Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт

по лабораторной работе №1

Название: Определение функций пользователя.

Дисциплина: Функциональное и логическое программирование

Студент	ИУ7-64Б		Л.Е.Тартыков
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			Н.Б.Толпинская
		(Подпись, дата)	(И.О. Фамилия)
Преподаватель			Ю.В.Строганов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

1 Теоретические вопросы

1.1 Базис Lisp

Базис - минимальный набор конструкций языка и структур данных, с помощью которых можно решить любую задачу.

Базис Lisp образуют: атомы, структуры, базовые функции, базовые функционалы.

Базисные функции – минимальный набор функций, которые позволяют решить любую задачу.

1.2 Классификация функций

1. Чистые математические функции.

   Имеют фиксированное число аргументов и возвращают один результат. Сначала вычисляются все аргументы, затем к ним применяется исходная функция.

2. Специальные функции (формы).

   Специальные функции – функции, у которых переменное число аргументов или они обрабатываются по-разному (один вычисляется, другой - нет).

3. Псевдофункции.

   Псевдофункции – функции, которые создают "спецэффекты"; например, вывод на экран.

4. Функции с вариантами значений – выбирают какое-то одно значение.

5. Функционалы.

   Функционалы – функции, которые в качестве аргументов используют функции или возвращают в качестве результата функцию. Также они называются *функциями более высокого порядка*. Позволяют создавать

синтаксически управляемые программы (программы, которые сами создают какие-то функции; эти функции затем выполняются).

6. Рекурсивные.

1.3 Способы создания функций

1. lambda-выражение. Данный способ представлен с помощью формулы (1.1).

$$(lambda\ \lambda\text{-список форма}), \tag{1.1}$$

где λ-список – список формальных параметров, форма – тело функции.

lambda-выражение не хранится в памяти и не имеет имени. Вычисляется сразу же. Используется для повторных вычислений.

Вызов lambda-функции выполняется по формуле (1.2).

$$(\lambda\text{-выражение последовательность форм}) \tag{1.2}$$

2. С помощью *defun* по формуле (1.3).

$$(defun\ f\ \lambda\text{-выражение}) \tag{1.3}$$

Система по имени символьного атома находит его определение.

1.4 Функции Car и Cdr

car и *cdr* являются базовыми функциями доступа к данным.

car – принимает на вход один аргумент и возвращает первый элемент списка.

Пример использования car и списка представлен на листинге 1.1.

Листинг 1.1 – Использование car и списка.

```
1   (car '(A B C D))      ;; (A)
```

car – принимает на вход один аргумент и возвращает хвост списка.

Пример использования cdr и списка представлен на листинге 1.2.

Листинг 1.2 – Использование cdr и списка.

```
(cdr '(A B C D))      ;; (B C D)
```

1.5 Назначение и отличие в работе Cons и List

cons – имеет два аргумента и возвращает бинарный узел. Если вторым аргументом является атом, то возвращается точечная пара; если список – список.

list – имеет произвольное число аргументов и возвращает список.

Пример использования cons представлен на листинге 1.3.

Листинг 1.3 – Использование cons.

```
(cons 'A 'B)            ;; (A.B)
(cons 'A '(B C D))      ;; (A B C D)
(cons '(A B) '(C D))    ;; ((A B) C D)
```

Пример использования list представлен на листинге 1.4.

Листинг 1.4 – Использование list.

```
(list 'A 'B)            ;; (A B)
(list 'A '(B C) 'D)     ;; (A (B C) D)
```

функция list может быть представлена с помощью cons (листинг 1.5).

Листинг 1.5 – Представление list с помощью cons.

```
(defun list_2 (ar1 ar2) (cons ar1 (cons ar2 ())))
(defun list_3 (ar1 ar2 ar3) (cons ar1 (cons ar2 (cons ar3 ()))))
...
```

2 Практические задания

2.1 Представить следующие списки в виде списочных ячеек: