**Отчет по лабораторной работе №12**

**Тема**

**Списки**

*РК6-16Б*

16. Петраков Станислав Альбертович

Преподаватель: Федорук Елена Владимировна

**Цель работы**

Изучение организации структур в языке Си. Изучение функций управления динамической памятью. Применение этих функций для организации динамических связных списков. Изучение функций работы с файлом.

**Ответы на контрольные вопросы**

1. *Как организуется линейный двусвязный список?*

Каждый узел двунаправленного (двусвязного) линейного списка (ДЛС) содержит два поля указателей — на следующий и на предыдущий узлы. Указатель на предыдущий узел корня списка содержит нулевое значение. Указатель на следующий узел последнего узла также содержит нулевое значение.

1. *Что такое буферизованый ввод-вывод? Для чего он используется?*

Буферизация - способ организации ввода-вывода в программе, позволяющий минимизировать число обращений к устройству. При исчерпании буфера вновь выполняется чтение с диска, и буфер пополняется. Обращение к диску лишь при необходимости обеспечивает существенную экономию времени. Программисты могут читать из файла множеством способов (посимвольно, построчно и т.д.) и не задумываться о минимизации количества обращений к диску. Стандартный пакет ввода-вывода гарантирует решение этой проблемы.

1. *Что такое структура FILE?*

Структура FILE — это структура, в которой хранится информация о том, как открыт файл (например, для чтения), и где позиция следующего чтения или записи.

1. *Как открывается файл?*

С помощью функции fopen().FILE \*fopen(const char \*имя\_файла, const char \*тип)/

1. *Какие функции организуют чтение из файла?*

fgetc – для одного символа, fgets – читает n количество символов, fscanf – аналогично, как и scanf, fread – читает из потока в массив.

1. *Какие функции организуют запись в файл?*

fputc – помещает 1 символ, fputs – помещает строку, fprintf – аналогично printf, fwrite – из массива в поток.

1. *Для чего надо закрывать файл?*

Во-первых, при этом освобождается место в памяти, которое может быть использовано для открытия нового файла. Во-вторых, буфер, если он есть, записывается за приемлемое время. Одновременно в программе может быть открыто не более 20 файлов.

**Программа**

*Задание:* *Разработайте программу, которая читает из стандартного потока ввода слова и размещает их в односвязный список следующим образом. Если слово встретилось первый раз, то для него добавляется новый элемент в конец списка. В противном случае в соответствующем элементе списка увеличивается счетчик слов.*

*После завершения ввода содержимое списка записываются в файл, имя которого задается в командной строке. Необходимо организовать грамотную работу с файлом.*

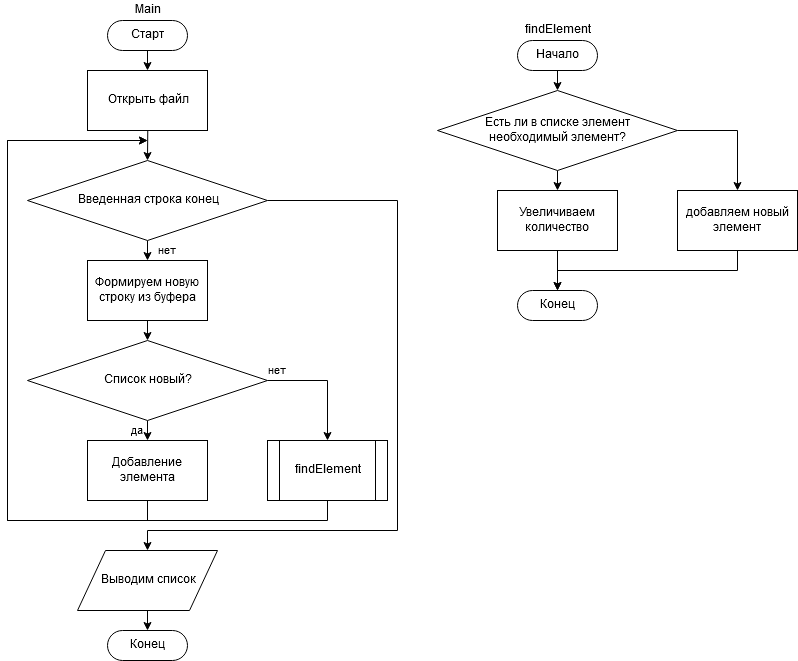
*Элемент списка — структура, имеющая следующие поля:*

*1. Адрес строки, в которой хранится слово*

*2. Количество повторений данного слова в потоке*

*3. Указатель на следующий элемент списка.*

**Блок-схема:**



**Сама программа:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define BUF\_SIZE 1024

typedef struct element

{

char\* word;

int amount;

struct element\* next;

}element\_s;

element\_s\* addElement(char\* str)

{

element\_s\* temp;

if ((temp = (element\_s\*)malloc(sizeof(element\_s))) == NULL)

{

printf("Memory allocation error. Terminate.\n");

exit(2);

}

temp->word = str;

temp->amount = 1;

temp->next = NULL;

return temp;

}

void findElement(element\_s\* first, char\* word)

{

element\_s\* last = NULL;

for (; first != NULL; first = first->next)

{

if (!strcmp(first->word, word))

{

first->amount++;

return;

}

if (first->next == NULL)

last = first;

}

last->next = addElement(word);

return;

}

void output(element\_s\* first, FILE\* file)

{

while (first != NULL)

{

fprintf(file, "%d: %s\n", first->amount, first->word);

element\_s\* forDelete = first;

first = first->next;

free(forDelete);

}

}

int main(int argc, char\* argv[])

{

if (argc != 2)

{

printf("No such arguments. Terminate.\n");

exit(2);

}

FILE\* file;

if ((file = fopen(argv[1], "w+")) == NULL)

{

printf("Cannot open file. Terminate");

exit(3);

}

//variables for buffer

char buffer[BUF\_SIZE];

//variables for list

element\_s\* Core\_Element = NULL;

while (scanf("%s", buffer) != EOF && strcmp(buffer, "~"))

{

char\* temp;

if ((temp = (char\*)malloc(strlen(buffer) \* sizeof(char))) == NULL)

{

printf("Memory allocation error. Terminate.\n");

exit(2);

}

strcpy(temp, buffer);

if (Core\_Element == NULL)

Core\_Element = addElement(temp);

else

findElement(Core\_Element, temp);

}

output(Core\_Element, file);

fclose(file);

return 0;

}

**Результаты работы программы**

Ввод:

qqq wqwe qq qq qq

q

q

~

Вывод:

1: qqq

1: wqwe

3: qq

2: q

**Вывод**

Изучил работу с файлами. Изучил способ добавления элемента в конец. Изучил поиск в по списку