МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Вычислительной техники

ОТЧЕТ

по лабораторной работе № 8
по дисциплине «Программирование»
Тема: Линейные односвязные списки

Студент гр. 3312 _____ Мохно Даниил.

Преподаватель _____ Аббас Саддам

Санкт-Петербург
2024

Цель работы.

Целью работы является изучение линейных односвязных списков в языке Си и получение практических навыков в программировании на этом языке.

Задание (вариант 3)

С использованием структуры, созданной при выполнении лабораторной работы №7 (по выбранной предметной области), создать односвязный линейный список и выполнить задание в соответствии с вариантом.

Разработать подалгоритм и написать функцию, удаляющую в односвязном списке заданный по номеру элемент. Номер элемента задается с конца списка. При недостаточном количестве элементов в списке удалить элемент из начала списка (первый). При пустом списке вывести соответствующее сообщение.

Постановка задачи и описание решения

Для начала объявим структуру. Структура Smartphone содержит 9 полей. Два символьных поля: model – модель смартфона и brand – марка смартфона. Три целочисленных поля: ram – объём оперативной памяти, memory – объём постоянной памяти, number_of_cameras – количество камер в смартфоне. Три поля с плавающей точкой: screen_size – размер экрана, weight – вес, price – цена. И целочисленный массив camera_resolution, содержащий разрешения всех камер смартфона. Затем объявим структуры для головы и узлов односвязного списка. Структура головы односвязного списка Storage содержит целочисленное поле size хранящее кол-во элементов в списке и указатель на первый элемент списка first_pos. Структуры элементов списка (Position) содержат в себе целочисленное поле index, хранящее порядковый номер элемента в списке, указатель на структуру Smartphone и указатель на следующий такой же элемент – next. Для удобства выделим структуры в новые типы данных.

В главной функции объявляем переменную Market – указатель на Storage, она будет являться головой односвязного списка. Затем запрашиваем у пользователя имя файла, выделяем память в Market, и присваиваем значение

размера списка 0. Далее вызываем функцию заполнения списка из файла, куда передаём Market и имя файла. Теперь проверяем чтобы размер списка был больше 0 и вызываем функцию удаления элемента по выбору индекса. В конце освобождаем память из-под массива.

В функции заполнения списка мы открываем файл, и запускаем цикл, пока считываются строки из файла. В цикле создаём новый элемент списка, с помощью ф-ии создания элемента списка, в которую передаём строку и индекс позиции начиная с 1. Затем вызываем функцию добавления элемента, куда передаём указатель на голову списка и наш новый элемент. В случае если файл не открылся, выводим сообщение об ошибке.

В ф-и создания элемента мы выделяем память под новый элемент связного списка присваиваем переданный индекс полю index и вызываем ф-ю присвоения значений в которую передаём строку из файла и поле smartphone. В конце возвращаем созданный элемент связного списка.

В функции присвоения значений мы выделяем место под строки модели и бренда и строку содержащую подстроку с камерами. Присваиваем значения из строки в поля структуры и строку с камерами путём форматированного сканирования строки функцией sscanf. Затем вызываем функцию разделения строки с разрешениями камер и заполнения массива камер. Освобождаем место из-под строки с разрешениями камер.

В функции разделения строки с камерами мы присваиваем длину строки в переменную len и перебираем строку считая разделённые числа. Далее выделяем память для массива разрешений камер в структуре и для строки временного хранения отделённых значений. Если кол-во чисел в строке не равно 0, то циклом проходимся по строке. Цикл начинается с инициализации трех переменных: i, j и k, которые используются как счетчики и индексы. Затем цикл выполняется до тех пор, пока i не превысит длину строки len. Внутри цикла проверяется каждый символ строки str по индексу i. Если символ не является знаком разделителем и не является символом конца строки ('\0'), то он копируется в временную строку по индексу j. Если символ равен символу

разделителю или является символом конца строки, то производится преобразование временной строки в целое число с помощью atoi, которое затем присваивается элементу массива разрешений камер по индексу k. После этого временная строка обнуляется, а счетчики k и j обновляются для следующей итерации. В конце присваиваем полю количества камер кол-во разделённых чисел и освобождаем временную строку.

В функции добавления элемента, мы вставляем элемент в начало списка. Для этого мы в поле next нового элемента записываем указатель first_pos из головы списка. Затем записываем в first_pos новый элемент и инкрементируем значение размера списка.

В функции удаления элемента по выбору индекса мы запускаем цикл с постусловием, работающим, пока пользователь не введёт 0. В цикле мы выводим таблицу из значений полей структуры Smartphone каждого узла, передав в функцию указатель на первый элемент списка, если размер списка больше 0. Затем запрашиваем у пользователя индекс элемента, который он хочет удалить из списка. Если значение, введённое пользователем больше 0, то мы вызываем функцию удаления выбранного элемента.

Функция вывода таблицы реализована рекурсивно, так как индексация списка идёт по убыванию, а нам нужно вывести его последовательно начиная с элемента с индексом 1, то есть с конца. В функции мы проверяем чтобы элемент не был равен NULL. Если он существует, мы рекурсивно вызываем функцию вывода таблицы, которой передаём поле next текущего элемента. Затем мы вызываем функцию вывода полей print, в которую передаём поле smartphone текущего узла и его индекс.

В функции вывода print мы склеиваем значения камер из поля с массивом камер в одну строку и выводим её и другие поля в таблицу. Для формирования строки с разрешениями камер cameras_res из поля с массивом со значениями мы объявляем заполняемою строку с разрешениями камер и промежуточную строку для перевода числа в строку tmp_cam. Выделяем память для строки cameras_res. Затем циклом проходимся по массиву разрешений камер. В нём

преобразуем число из массива в строку, записанную в tmp_cam с помощью функции sprintf. Затем вложенный цикл проходится по символам этой строки, копируя их в строку cameras_res по индексу j. Если это не последняя камера, после копирования числа добавляется символ '+' в строку cameras_res, и счетчик j обновляется. После вывода освобождаем строку cameras_res.

В функции удаления выбранного пользователем элемента, мы создаём переменную указатель текущего элемента в которую присваиваем, адрес первого элемента списка из поля first_pos головы списка. Затем мы проверяем равен ли размер списка 0, если да, то мы выводим сообщение о пустом списке. В ином случае, мы проверяем больше или равен ли переданный индекс размеру списка. Если это так, то нам нужно удалить элемент с последним индексом, то есть первый элемент списка. Для этого просто в указатель на первый элемент first_pos в голове передаём адрес из поля next текущего, то есть первого элемента и освобождаем память этого элемента. В случае если переданный индекс меньше размера массива, мы запускаем цикл, который будет идти, пока текущий элемент не равен NULL. Если индекс следующего, относительно текущего, элемента, равен переданному индексу, то в дополнительную переменную мы помещаем следующий элемент, а в поле next текущего элемента помещаем адрес из поля next следующего элемента. Затем очищаем память из-под временного элемента, уменьшаем индекс текущего элемента, и присваиваем переменной текущего элемента NULL, завершая цикл. Если же индексы оказались не равны, мы декрементируем индекс текущего элемента, и в текущий элемент присваиваем адрес следующего элемента.

Описание переменных

Функция – int main():

№	Имя переменной	Тип	Назначение
1	Market	struct Storage	Указатель на голову связного списка
2	len	int	Длинна названия файла
3	filename	char	Название файла

Функция создания головы списка – Storage *create_storage():

№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка

Функция заполнения списка из файла – void fill_storage(Storage *storage, char *filename):

№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка
2	filename	char	Название файла
3	source	FILE	Указатель на файл
4	new_pos	struct Position	Указатель на новый элемент списка
5	tmp_str	char	Строка из файла
6	i	int	Итератор

Функция создания узла списка – Position create_position(Storage *storage, char *filename):

№	Имя переменной	Тип	Назначение
1	position	struct Position	Указатель на элемент списка
2	string	char	Строка из файла
3	index	int	Порядковый номер в списке

Функция присвоения значений полям структуры – void set_values(Smartphone *smartphone, char *str)

№	Имя переменной	Тип	Назначение
1	smartphone	struct Smartphone	Указатель на структуру
2	str	char	Строка из файла
3	cameras	char	Подстрока с разрешениями камер

Функция разделения разрешений камер – void split_camera_resolution(Smartphone *smartphone, char *str)

№	Имя переменной	Тип	Назначение
1	smartphone	struct Smartphone	указатель на структуру
2	str	char	Строка с разрешениями камер
3	i	int	итератор
4	j	int	итератор
5	k	int	итератор
6	tmp_str	char	Временная строка для хранения подстрок с числами из строки с расширениями
7	len	int	Длинна строки с расширениями камер
8	number	int	Счётчик чисел в строке разделённых знаком

Функция добавления элемента в список – void add(storage, new_position)

№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка
2	new_position	struct Position	Указатель на новый элемент списка

Функция выбора индекса удаляемого элемента– void choose_to_delete(storage)

№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка
2	val	int	Индекс элемента выбранного пользователем

Функция вывода таблицы – void print_table(position)

№	Имя переменной	Тип	Назначение
1	posititon	struct Position	Указатель на голову связного списка

Функция вывода полей структуры – void print(Smartphone *smartphone, int index)

№	Имя переменной	Тип	Назначение
1	smartphone	struct Smartphone	Указатель на структуру
2	i	int	Итератор
3	J	int	Итератор
4	k	int	Итератор
5	tmp_cam	char	Строка для преобразования чисел в строку
6	camera_res	char	Строка для склеивания всех разрешений из массива разрешений камер
7	index	int	Порядковый номер элемента в списке

Функция удаления выбранного элемента– void delete_selected (storage)

№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка
2	index	int	Порядковый номер элемента в списке
3	cur	struct Position	Текущий элемент списка, который проверяется на соответствие условиям удаления.
4	tmp	struct Position	Используется для временного хранения указателя на элемент, который будет удален из списка

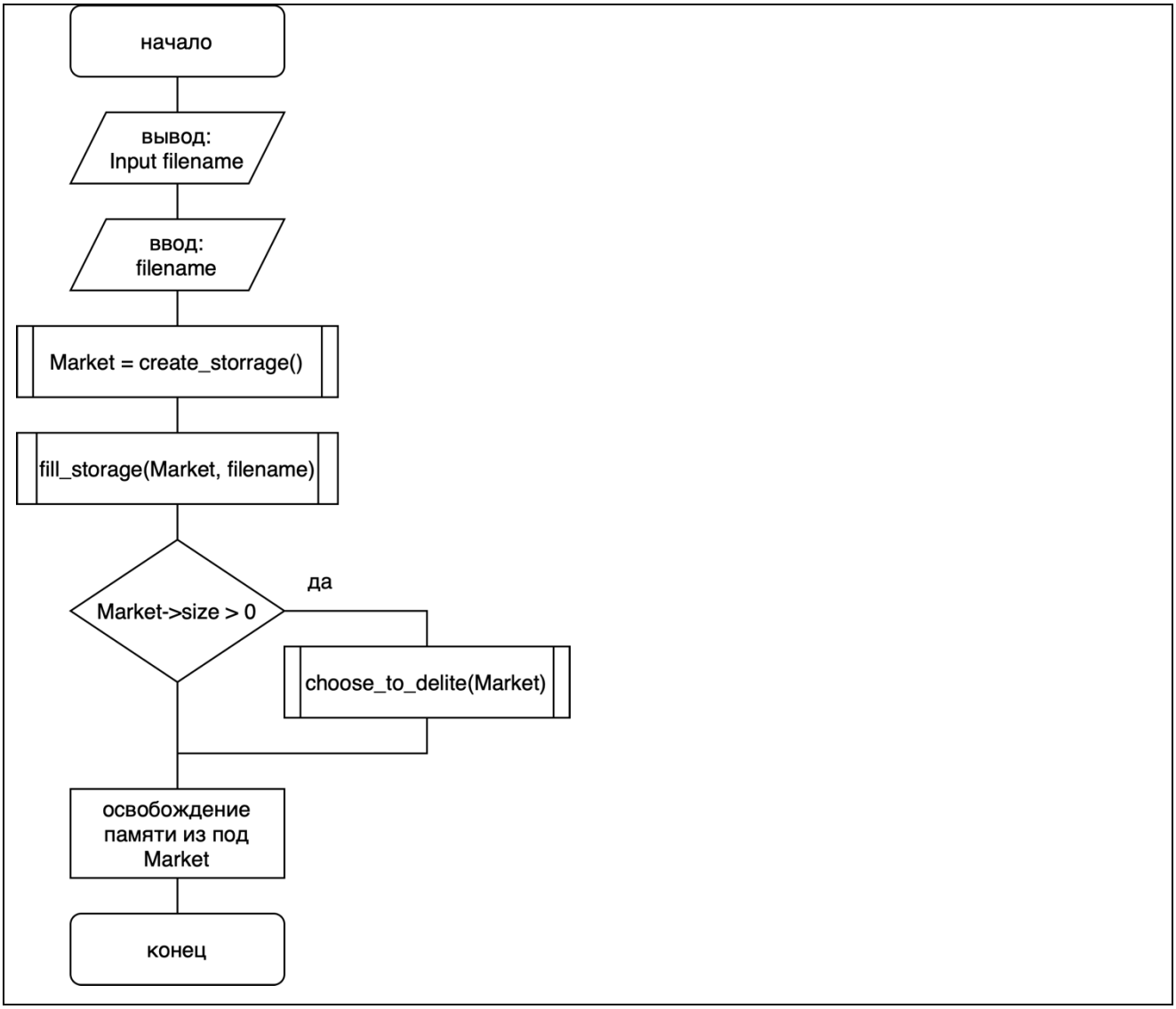
Функция освобождения памяти узла списка – void delete_position(storage)

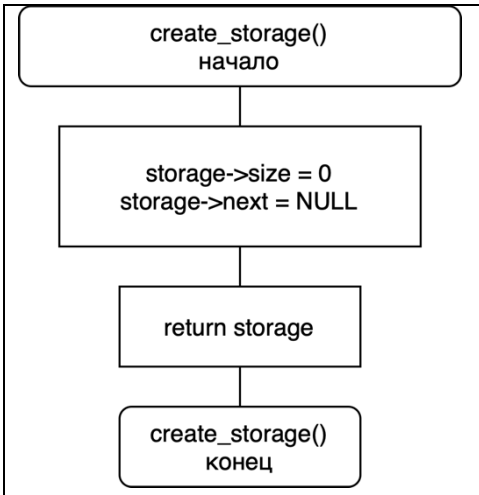
№	Имя переменной	Тип	Назначение
1	posititon	struct Position	Указатель на голову связного списка

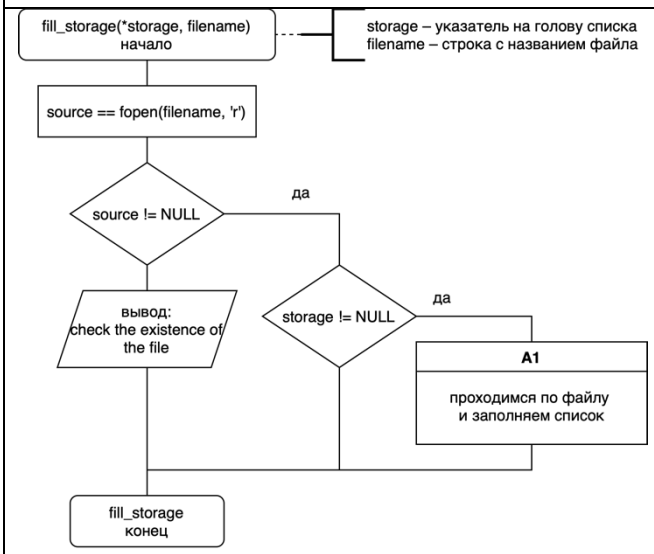
Функция удаления списка– void delete_storage(storage)

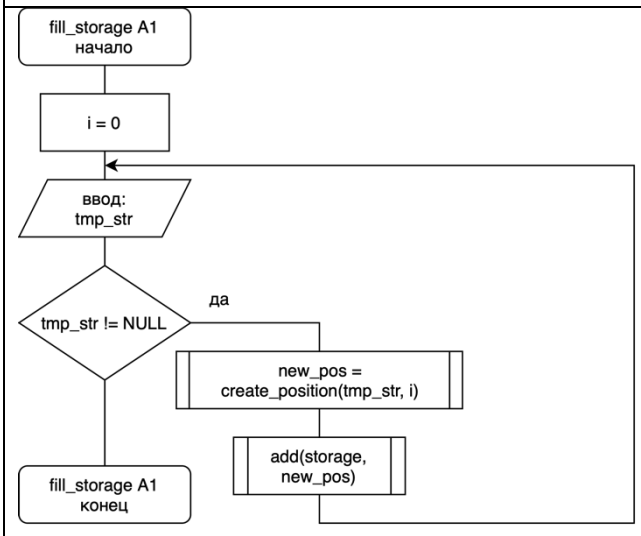
№	Имя переменной	Тип	Назначение
1	storage	struct Storage	Указатель на голову связного списка
2	index	int	Порядковый номер элемента в списке
3	cur	struct Position	Указатель на текущий элемент списка, который нужно удалить.
4	next	struct Position	Указатель на следующий элемент, который нужно, не потерять при удалении текущего элемента.

