатольевич, доц. каф. 806
Отчет сдан:
Итоговая оценка:
Подпись преподавателя:

1. Тема работы

Обобщённые функции, методы и классы объектов.

2. Цель работы

Изучить обобщённые функции, методы и классы объектов, а также методы работы с ними в Коммон Лисп.

3. Задание (вариант № 5.23)

Задание: Определить обобщённую функцию и методы on-single-line3-p - предикат, принимающий в качестве аргументов три точки (радиус-вектора) и необязательный параметр tolerance (допуск), возвращающий T, если три указанные точки лежат на одной прямой (вычислять с допустимым отклонением tolerance).

Точки могут быть заданы как декартовыми координатами (экземплярами cart), так и полярными (экземплярами polar).

4. Оборудование студента

Процессор: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz, память: 3,8 Gb, разрядность системы: 64.

5. Программное обеспечение

UBUNTU 18.04.5 LTS, компилятор sbcl

6. Идея, метод, алгоритм

Идея в том, чтобы воспользоваться уравнением прямой, проходящей через 2 точки и подставить вместо x, y координаты третьей точки и подсчитать выражение, и если эта разность по модулю будет <= введённого значения tolerance, то условие выполняется и выводится T. Также стоит прописать отдельные методы для представления точек в декартовых и полярных координтах.

7. Сценарий выполнения работы

- Анализ возможных реализаций поставленной задачи на Коммон Лисп
- Изучение синтаксиса и основных функций работы с обобщёнными фукнциями и классами Коммон Лисп
- Реализация поставленной задачи на Коммон Лисп

8. Распечатка программы и её результаты

8.1. Исходный код

```
(defun square (x) (* x x))

(defclass cart ()                ; имя класса и надклассы
  ((x :initarg :x :reader cart-x)   ; дескриптор слота x
   (y :initarg :y :reader cart-y))) ; дескриптор слота y


(defmethod print-object ((c cart) stream)
    (format stream "[CART x:~d y:~d]"
        (cart-x c) (cart-y c)
    )
)

(defclass polar ()
  ((radius :initarg :radius :accessor radius)  ; длина >=0
   (angle  :initarg :angle  :accessor angle))) ; угол (-pi;pi]


(defmethod print-object ((p polar) stream)
  (format stream "[POLAR radius ~d angle ~d]"
          (radius p) (angle p)))
```

```
(defmethod radius ((c cart))
  (sqrt (+ (square (cart-x c))
           (square (cart-y c)))))


(defmethod angle ((c cart))
  (atan (cart-y c) (cart-x c))) ; atan2 в Си


(defmethod cart-x ((p polar))
  (* (radius p) (cos (angle p))))


(defmethod cart-y ((p polar))
  (* (radius p) (sin (angle p))))

(defun on-single-line3 (data)
(<= (abs (- (* (- (cart-x (second data)) (cart-x (first data)))
       (- (cart-y (third data)) (cart-y (first data))))
       (* (- (cart-x (third data)) (cart-x (first data)))
       (- (cart-y (second data)) (cart-y (first data)))))
    ) (fourth data)
)
)

(defgeneric on-single-line3-p (v1 v2 v3 &optional tolerance))

(defmethod on-single-line3-p ((v1 cart) (v2 cart) (v3 cart) &optional (tolerance 0.00
(on-single-line3 (list v1 v2 v3 tolerance)))

(defmethod on-single-line3-p ((v1 polar) (v2 polar) (v3 polar) &optional (tolerance 0
(on-single-line3 (list v1 v2 v3 tolerance)))
```

8.2. Результаты работы

```
* (setq v1 (make-instance 'cart :x 0 :y 0))

[CART x:0 y:0]
* (setq v2 (make-instance 'cart :x 1 :y 1))

[CART x:1 y:1]
* (setq v3 (make-instance 'cart :x 2 :y 2))
```

```
[CART x:2 y:2]
* (on-single-line3-p (v1 v2 v3 0.0001))

T
* (setq v4 (make-instance 'polar :radius 1 :angle (/ pi 4)))

[POLAR radius 1 angle 0.7853981633974483d0]
* (setq v5 (make-instance 'polar :radius 0 :angle 0))

[POLAR radius 0 angle 0]
* (setq v6 (make-instance 'polar :radius 0 :angle 2))

[POLAR radius 0 angle 2]
* (on-single-line3-p (v4 v5 v6))

NIL
```

9. Дневник отладки

Дата	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы

11. Выводы

В ходе данной работы мне удалось познакомиться с обобщёнными функциями, методами и классами в Коммон Лисп. Лисп был создан за пару десятилетий до того момента, когда объектно-ориентированное программирование стало популярным. Хотел бы отметить, что Коммон Лисп является полноценным объектно-ориентированным языком и в данном языке заложены основные парадигмы/принципы объектно-ориентированного программирования.