Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных технология и систем»

Лабораторная работа №5

**«ФУНКЦИОНАЛЫ»**

по дисциплине:

«Функциональное и логическое программирование»

**Отчёт**

Принял преподаватель:

\_\_\_\_\_\_\_\_\_\_ / Михайлов Д.В

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Лехновский А. Д.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**Великий Новгород**

**2021**

1. **Цель и задачи данной лабораторной работы**

Целью лабораторной работы является изучение отображающих и применяющих функционалов.

1. **Задания на лабораторную работу**

**Задача №1:**

Написать программу обработки текста естественного языка с использованием отображающих функционалов в соответствии с заданием из таблицы. Текст рекомендуется представлять списком списков: каждое предложение – список слов, весь текст – список предложений.

*Дан текст. Вставить заданное новое слово после каждого вхождения другого заданного слова.*

**Задача №2:**

Дана фраза русского языка. Написать программу, которая разбивает каждое слово фразы на слоги.

**Задача №3:**

“Язык сплетника”. Есть ключевое слово, например, “сплетня”. Слово переводится на язык сплетника путем отделения первого слога в переводимом и ключевом слове (*например, сло-во и спле-тня*) с последующей перестановкой по определенным правилам*: ‘(слово сплетня)* преобразуется в *‘(сплево слотня)*.

Каждое слово преобразуется в пару слов. Первое слово есть конкатенация первого слога ключевого слова и части переводимого слова, оставшейся после отделения от него первого слога. Второе слово есть конкатенация первого слога переводимого слова и части ключевого слова, оставшейся после отделения от него первого слога. Написать программу перевода предложения русского языка на заданный таким образом “тайный” язык.

**Задача №4:**

Написать программу, исключающую в исходном тексте из каждого слова его окончание по словарю. Словарь окончаний представлять списком строк.

1. **Решение поставленных задач**

**Задача №1:**

Функция task\_1 запускает нашу программу, где мы на входе поддаем исходный текст (список предложений), ключевое слово и вставку (то есть слово, которое должно быть вставлено после каждого вхождения ключевого).

В функции myFunc мы работаем с каждым предложением отдельно, затем в функции inSentence разбиваем данное предложение на слова и к каждому слову применяем функцию inWord, в которой проверяется данное слово на совпадение с ключевым, в случае успеха производится вставка после вхождения ключевого слова, в противном случае мы переходим к следующему слову в предложении. Программа была реализована с использованием отображающего функционала.

Код программы для решения задачи приведен в *Приложении 1*.

|  |  |
| --- | --- |
| *Входные данные* | *Результат, полученный с помощью программы* |
| (task\_1 '(("House" "guy" "floor")) "house" "home") | (("House" "home" "guy" "floor")) |
| (task\_1 '(("HOUSE" "guy" "floor")) "house" "home") | (("HOUSE" "home" "guy" "floor")) |
| (task\_1 '(("house" "guy" "floor" "house")) "house" "home") | (("house" "home" "guy" "floor" "house" "home")) |
| (task\_1 '(("house" "guy" "floor" "house") ("wake" "me" "up" "inside")) "home" "house") | (("house" "guy" "floor" "house") ("wake" "me" "up" "inside")) |
| (task\_1 '(("dog" "sweet" "boy" "\_" "boy" "\_" "boy" "\_" "dreams" "little") ("my" "favorite" "BOY")) "boy" "girl") | (("dog" "sweet" "boy" "girl" "\_" "boy" "girl" "\_" "boy" "girl" "\_" "dreams" "little")  ("my" "favorite" "BOY" "girl")) |
| (task\_1 '(("добрый" "вечер" "господа" "ВЕЧЕР" "мадам" "утро" "вечер")("Вечер" "тяжкий")) "вечер" "темно") | (("добрый" "вечер" "темно" "господа" "ВЕЧЕР" "темно"  "мадам" "утро" "вечер" "темно")  ("Вечер" "темно" "тяжкий")) |

*Таблица №1. Тестовые наборы данных.*

**Задача №2:**

Была реализована программа, которая разбивает каждое слово фразы на слоги.

Правило слогораздела включает в себя следующие принципы:

* Сочетание двух шумных или двух сонорных согласных отходит к последующему слогу.
* Сочетание шумного и сонорного отходит к последующему слогу.

- Сочетание сонорного и шумного имеет раздел посередине сочетания.