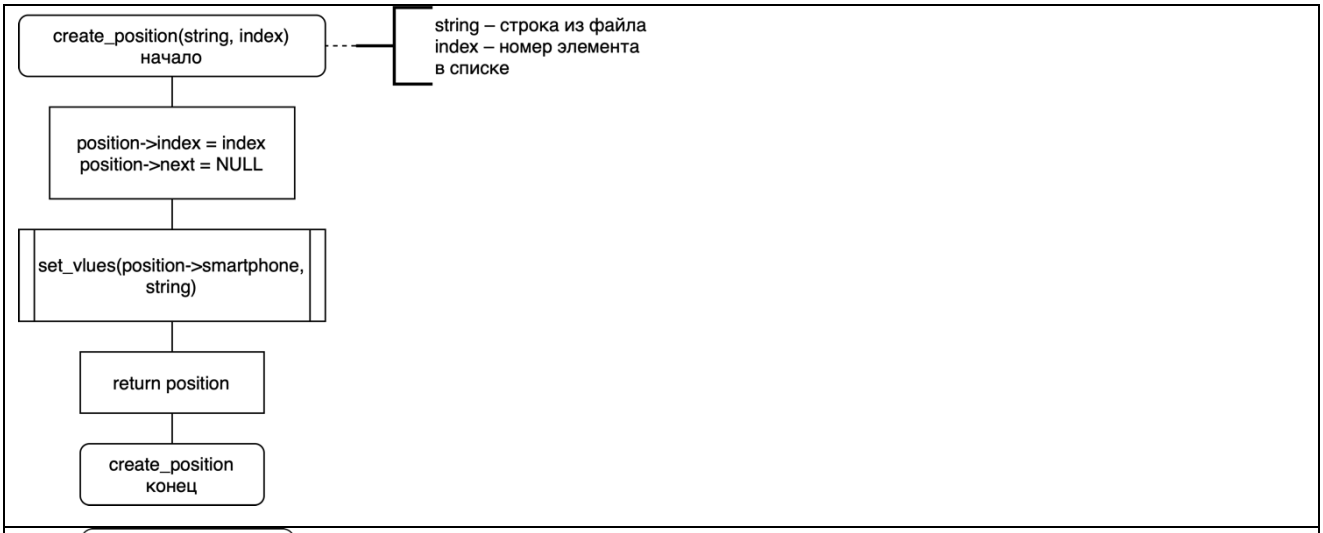
Схема алгоритма

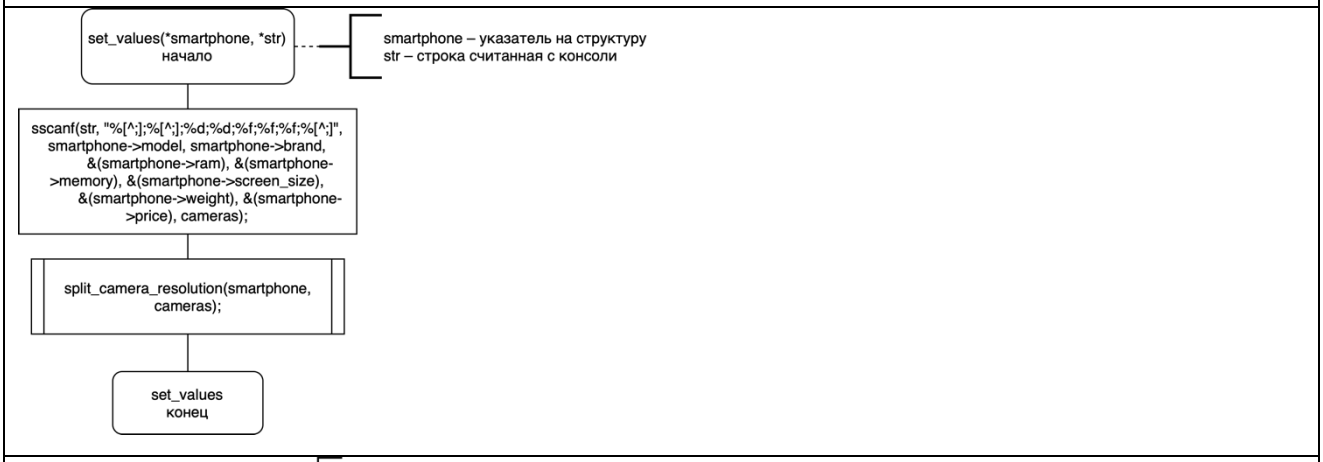


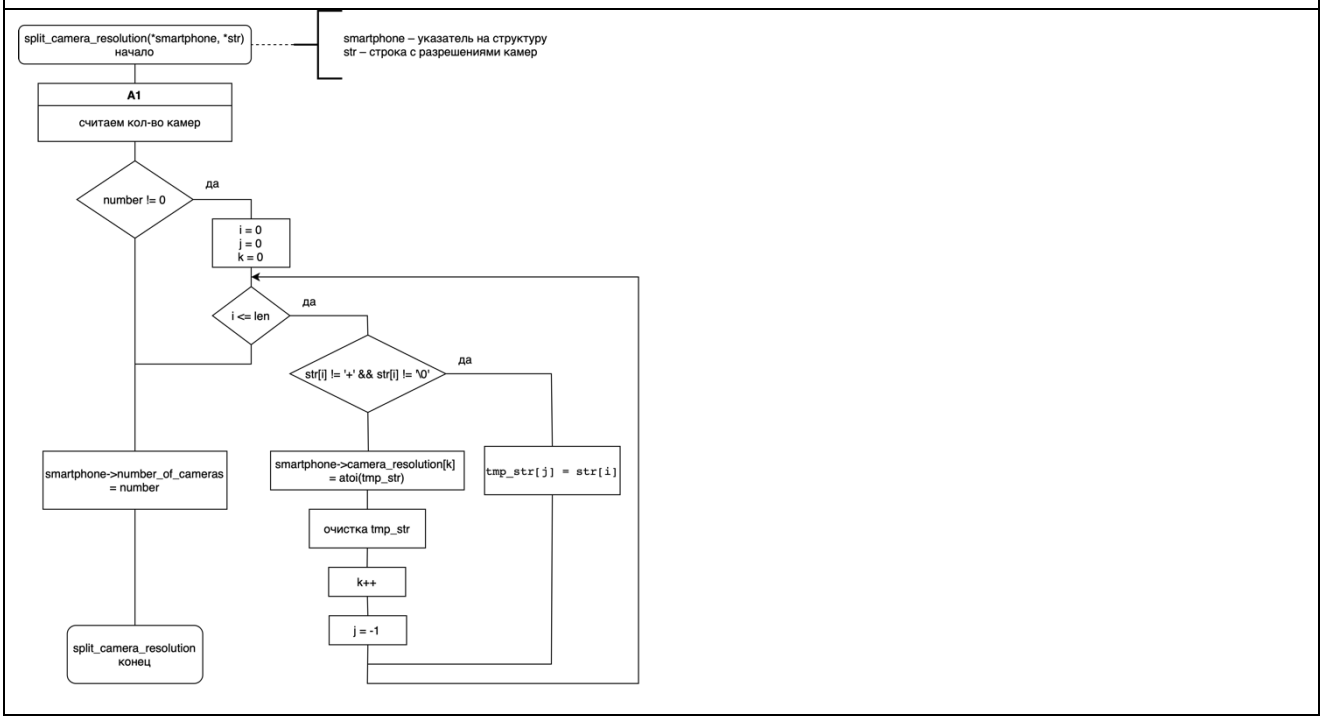

create_storage()
начало
storage->size = 0
storage->next = NULL
return storage
create_storage()
конец

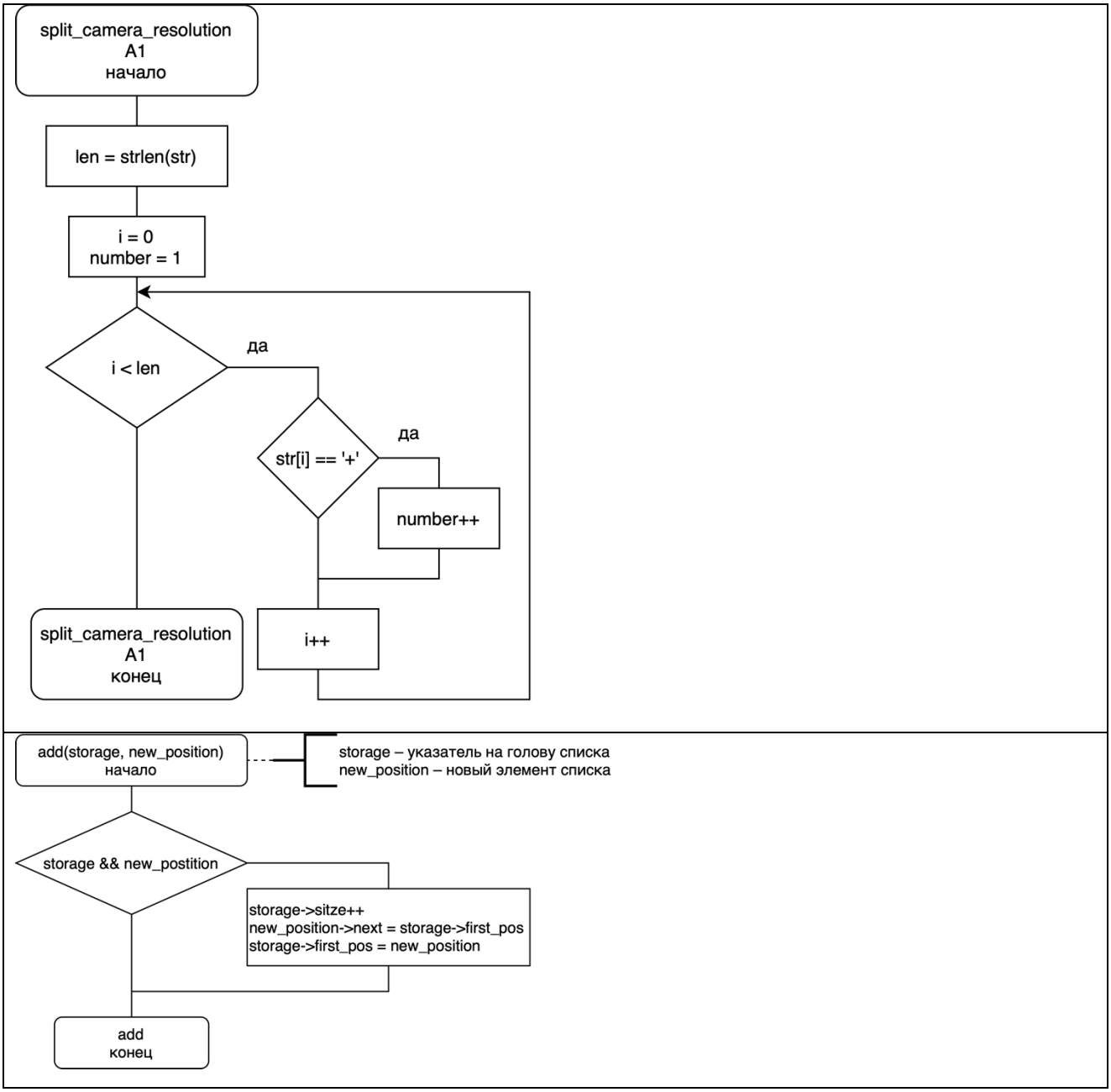

fill_storage(*storage, filename)
начало
storage – указатель на голову списка
filename – строка с названием файла
source == fopen(filename, 'r')
source != NULL
да
вывод:
check the existence of
the file
storage != NULL
да
A1
проходимся по файлу
и заполняем список
fill_storage
конец

