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

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

Контакты: alcuzmichev@yandex.ru  
Работа выполнена: 30.05.2021  
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

### Научиться определять простейшие классы, порождать экземпляры классов, считывать и изменять значения слотов, научиться определять обобщённые функции и методы.

### 3. Задание (вариант № 5.38)

Определите обобщённую функцию SUB2, производящую вычитание двух чисел либо многочленов.

(defgeneric sub2 (arg1 arg2)

...)

(defmethod sub2 ((p1 polynom) (p1 polynom))

...)

### 4. Оборудование студента

Оборудование ПЭВМ студента.

### 5. Программное обеспечение

Программное обеспечение ЭВМ студента.

### 6. Идея, метод, алгоритм

### Для начала предположим, что списки коэффициентов в многочленах отсортированы - это свойство несложно поддерживать и оно позволяет эффективнее проводить вычисления во многих случаях. Далее воспользуемся методом двух указателей - будем хранить индексы текущего обрабатываемого элемента для каждого массива. В случае, если термы по этим индексам имеют одинаковую степень - коэффициенты вычитаются, а индексы для каждого списка увеличиваются на 1. В ином случае, увеличивается индекс для списка, терм в котором имеет большую степень, а сам он записывается в результирующий список с изменением знака (если нужно). В конце из результирующего списка конструируется полином, который и является возвращаемым значением.

### 7. Сценарий выполнения работы

### 8. Распечатка программы и её результаты

**Программа**

(defun make-term(&key order coeff)

(list order coeff))

(defun order(term) (first term))

(defun coeff(term) (second term))

(defclass polynom ()

((var-symbol :initarg :var :reader var)

(term-list :initarg :terms :reader terms)))

(defgeneric zerop1 (arg)

(:method ((n number)) ; (= n 0)

(zerop n)))

(defgeneric minusp1 (arg)

(:method ((n number)) ; (< n 0)

(minusp n)))

(defmethod print-object ((p polynom) stream)

(format stream "[PL (~s) ~:{~:[~:[+~;-~]~d~[~2\*~;~s~\*~:;~s^~d~]~;~]~}]"

(var p)

(mapcar (lambda (term)

(list (zerop1 (coeff term))

(minusp1 (coeff term))

(if (minusp1 (coeff term))

(abs (coeff term))

(coeff term))

(order term)

(var p)

(order term)))

(terms p))))

(defgeneric sub2 (arg1 arg2))

(defmethod sub2((arg1 polynom) (arg2 polynom))

(let ((index1 0)(index2 0)(cur-term NIL))

(if (not (equal (var arg1) (var arg2)))

NIL

(make-instance 'polynom :var (var arg1) :terms (loop while (or (< index1 (length (terms arg1))) (< index2 (length (terms arg2))))

do

(setf cur-term

(cond ((and (< index1 (length (terms arg1))) (< index2 (length (terms arg2))))

(let((term1 (nth index1 (terms arg1)))(term2 (nth index2 (terms arg2))))

(cond ((= (order term1) (order term2)) (setf index1 (+ 1 index1)) (setf index2 (+ 1 index2)) (make-term :order (order term1) :coeff (- (coeff term1) (coeff term2))))

((< (order term1) (order term2)) (setf index2 (+ 1 index2)) (make-term :order (order term2) :coeff (- (coeff term2))))

((> (order term1) (order term2)) (setf index1 (+ 1 index1)) term1)

)

))

((< index1 (length (terms arg1))) (setf index1 (+ 1 index1)) (nth (- index1 1) (terms arg1)))

(else (setf index2 (+ 1 index2)) (make-term :order (order (nth (- index2 1) (terms arg2))) :coeff (- (coeff (nth (-index2 1) (terms arg2))))))

)

)

if (/= 0 (coeff cur-term)) collect cur-term

))

)

)

)

(sub2 (make-instance 'polynom :var 'x :terms '((5 2) (3 2) (1 4) (0 5)))

(make-instance 'polynom :var 'x :terms '((5 2) (4 1) (2 2) (1 3))))

**Результаты**

*CL-USER 10 >  
(sub2 (make-instance 'polynom :var 'x :terms '((5 2) (3 2) (1 4) (0 5)))  
(make-instance 'polynom :var 'x :terms '((5 2) (4 1) (2 2) (1 3))))  
[PL (X) -1X^4+2X^3-2X^2+1X+5]*

### 9. Дневник отладки

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

### 10. Замечания автора по существу работы

Выполняя пятую лабораторную работу по функциональному программированию, я написал самую трудоемкую и сложную программу на Common Lisp. Хотя, наиболее сложным этапом было продумать алгоритм, сама реализация тоже заняла какое-то время.

### 11. Выводы

Пятая лабораторная работа – последняя в курсе функционального программирования. Разумеется, язык Common Lisp не стал для меня прозрачным и полностью понятным, но я получил хорошее представления как о нем, так и о функциональном программировании в целом. Я уверен, что еще не раз столкнусь с ним, на Lisp, или на других языках.