Код программы для решения задачи приведен в *Приложении 2*.

|  |  |
| --- | --- |
| *Входные данные* | *Результат, полученный с помощью программы* |
| (task2 '(("пушка" "изба" "волна"))) | ("пу-шка" "и-зба" "во-лна") |
| (task2 '(("война" "майка"))) | ("вой-на" "май-ка") |
| (task2 '(("конспект" "молва"))) | ("кон-спект" "мол-ва") |
| (task2 '(("воробей" "богатство" "геркулес" "жигули"))) | ("во-ро-бей" "бо-га-тство" "гер-ку-лес" "жи-гу-ли") |
| (task2 '(("рукомойник" "война" "отец" "салфетка" "купэ" "конспект" "пушка" "волна" "творец" "крыша" "морс" "транспорт" "лейка" "ясли"))) | ("ру-ко-мой-ник" "вой-на" "о-тец" "сал-фе-тка" "ку-пэ"  "кон-спект" "пу-шка" "во-лна" "тво-рец" "кры-ша" "морс"  "тран-спорт" "лей-ка" "я-сли") |

*Таблица №2. Тестовые наборы данных.*

**Задача №3:**

В основу этой программы легла программа из предыдущей задачи, так как нам снова придется работать со слогами. Была добавлена функция Func, которая делает следующее:

Каждое слово преобразуется в пару слов. Первое слово есть конкатенация первого слога ключевого слова и части переводимого слова, оставшейся после отделения от него первого слога. Второе слово есть конкатенация первого слога переводимого слова и части ключевого слова, оставшейся после отделения от него первого слога.

Код программы для решения задачи приведен в *Приложении 3*.

|  |  |
| --- | --- |
| *Входные данные* | *Результат, полученный с помощью программы* |
| (task3 '(("сплетня")) '(("слово"))) | ("сплево" "слотня") |
| (task3 '(("рукомойник")) '(("салфетка"))) | ("руфетка" "салкомойник") |
| (task3 '(("сплетня" "дети")) '(("слово"))) | ("сплево" "слотня" "дево" "слоти") |
| (task3 '(("сплетня" "война" "кукушка")) '(("секрет"))) | ("сплекрет" "сетня" "войкрет" "сена" "кукрет" "секушка") |
| (task3 '(("сплетня" "война" "малыш" "добро" "сила")) '(("слово"))) | ("сплево" "слотня" "войво" "слона" "маво" "слолыш"  "дово" "слобро" "сиво" "слола") |

*Таблица №3. Тестовые наборы данных.*

**Задача №4:**

Функция task4 запускает нашу программу, где мы на входе поддаем исходный текст (список предложений) и словарь окончаний, представленный списком строк. Основная идея данной программы является создание обработчика слов, в котором проверяется на совпадение все элементы из словаря окончаний и окончание данного слова, которое получается реверсированием последовательности букв в слове и делением этой последовательности на N (длина слова) частей, где каждая последующая часть больше на 1 предыдущей (= 1, = 2, ..., = N), затем сравнивается полученные реверсивные части слова(где каждая часть больше предыдущей) с словарём, где окончания представлены в реверсивном виде. После этого нам возвращают длину окончания, которую надо будет отрезать, иными словами, при первом совпадении, выдаётся значение, сколько необходимо срезать букв с конца слова.

Код программы для решения задачи приведен в *Приложении 4*.

|  |  |
| --- | --- |
| *Входные данные* | *Результат, полученный с помощью программы* |
| (task4 '(("добрая" "человек" "малина" "ненасытный" "счастливый")) '("ый")) | (("добрая" "человек" "малина" "ненасытн" "счастлив")) |
| (task4 '(( "человек" "малина" "он" "счастливый")) '("ая")) | (("человек" "малина" "он" "счастливый")) |
| (task4 '(( "человек" "малина" "он" "счастливый")) '("ая" "ый" "а")) | (("человек" "малин" "он" "счастлив")) |
| (task4 '(( "человек" "малина" "он" "счастливый") ("хватит" "пугающе" "страшно" "мы" "мириться" )) '("ая" "ый" "а" "ся")) | (("человек" "малин" "он" "счастлив") ("хватит" "пугающе"  "страшно" "мы" "мирить")) |
| (task4 '(("подниматься" "веселый" "тебе" "сильно" "нравиться" "добрый" "молодая" "парнишка" "ружей") ("малый" "добрейший" "мама" "папа" "детей")) '("ся" "ий" "ый" "ая" "а" "ей")) | (("поднимать" "весел" "тебе" "сильно" "нравить" "добр"  "молод" "парнишк" "руж")  ("мал" "добрейш" "мам" "пап" "дет")) |

*Таблица №4. Тестовые наборы данных.*

1. **Вывод**

В данной лабораторной работе, я познакомился с отображающим функционалом и выполнил лабораторные работы по работе с текстом.