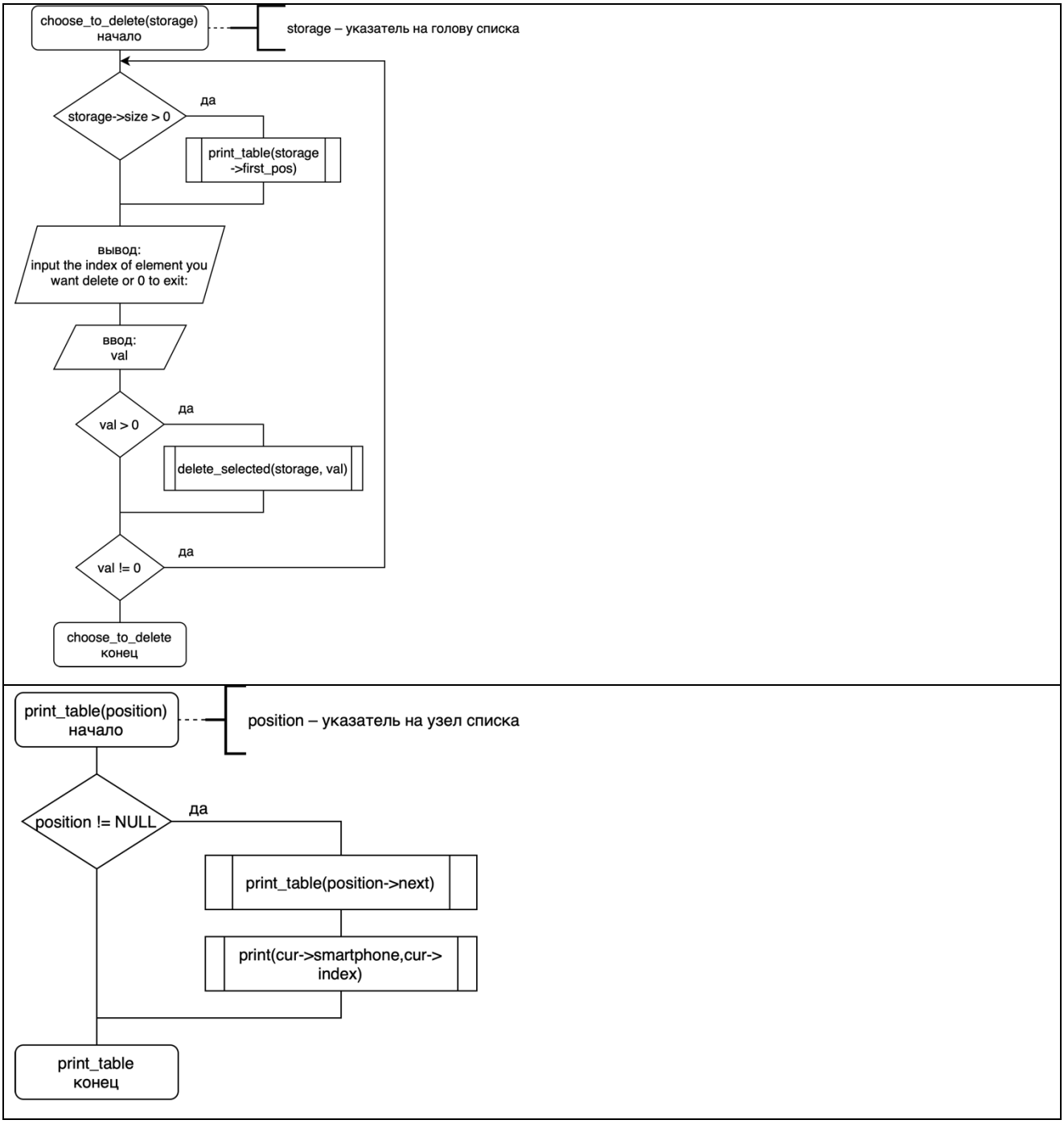

fill_storage A1
начало
i = 0
ввод:
tmp_str
tmp_str != NULL
да
new_pos =
create_position(tmp_str, i)
add(storage,
new_pos)
fill_storage A1
конец

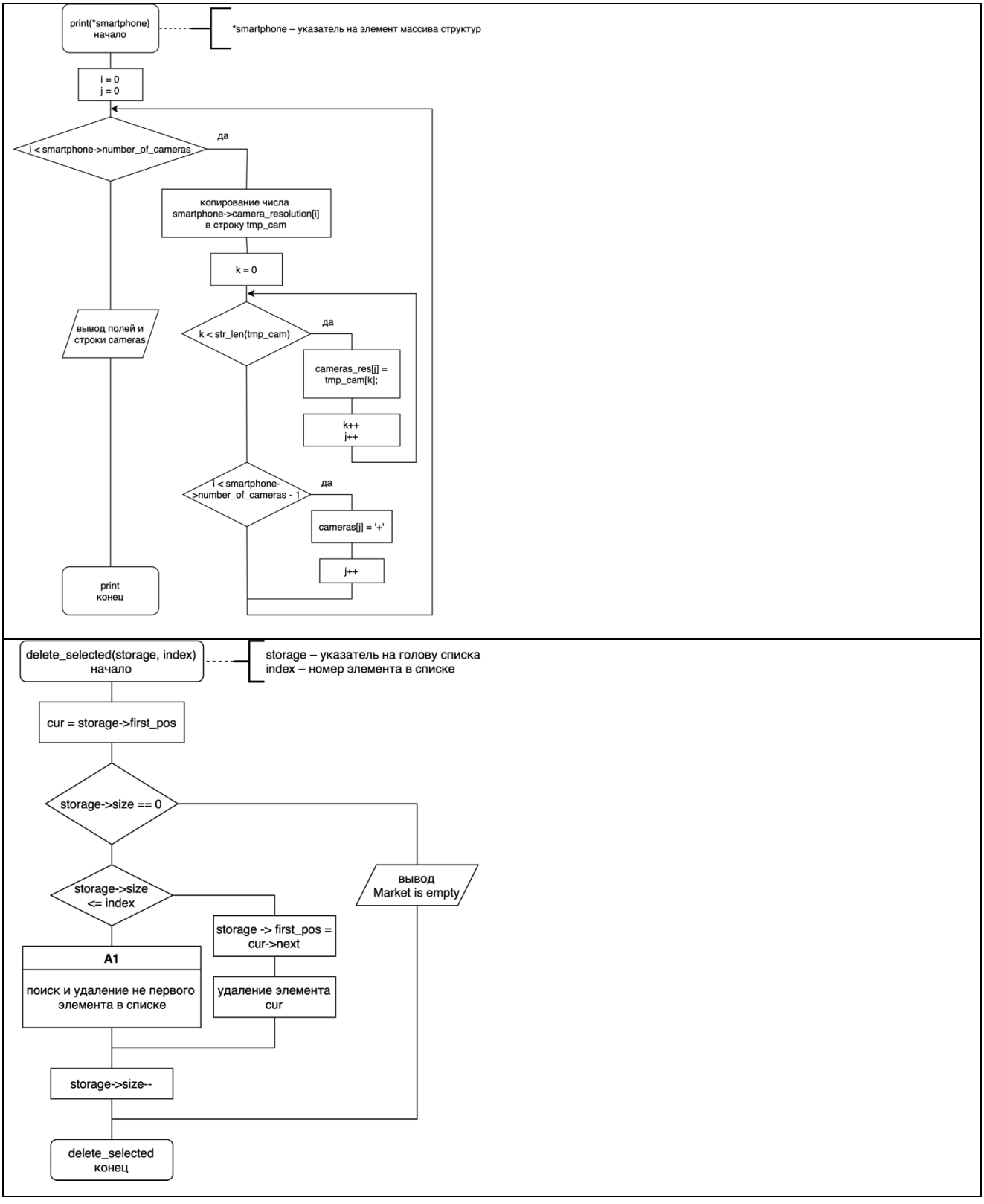

create_position(string, index)
начало
string – строка из файла
index – номер элемента
в списке
position->index = index
position->next = NULL
set_vlues(position->smartphone,
string)
return position
create_position
конец

