# МИНОБРНАУКИ РОССИИ

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

# «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

**Кафедра Вычислительной техники**

# ОТЧЕТ

**по лабораторной работе № 9**

# по дисциплине «Программирование» ТЕМА: **«Линейные двусвязные списки»**

|  |  |
| --- | --- |
| Студент гр. 0310 | Савенко Н.С. |
| Преподаватель | Хахаев И. А. |

Санкт-Петербург 2021

# Цель работы.

Целью работы является научится использовать односвязные списки для решения поставленных задач

# Задание (вариант 15)

# Разработать подалгоритм и написать функцию, удаляющую в линейном двусвязном списке элемент после первого с начала списка элемента с заданным значением числового поля. Если найденный элемент является последним, вывести сообщение о невозможности удаления.

# Постановка задачи и описание решения

# Задачей было написать программу считывающую данные из CSV файла, принимающую значение и удаляющую элемент в списке после элемента с этим значением

# 

# Описание переменных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменно | Тип | Назначение | Функция |
| 1 | string | char\* | Хранит в себе строку из файла | main |
| 2 | s2 | char\*\* | Хранит данные DUCKDUCK |
| 3 | sep | char | Хранит символ разделитель |
| 4 | count | int | Хранит количество строк в файле |
| 5 | i/j | int | Используется в цикле for |
| 6 | df | FILE\* | Переменная для совершения действий с файлами |
| 7 | duckPtr | DUCKDUCK\* | Указатель на утку |
| 8 | head | LHead | Хранит в себе данные головы списка |
| 9 | slen | int | Длина строки |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | i/j/k/m | int | Итерируемые значения в циклах | simple\_split |
| 2 | str\_array | \*\* char | Массив разделенных строк |
| 3 | key | int | Маркер успешности |
| 4 | count | Int | Количество строк |
| 5 | Str | \* char | Строка для разделения |
| 6 | Length | Int | Длинна строки |
| 7 | sep | Char | Символ делитель |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | my\_head | \*LHead | Голова | delete\_node |
| 2 | current\_node | \*DUCKDUCK | Текущая утка |
| 3 | q/q1 | \*DUCKDUCK | Буферные значения |
| 1 | q/q1 | \*DUCKDUCK | Буферные значения | clear\_list |
| 2 | my\_head | \*LHead | Голова |
| 1 | duckPtr | DUCKDUCK\* | Указатель на утку | delete\_target |
| 2 | choosen\_value | int | Выбранное значение |
| 3 | success | int | Маркер успешности |

**Схема алгоритма**

См. файл 0310Savenko- lab09.jpg

**Контрольные примеры**

Все контрольные примеры предполагают наличие csv файла, предоставляемого вместе с программой.

Пример 1:

Входные данные:

13

Csv:

Last James;mallard;13;10;1.5;2.5;2;2

Some James;mallard;14;12;1.55;2.5;2;2

New James;mallard;15;13;1.45;2.5;2;2

last James;mallard;16;14;1.37;2.5;2;2

Last Kyle;rubber;17;15;1.98;2.5;2;2

Last Mike;rubber;19;17;1.1;2.5;2;2

Выходные данные:

Source duck Xpos: 13

Source duck Xpos: 15

Source duck Xpos: 16

Source duck Xpos: 17

Source duck Xpos: 19

Пример 2:

Входные данные:

19

Csv:

Last James;mallard;13;10;1.5;2.5;2;2

Some James;mallard;14;12;1.55;2.5;2;2

New James;mallard;15;13;1.45;2.5;2;2

last James;mallard;16;14;1.37;2.5;2;2

Last Kyle;rubber;17;15;1.98;2.5;2;2

Last Mike;rubber;19;17;1.1;2.5;2;2

Выходные данные:

Cant perform action

No elements found

Finished list:

Source duck Xpos: 13

Source duck Xpos: 14

Source duck Xpos: 15

Source duck Xpos: 16

Source duck Xpos: 17

Source duck Xpos: 19

# Примеры выполнения программы

Все примеры предусматривают изначально заготовленный файл csv

# Пример 1:

# 

# 

# Пример 2:

# 

**Выводы.**

Научился использовать двусвязные списки.