**Приложение 1**

Код программы:

;Дан текст. Вставить заданное новое слово после каждого вхождения другого

;заданного слова.

;на входе поддаем исходный текст (список предложений), ключевое слово и вставку (то есть слово, которое должно быть вставлено после каждого вхождения ключевого).

(define (task\_1 text keyword insertword)

(if (empty? text)

"empty"

(map flat (myFunc text keyword insertword 0))

)

)

;работаем с каждым предложением отдельно

(define (myFunc text keyword insertword i)

(if (< i (length text))

(cons (inSentence text keyword insertword i) (myFunc text keyword insertword (+ i 1)))

'()

)

)

;разбиваем данное предложение на слова и к каждому слову применяем функцию inWord

(define (inSentence text keyword insertword i)

(list (inWord (nth i text) keyword insertword 0))

)

;проверяется данное слово на совпадение с ключевым, в случае успеха производится вставка после вхождения ключевого слова, в противном случае мы переходим к следующему слову в предложении.

(define (inWord predloj keyword insertword i)

(if (< i (length predloj))

(cond

((= (lower-case (nth i predloj)) (lower-case keyword)) (cons (nth i predloj) (cons insertword (inWord predloj keyword insertword (+ i 1))))

)

(true (cons (nth i predloj) (inWord predloj keyword insertword (+ i 1)))

)

)

'()

)

)

(task\_1 '(("House" "guy" "floor")) "house" "home")

(task\_1 '(("HOUSE" "guy" "floor")) "house" "home")

(task\_1 '(("house" "guy" "floor" "house")) "house" "home")

(task\_1 '(("house" "guy" "floor" "house") ("wake" "me" "up" "inside")) "home" "house")

(task\_1 '(("dog" "sweet" "boy" "\_" "boy" "\_" "boy" "\_" "dreams" "little") ("my" "favorite" "BOY")) "boy" "girl")

(task\_1 '(("добрый" "вечер" "господа" "ВЕЧЕР" "мадам" "утро" "вечер")("Вечер" "тяжкий")) "вечер" "темно")

**Приложение 2**

Код программы:

(define (task2 text)

(if (empty? text)

"empty"

(inSentence text 0)

)

)

(define (inSentence text i)

(chop (inWord (nth i text) 0))

)

(define (inWord sentence i)

(if (< i (length sentence))

(cons

(join (addTire (join (inLetter1 (nth i sentence) 0))

(join (inLetter2 (nth i sentence) 0)) 0))

(inWord sentence (+ i 1))

)

)

)

;Перевод слова в набор цифр

(define (inLetter1 word)

(map letterToNumber (inLetter2 word 0))

)

;Разбиваем слова на отдельные буквы

(define (inLetter2 word i)

(if (< i (utf8len word))

(flat (list (nth i word) (inLetter2 word (+ i 1))))

'()

)

)

;Задаём каждой букве цифру 1-4

(define (letterToNumber lit)

(cond

;Шумные согласные

((or (= lit "п")

(= lit "ф")

(= lit "т")

(= lit "с")

(= lit "ш")

(= lit "к")

(= lit "ц")

(= lit "х")

(= lit "б")

(= lit "в")

(= lit "г")

(= lit "д")

(= lit "ж")

(= lit "з")

(= lit "щ")

(= lit "ч")) "1")

;Сонорные согласные

((or (= lit "м")

(= lit "н")

(= lit "л")

(= lit "р")) "2")

;Гласные

((or (= lit "а")

(= lit "о")

(= lit "у")

(= lit "ы")

(= lit "и")

(= lit "э")

(= lit "е")

(= lit "я")

(= lit "ё")

(= lit "ю")) "3")

;Й

((or (= lit "й")) "4")

)

)

;Функция добавления тире для деления слова на слоги по правилам

(define (addTire numlst litlst i)

(if (< i (length numlst))

(if (= (nth i numlst) "3") ;Если гласная

(if (> (+ i 3) (length numlst)) ;Если после гласной меньше двух букв

(if (= (+ i 1) (length numlst))

(flat(list(nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

)

(if (and(= (nth (+ i 1) numlst) "1")(= (nth (+ i 2) numlst) "1"));Правило 1 (Сочетание двух шумных или двух сонорных согласных отходит к последующему слогу.)

(if (= (+ i 3) (length numlst))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

)

(if (and(= (nth (+ i 1) numlst) "1")(= (nth (+ i 2) numlst) "2")) ;Правило 2 (Сочетание шумного и сонорного отходит к последующему слогу.)