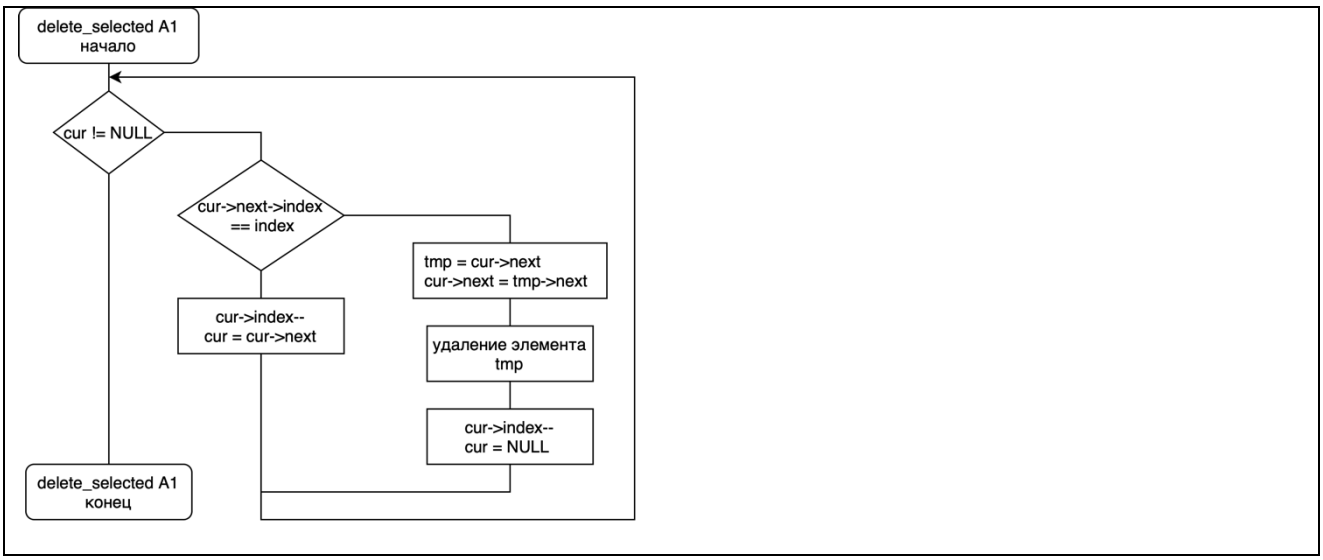

set_values(*smartphone, *str)
начало
smartphone – указатель на структуру
str – строка считанная с консоли
sscanf(str, "%[^;];%[^;];%d;%d;%f;%f;%f;%[^;]",
smartphone->model, smartphone->brand,
&(smartphone->ram), &(smartphone-
>memory), &(smartphone->screen_size),
&(smartphone->weight), &(smartphone-
>price), cameras);
split_camera_resolution(smartphone,
cameras);
set_values
конец


split_camera_resolution(*smartphone, *str)
начало
smartphone – указатель на структуру
str – строка с разрешениями камер
A1
считаем кол-во камер
number != 0
да
i = 0
j = 0
k = 0
i <= len
да
str[i] != '+' && str[i] != '\0'
да
smartphone->number_of_cameras
= number
smartphone->camera_resolution[k]
= atoi(tmp_str)
tmp_str[j] = str[i]
очистка tmp_str
k++
j = -1
split_camera_resolution
конец


split_camera_resolution
A1
начало
len = strlen(str)
i = 0
number = 1
i < len
да
str[i] == '+'
да
number++
i++
split_camera_resolution
A1
конец
add(storage, new_position)
начало
storage – указатель на голову списка
new_position – новый элемент списка
storage && new_postition
storage->sitze++
new_position->next = storage->first_pos
storage->first_pos = new_position
add
конец


choose_to_delete(storage)
начало
storage – указатель на голову списка
storage->size > 0
да
print_table(storage
->first_pos)
вывод:
input the index of element you
want delete or 0 to exit:
ввод:
val
val > 0
да
delete_selected(storage, val)
val != 0
да
choose_to_delete
конец
print_table(position)
начало
position – указатель на узел списка
position != NULL
да
print_table(position->next)
print(cur->smartphone,cur->
index)
print_table
конец


print(*smartphone)
начало
*smartphone – указатель на элемент массива структур
i = 0
j = 0
i < smartphone->number_of_cameras
да
копирование числа
smartphone->camera_resolution[i]
в строку tmp_cam
k = 0
вывод полей и
строки cameras
k < str_len(tmp_cam)
да
cameras_res[j] =
tmp_cam[k];
k++
j++
i < smartphone-
>number_of_cameras - 1
да
cameras[j] = '+'
j++
print
конец
delete_selected(storage, index)
начало
storage – указатель на голову списка
index – номер элемента в списке
cur = storage->first_pos
storage->size == 0
вывод
Market is empty
storage->size
<= index
storage -> first_pos =
cur->next
A1
поиск и удаление не первого
элемента в списке
удаление элемента
cur
storage->size--
delete_selected
конец



Контрольные примеры

Пример 1:

Содержимое файла Data.csv
iPhone11;Apple;4;128;6.06;194;599;12
iPhone12;Apple;4;256;6.1;215;799;12
Galaxy S20;Samsung;6;128;6.2;163;699;64
OnePlus 8;OnePlus;8;256;6.55;180;699;48
Pixel 5;Google;8;128;6.0;151;699;12
Xperia 1 II;Sony;8;256;6.5;181;1099;12+12+12+12+12