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

(if (and(= (nth (+ i 1) numlst) "2")(= (nth (+ i 2) numlst) "1")) ;Правило 3 (Сочетание сонорного и шумного имеет раздел посередине сочетания.)

(if (= (+ i 3) (length numlst))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) (nth (+ i 1) litlst) "-" (addTire numlst litlst (+ i 2))))

)

(if (or (and(= (nth (+ i 1) numlst) "4") (= (nth (+ i 2) numlst) "2")) (and (= (nth (+ i 1) numlst) "4") (= (nth (+ i 2) numlst) "1"))) ;Правило 4 (Для случая с Й)

(flat(list (nth i litlst) (nth (+ i 1) litlst) "-" (addTire numlst litlst (+ i 2))))

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

)

)

)

)

)

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

)

'()

)

)

(task2 '(("пушка" "изба" "волна")))

(task2 '(("война" "майка")))

(task2 '(("конспект" "молва")))

(task2 '(("воробей" "богатство" "геркулес" "жигули")))

(task2 '(("рукомойник" "война" "отец" "салфетка" "купэ" "конспект" "пушка" "волна" "творец" "крыша" "морс" "транспорт" "лейка" "ясли")))

**Приложение 3**

Код программы:

;Кодируемый текст ;переводимое слово

(define (task3 keyword word)

(if (empty? keyword)

"empty"

(Func (parse (nth 0 (inSentence keyword 0))"-") (parse (nth 0 (inSentence word 0))"-") 0 (inSentence keyword 0))

)

)

;Первое слово есть конкатенация первого слога ключевого слова и части переводимого слова, оставшейся после отделения от него первого слога. Второе слово есть конкатенация первого слога переводимого слова и части ключевого слова, оставшейся после отделения от него первого слога.

(define (Func lstkeyword lstword i sentence )

(if (< (+ 1 i) (length sentence))

(flat(list (join(flat(list (nth 0 lstkeyword) (rest lstword))))

(join(flat(list (nth 0 lstword) (rest lstkeyword))))

(Func (parse (nth (+ 1 i) sentence)"-") lstword (+ 1 i) sentence)))

(flat(list (join(flat(list (nth 0 lstkeyword) (rest lstword))))

(join(flat(list (nth 0 lstword) (rest lstkeyword))))))

)

)

(define (inSentence text i)

(chop (inWord (nth i text) 0))

)

(define (inWord sentence i)

(if (< i (length sentence))

(cons

(join (addTire (join (inLetter1 (nth i sentence) 0))

(join (inLetter2 (nth i sentence) 0)) 0))

(inWord sentence (+ i 1))

)

)

)

;Перевод слова в набор цифр

(define (inLetter1 word)

(map letterToNumber (inLetter2 word 0))

)

;Разбиваем слова на отдельные буквы

(define (inLetter2 word i)

(if (< i (utf8len word))

(flat (list (nth i word) (inLetter2 word (+ i 1))))

'()

)

)

(define (letterToNumber lit)

(cond

;Шумные согласные

((or (= lit "п")

(= lit "ф")

(= lit "т")

(= lit "с")

(= lit "ш")

(= lit "к")

(= lit "ц")

(= lit "х")

(= lit "б")

(= lit "в")

(= lit "г")

(= lit "д")

(= lit "ж")

(= lit "з")

(= lit "щ")

(= lit "ч")) "1")

;Сонорные согласные

((or (= lit "м")

(= lit "н")

(= lit "л")

(= lit "р")) "2")

;Гласные

((or (= lit "а")

(= lit "о")

(= lit "у")

(= lit "ы")

(= lit "и")

(= lit "э")

(= lit "е")

(= lit "я")

(= lit "ё")

(= lit "ю")) "3")

;Й

((or (= lit "й")) "4")

)

)

(define (addTire numlst litlst i)

(if (< i (length numlst))

(if (= (nth i numlst) "3") ;Если гласная

(if (> (+ i 3) (length numlst)) ;Если после гласной меньше двух букв

(if (= (+ i 1) (length numlst))

(flat(list(nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

)

(if (and(= (nth (+ i 1) numlst) "1")(= (nth (+ i 2) numlst) "1"));Правило 1 (Сочетание двух шумных или двух сонорных согласных отходит к последующему слогу.)

(if (= (+ i 3) (length numlst))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

)

(if (and(= (nth (+ i 1) numlst) "1")(= (nth (+ i 2) numlst) "2")) ;Правило 2 (Сочетание шумного и сонорного отходит к последующему слогу.)