Входные данные
Data.csv 1 5 77 2 0

Выходные данные

```
| №  | Model          | Brand       | RAM   | Storage | Screen | Weight  | Price    | Camera Resolution |
|  1 | iPhone11       | Apple       | 4  GB | 128  GB | 6.06 " | 194.00g | $599.00  |            12mp |
|  2 | iPhone12       | Apple       | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |            12mp |
|  3 | Galaxy S20     | Samsung     | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |            64mp |
|  4 | OnePlus 8      | OnePlus     | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |            48mp |
|  5 | Pixel 5        | Google      | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |            12mp |
|  6 | Xperia 1 II    | Sony        | 8  GB | 256  GB | 6.50 " | 181.00g | $1099.00 | 12+12+12+12+12mp |

| №  | Model          | Brand       | RAM   | Storage | Screen | Weight  | Price    | Camera Resolution |
|  1 | iPhone12       | Apple       | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |            12mp |
|  2 | Galaxy S20     | Samsung     | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |            64mp |
|  3 | OnePlus 8      | OnePlus     | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |            48mp |
|  4 | Pixel 5        | Google      | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |            12mp |
|  5 | Xperia 1 II    | Sony        | 8  GB | 256  GB | 6.50 " | 181.00g | $1099.00 | 12+12+12+12+12mp |

| №  | Model          | Brand       | RAM   | Storage | Screen | Weight  | Price    | Camera Resolution |
|  1 | iPhone12       | Apple       | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |            12mp |
|  2 | Galaxy S20     | Samsung     | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |            64mp |
|  3 | OnePlus 8      | OnePlus     | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |            48mp |
|  4 | Pixel 5        | Google      | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |            12mp |

| №  | Model          | Brand       | RAM   | Storage | Screen | Weight  | Price    | Camera Resolution |
|  1 | iPhone12       | Apple       | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |            12mp |
|  2 | Galaxy S20     | Samsung     | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |            64mp |
|  3 | OnePlus 8      | OnePlus     | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |            48mp |

| №  | Model          | Brand       | RAM   | Storage | Screen | Weight  | Price    | Camera Resolution |
|  1 | iPhone12       | Apple       | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |            12mp |
|  2 | OnePlus 8      | OnePlus     | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |            48mp |
```

Пример 2:

Содержимое файла Data.csv

```
Galaxy S21;Samsung;8;128;6.2;171;799;64+12+12
iPhone SE 2020;Apple;3;64;4.7;148;399;12
OnePlus 9;OnePlus;12;256;6.55;183;899;48+50+2
```

Входные данные
Data2.csv 2 33 1 8 0

Выходные данные
| № | Model | Brand | RAM | Storage | Screen | Weight | Price | Camera Resolution | | 1 | Galaxy S21 | Samsung | 8 GB | 128 GB | 6.20 " | 171.00g | $799.00 | 64+12+12mp | | 2 | iPhone SE 2020 | Apple | 3 GB | 64 GB | 4.70 " | 148.00g | $399.00 | 12mp | | 3 | OnePlus 9 | OnePlus | 12 GB | 256 GB | 6.55 " | 183.00g | $899.00 | 48+50+2mp | | № | Model | Brand | RAM | Storage | Screen | Weight | Price | Camera Resolution | | 1 | Galaxy S21 | Samsung | 8 GB | 128 GB | 6.20 " | 171.00g | $799.00 | 64+12+12mp | | 2 | OnePlus 9 | OnePlus | 12 GB | 256 GB | 6.55 " | 183.00g | $899.00 | 48+50+2mp | | № | Model | Brand | RAM | Storage | Screen | Weight | Price | Camera Resolution | | 1 | Galaxy S21 | Samsung | 8 GB | 128 GB | 6.20 " | 171.00g | $799.00 | 64+12+12mp |

Текст программы

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#ifdef WIN32
#define CLS system("cls")
#else
#define CLS system("clear")
#endif

enum
{
    MAX_STR_IN_FILE_LEN = 200,
    MAX_MODEL_NAME_LEN = 30,
    MAX_FILENAME_LEN = 20
};

typedef struct Smartphone
{
    char *model;            /* Model */
    char *brand;            /* Brand */
    int ram;                /* RAM, GB */
    int memory;             /* Storage, GB */
    float screen_size;      /* Screen size, inches */
    float weight;           /* Weight, grams */
    float price;            /* Price, dollars */
    int *camera_resolution; /* Camera resolution (per camera) */

} Smartphone;
```

```c
typedef struct Position
{
    Smartphone *smartphone;
    int index;
    struct Position *next;
} Position;

typedef struct Storage
{
    int size;
    Position *first_pos;
} Storage;

Storage *create_storage();

void fill_storage(Storage *storage, char *filename);

Position *create_position(char *string, int index);

void set_values(Smartphone *smartphone, char *str);

void split_camera_resolution(Smartphone *smartphone, char *str);

void add(Storage *storage, Position *new_position);

void print_table(Position* position);

void print_header();

void print(Smartphone *smartphone, int index);

void choose_to_delete(Storage *storage);

void delete_selected(Storage *storage, int index);

void delete_storage(Storage *storage);

void delete_position(Position *position);

int main()
{
    Storage *Market;
    int len;
    char filename[MAX_FILENAME_LEN];

    /* ------------get filename--------------- */
    printf("Input filename: ");
    fgets(filename, MAX_FILENAME_LEN, stdin);
    len = (int)strlen(filename);
    filename[len - 1] = '\0';
    /* ------------------------------------- */

    Market = create_storage();

    fill_storage(Market, filename);
    if (Market->size > 0)
        choose_to_delete(Market);


    delete_storage(Market);
    return 0;
}

Storage *create_storage()
{
    Storage *storage = NULL;
    storage = malloc(sizeof(Storage));
    if (storage)
    {
```

```
        storage->size = 0;
        storage->first_pos = NULL;
    }
    return storage;
}

void fill_storage(Storage *storage, char *filename)
{
    FILE *source;
    Position *new_pos;
    char tmp_str[MAX_STR_IN_FILE_LEN];
    int i;
    source = fopen(filename, "r");
    if (source != NULL)
    {
        if (storage != NULL)
        {
            // проходимся по файлу
            for (i = 1; fgets(tmp_str, MAX_STR_IN_FILE_LEN, source); i++)
            {
                new_pos = create_position(tmp_str, i);
                add(storage, new_pos);
            }
            fclose(source);
        }
    } else
        printf("Check the existence of the file\n");
}

Position *create_position(char *string, int index)
{
    Position *position = NULL;
    position = (Position *) malloc(sizeof(Position));
    if (position)
    {
        position->smartphone = malloc(sizeof(Smartphone));
        position->index = index;
        position->next = NULL;
        set_values(position->smartphone, string);
    }
    return position;
}

void set_values(Smartphone *smartphone, char *str)
{
    char *cameras;
    cameras = malloc(MAX_MODEL_NAME_LEN * sizeof(char));
    smartphone->model = malloc(sizeof(char) * MAX_MODEL_NAME_LEN);
    smartphone->brand = malloc(sizeof(char) * MAX_MODEL_NAME_LEN);
    sscanf(str, "%[^;];%[^;];%d;%d;%f;%f;%f;%[^;]", smartphone->model, smartphone->brand,
           &(smartphone->ram), &(smartphone->memory), &(smartphone->screen_size),
           &(smartphone->weight), &(smartphone->price), cameras);
    split_camera_resolution(smartphone, cameras);
    free(cameras);
}

void split_camera_resolution(Smartphone *smartphone, char *str)
{
    int i, j, k;
    char *tmp_str;
    int len;
    int number;
    len = (int)strlen(str);
    for (i = 0, number = 1; i < len; i++) if (str[i] == '+') number++;
    tmp_str = malloc(4 * sizeof(char));
    smartphone->camera_resolution = malloc(sizeof(int) * number);
    if (number != 0)
    {
        for (i = 0, j = 0, k = 0; i <= len; i++, j++)
        {
```

```
            if (str[i] != '+' && str[i] != '\0')
            {
                tmp_str[j] = str[i];
            } else
            {
                smartphone->camera_resolution[k] = atoi(tmp_str);
                strcpy(tmp_str, "    ");
                k++;
                j = -1;
            }
        }
    }
    smartphone->number_of_cameras = number;
    free(tmp_str);
}

void add(Storage *storage, Position *new_position)
{
    if (storage && new_position)
    {
        storage->size++;
        new_position->next = storage->first_pos;
        storage->first_pos = new_position;
    }
}

void choose_to_delete(Storage *storage)
{
    int val;
    do
    {
        if (storage->size > 0)
        {
            print_header();
            print_table(storage->first_pos);
        }
        printf("input the index of element you want delete or 0 to exit: ");
        scanf("%d", &val);
        if (val > 0)
            delete_selected(storage, val);

    } while (val != 0);
}

void print_header()
{
    printf("| %-5s | %-20s | %-15s | %-5s | %-5s | %-6s | %-7s | %-8s | %-17s |\n",
           "№", "Model", "Brand", "RAM", "Storage", "Screen", "Weight", "Price", "Camera Resolu-
tion");
}

void print_table(Position* position)
{
    if(position != NULL)
    {
        print_table(position->next);
        print(position->smartphone, position->index);
    }
}

void print(Smartphone *smartphone, int index)
{
    int i, j, k;
    char *cameras_res;
    char tmp_cam[4];

    cameras_res = malloc(smartphone->number_of_cameras * 3 * sizeof(char));
    for (i = 0, j = 0; i < smartphone->number_of_cameras; i++)
    {
        snprintf(tmp_cam, sizeof(tmp_cam), "%d", smartphone->camera_resolution[i]);
```

```
        for (k = 0; k < strlen(tmp_cam); k++, j++)
            cameras_res[j] = tmp_cam[k];
        if (i < smartphone->number_of_cameras - 1)
        {
            cameras_res[j] = '+';
            j++;
        }
    }
    printf("| %3i | %-20s | %-15s | %-3dGB | %-5dGB | %-5.2f\" | %-6.2fg | $%-7.2f | %15smp |\n",
           index, smartphone->model, smartphone->brand, smartphone->ram, smartphone->memory,
           smartphone->screen_size, smartphone->weight, smartphone->price, cameras_res);
    free(cameras_res);
}

void delete_selected(Storage *storage, int index)
{
    Position *cur, *tmp;
    cur = storage->first_pos;

    if (storage->size == 0) printf("Market is empty\n");
    else
    {
        if (storage->size <= index)
        {
            storage->first_pos = cur->next;
            delete_position(cur);
        } else
        {
            while (cur != NULL)
            {
                if (cur->next->index == index)
                {
                    tmp = cur->next;
                    cur->next = tmp->next;
                    delete_position(tmp);
                    cur->index--;
                    cur = NULL;
                } else
                {
                    cur->index--;
                    cur = cur->next;
                }
            }
        }
        storage->size--;
    }
}

void delete_position(Position *position)
{
    free(position->smartphone->model);
    free(position->smartphone->brand);
    free(position->smartphone->camera_resolution);
    free(position->smartphone);
    position->next = NULL;
    free(position);
}

void delete_storage(Storage *storage)
{
    Position *cur, *next;
    cur = storage->first_pos;
    while (cur != NULL)
    {
        next = cur->next;
        delete_position(cur);
        cur = next;
    }
    free(storage);
}
```

Примеры выполнения программы

Пример 1

```
"/Users/daniilmohno/Library/Mobile Documents/com~apple~CloudDocs/Student-staff/Прога/лабы по прое/2сем/Laboratory_work_8/
Input filename: Data.csv
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | iPhone11            | Apple           | 4  GB | 128  GB | 6.06 " | 194.00g | $599.00  |              12mp |
|   2 | iPhone12            | Apple           | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |              12mp |
|   3 | Galaxy S20          | Samsung         | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |              64mp |
|   4 | OnePlus 8           | OnePlus         | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |              48mp |
|   5 | Pixel 5             | Google          | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |              12mp |
|   6 | Xperia 1 II         | Sony            | 8  GB | 256  GB | 6.50 " | 181.00g | $1099.00 |  12+12+12+12+12mp |
input the index of element you want delete or 0 to exit: 1
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | iPhone12            | Apple           | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |              12mp |
|   2 | Galaxy S20          | Samsung         | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |              64mp |
|   3 | OnePlus 8           | OnePlus         | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |              48mp |
|   4 | Pixel 5             | Google          | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |              12mp |
|   5 | Xperia 1 II         | Sony            | 8  GB | 256  GB | 6.50 " | 181.00g | $1099.00 |  12+12+12+12+12mp |
input the index of element you want delete or 0 to exit: 5
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | iPhone12            | Apple           | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |              12mp |
|   2 | Galaxy S20          | Samsung         | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |              64mp |
|   3 | OnePlus 8           | OnePlus         | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |              48mp |
|   4 | Pixel 5             | Google          | 8  GB | 128  GB | 6.00 " | 151.00g | $699.00  |              12mp |
input the index of element you want delete or 0 to exit: 77
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | iPhone12            | Apple           | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |              12mp |
|   2 | Galaxy S20          | Samsung         | 6  GB | 128  GB | 6.20 " | 163.00g | $699.00  |              64mp |
|   3 | OnePlus 8           | OnePlus         | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |              48mp |
input the index of element you want delete or 0 to exit: 2
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | iPhone12            | Apple           | 4  GB | 256  GB | 6.10 " | 215.00g | $799.00  |              12mp |
|   2 | OnePlus 8           | OnePlus         | 8  GB | 256  GB | 6.55 " | 180.00g | $699.00  |              48mp |
input the index of element you want delete or 0 to exit: 0

Process finished with exit code 0
```

Пример 2

```
"/Users/daniilmohno/Library/Mobile Documents/com~apple~CloudDocs/Student-staff/Прога/лабы по прое/2сем/Laboratory_work_8/
Input filename: Data2.csv
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | Galaxy S21          | Samsung         | 8  GB | 128  GB | 6.20 " | 171.00g | $799.00  |        64+12+12mp |
|   2 | iPhone SE 2020      | Apple           | 3  GB | 64   GB | 4.70 " | 148.00g | $399.00  |              12mp |
|   3 | OnePlus 9           | OnePlus         | 12 GB | 256  GB | 6.55 " | 183.00g | $899.00  |         48+50+2mp |
input the index of element you want delete or 0 to exit: 2
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | Galaxy S21          | Samsung         | 8  GB | 128  GB | 6.20 " | 171.00g | $799.00  |        64+12+12mp |
|   2 | OnePlus 9           | OnePlus         | 12 GB | 256  GB | 6.55 " | 183.00g | $899.00  |         48+50+2mp |
input the index of element you want delete or 0 to exit: 33
| №   | Model               | Brand           | RAM    | Storage | Screen | Weight  | Price    | Camera Resolution |
|   1 | Galaxy S21          | Samsung         | 8  GB | 128  GB | 6.20 " | 171.00g | $799.00  |        64+12+12mp |
input the index of element you want delete or 0 to exit: 1
input the index of element you want delete or 0 to exit: 8
Market is empty
input the index of element you want delete or 0 to exit: 0

Process finished with exit code 0
```

Выводы.

В результате выполнения работы были изучены односвязные линейные списки в языке Си и получены практические навыки в программировании на этом языке.