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

(if (and(= (nth (+ i 1) numlst) "2")(= (nth (+ i 2) numlst) "1")) ;Правило 3 (Сочетание сонорного и шумного имеет раздел посередине сочетания.)

(if (= (+ i 3) (length numlst))

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

(flat(list (nth i litlst) (nth (+ i 1) litlst) "-" (addTire numlst litlst (+ i 2))))

)

(if (or (and(= (nth (+ i 1) numlst) "4") (= (nth (+ i 2) numlst) "2")) (and (= (nth (+ i 1) numlst) "4") (= (nth (+ i 2) numlst) "1"))) ;Правило 4 (Для случая с Й)

(flat(list (nth i litlst) (nth (+ i 1) litlst) "-" (addTire numlst litlst (+ i 2))))

(flat(list (nth i litlst) "-" (addTire numlst litlst (+ i 1))))

)

)

)

)

)

(flat(list (nth i litlst) (addTire numlst litlst (+ i 1))))

)

'()

)

)

(task3 '(("сплетня")) '(("слово")))

(task3 '(("рукомойник")) '(("салфетка")))

(task3 '(("сплетня" "дети")) '(("слово")))

(task3 '(("сплетня" "война" "кукушка")) '(("секрет")))

(task3 '(("сплетня" "война" "малыш" "добро" "сила")) '(("слово")))

**Приложение 4**

Код программы:

;Написать программу, исключающую в исходном тексте из каждого слова его окончание по словарю. Словарь окончаний представлять списком строк.

(define (task4 text slovar)

(if (empty? text)

"empty"

(startFunc text (reverseEnding slovar 0) 0)

)

)

(define (startFunc text slovar i)

(if (< i (length text))

(cons (inSentence text slovar i) (startFunc text slovar (+ i 1)))

'()

)

)

(define (inSentence text slovar i)

(inWord (nth i text) slovar 0)

)

(define (inWord sentence slovar i)

(if (< i (length sentence))

(cons (join (clipping (inLetter2 (nth i sentence) 0) slovar)) (inWord sentence slovar (+ i 1)))

'()

)

)

(define (reverseEnding sentence i)

(if (< i (length sentence))

(cons(join (reverse (inLetter2 (nth i sentence) 0) ))(reverseEnding sentence (+ i 1)))

'()

)

)

;Обрезает окончание нужной длины

(define (clipping lst revslovar)

(chop lst (countCut lst revslovar))

)

;Возвращает длину окончания, которую надо будет отрезать

(define (countCut lst revslovar)

(cut (createEnding (reverse lst) revslovar 0) revslovar 0)

)

;Функция cut сравнивает полученные реверсивные части слова(где каждая часть больше предыдущей) с словарём, где окончания представлены в реверсивном виде.

(define (cut lst revslovar i)

(if (< i (length lst))

(if (nil? (find (nth i lst) revslovar))

(if (= (+ i 1) (length lst))

0

(cut lst revslovar (+ i 1))

)

(+ i 1)

)

)

)

;в данную функцию в качестве аргументов поддается реверсивная последовательность букв исходного слова, словарь, где окончания представлены в реверсивном виде, а также счётчик.Функция возвращает n частей слова, состоящего из n букв. Причём каждая следующая часть больше предыдущей на 1.

(define (createEnding revlst revslovar i)

(if (< i (length revlst))

(cons (join (getEnding revlst i)) (createEnding revlst revslovar (+ i 1)))

'()

)

)

;Данная функция возвращает часть слова N0 = 1, N1 = 2, ..., N(N-1) = N

(define (getEnding revlst i)

(if (>= i 0)

(cons (first revlst) (getEnding (rest revlst) (- i 1)))

'()

)

)

;Разбиваем слова на отдельные буквы

(define (inLetter2 word i)

(if (< i (utf8len word))

(flat (list (nth i word) (inLetter2 word (+ i 1))))

'()

)

)

(task4 '(("добрая" "человек" "малина" "ненасытный" "счастливый")) '("ый"))

(task4 '(( "человек" "малина" "он" "счастливый")) '("ая"))

(task4 '(( "человек" "малина" "он" "счастливый")) '("ая" "ый" "а"))

(task4 '(( "человек" "малина" "он" "счастливый") ("хватит" "пугающе" "страшно" "мы" "мириться" )) '("ая" "ый" "а" "ся"))

(task4 '(("подниматься" "веселый" "тебе" "сильно" "нравиться" "добрый" "молодая" "парнишка" "ружей") ("малый" "добрейший" "мама" "папа" "детей")) '("ся" "ий" "ый" "ая" "а" "ей"))