Балтийский государственный технический университет
«ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра О7 «Информационные системы и программная инженерия»

Практическая работа №1
по дисциплине «Структуры и организация данных»
на тему «Линейные структуры данных»
часть 1 «Связный линейный список»

вариант 5

Выполнил:
Студент Вяткин Н.А.
Группа О722Б

Преподаватель: Гладевич А.А

Санкт-Петербург
2023 г.

Уровень сложности – низкий уровень.

Данные хранятся в бинарном файле записей, а для обработки считываются в односвязный линейный список (если файл не существует, то создается пустой список). При выходе из программы обработанные данные сохраняются в том же файле. Имя файла с данными задается константой или вводится с клавиатуры. Для взаимодействия с пользователем должно использоваться меню. Обязательные операции для списка: добавление элемента в начало списка, удаление первого элемента списка, просмотр списка, поиск в соответствии с индивидуальным вариантом (две функции). Вывод списка на экран можно выполнять в любом (главное, читабельном) виде.

Поля данных: название животного, природная зона, затраты на корм одного животного в день, количество в зоопарке. Вывести наибольшее количество животных одного вида, находящихся в зоопарке, и определить, сколько денег тратится на содержание животных определенной природной 4 зоны в месяц.

Созданные типы данных:

Структура записи об одном животного

```
struct animal
{
    char NameAnimal[30]; // имя животного
    char NameArea[30];   // природная зона
    float MoneyDay;      // затраты на 1 день
    int count;           // количество животных
};
```

Тип данных для хранения записи об одном животном

```
typedef struct animal DataType;
```

Структура записи односвязного списка

```
typedef struct Node
{
    DataType individual;//тип каждой записи
    struct Node *next;// указатель на следующий элемент
};
```

Структура для односвязного списка

```
typedef struct Node *list;
```

Используемые функции:

1. Заполнение одной записи о животном.

```
DataType InputIndividual(void);
```

– Входные данные: отсутствуют.

– Выходные данные: запись о животном типа DataType.

2. Добавление записи в односвязный список.

```
list NewNode(list, DataType);
```

– Входные данные: указатель на начало односвязного списка типа list, запись о животном для добавления типа DataType.

– Выходные данные: указатель на начало односвязного списка.

3. Вывод на экран односвязного списка.

```
void ShowList(list);
```

– Входные данные: указатель на начало односвязного списка типа list.

– Выходные данные: отсутствуют.

4. Чтение записей из файла в односвязный список.

```
list ReadFile(char *filename);
```

– Входные данные: имя файла для прочтения.

– Выходные данные: указатель на начало односвязного списка.

5. Запись односвязного списка в бинарный файл.

```
int WriteFile(char *, list);
```

– Входные данные: указатель на начало односвязного списка типа list, имя файла для записи;

– Выходные данные: 0 – если записать не удалось, 1 – если запись прошла успешно.

6. Функция очищения памяти и удаления списка.

```
void DeleteList(list);
```

– Входные данные: указатель на начало односвязного списка типа list.

– Выходные данные: отсутствуют.

7. Поиск записей в односвязном списке (или поиск количества животных одного вида или затраты на корм для животных определенной природной зоны).

```
void Search(list);
```

– Входные данные: указатель на начало односвязного списка типа list.

– Выходные данные: отсутствуют.

8. Функция удаления записи с начала.

```
list DeleteNode(list);
```

– Входные данные: указатель на начало односвязного списка типа list.

– Выходные данные: указатель на начало односвязного списка.

Текст программы:

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

struct animal
{
    char NameAnimal[30]; // имя животного
    char NameArea[30];   // природная зона
    float MoneyDay;      // затраты на 1 день
    int count;           // количество животных
};

typedef struct animal DataType; // тип данных

// односвязный список
```

```
typedef struct Node
{
    DataType individual;
    struct Node *next;
};

typedef struct Node *list;

DataType InputIndividual(void); // заполнение одной записи о животном
list NewNode(list, DataType);    // добавление записи в односвязный
список
void ShowList(list);             // вывод на экран односвязного списка
list ReadFile(char *filename);  // чтение записей из файла в односвязный
список
int WriteFile(char *, list);    // запись односвязного списка в бинарный
файл
void DeleteList(list);           // функция очищения памяти и удаления
списка
void Search(list);              // поиск записей в односвязном списке(или
поиск количества животных одного вида или затраты на корм для животных
определенной природной зоны)
list DeleteNode(list);           // функция удаления записи с начала

int main(int argc, char *argv[])
{
    char filename[50];
    int menu;
    list animals = NULL;
    puts("Enter name file:");
    gets(filename);
    animals = ReadFile(filename);
    do
    {
        system("cls");
        puts("1. Add");
        puts("2. Show list");
        puts("3. Search");
        puts("4. Delete");
        puts("5. Exit");
        scanf("%d%*c", &menu);
        switch (menu)
        {
        case 1:
        {
            animals = NewNode(animals, InputIndividual());
            break;
        }
        case 2:
        {
            ShowList(animals);
            break;
        }
        case 3:
        {
            Search(animals);
            break;
        }
        case 4:
```

```
            {
                animals = DeleteNode(animals);
            }
            }
        } while (menu != 5);
        if (WriteFile(filename, animals))
            puts("File saved");
        else
            puts("File not saved");
        DeleteList(animals);
        return 0;
    }
    // заполнение одной записи о животном
    DataType InputIndividual(void)
    {
        DataType individual;
        puts("Name of animal");
        gets(individual.NameAnimal);
        puts("Name area");
        gets(individual.NameArea);
        puts("Money per day");
        scanf("%f", &individual.MoneyDay);
        puts("Count of this animal");
        scanf("%d", &individual.count);
        getchar();
        return individual;
    }
    // добавление записи в односвязный список
    list NewNode(list begin, DataType one)
    {
        list temp = (list)malloc(sizeof(struct Node));
         temp->individual = one; // заполняем поля данных записи списка,
копируем указатель значения переменной begin
        temp->next = begin;
        return temp; // указатель начала списка перемещаем на новый элемент
    }
    // вывод на экран односвязного списка
    void ShowList(list cur)
    {
        int k = 0;
        system("cls");
        if (cur == NULL) // проверка на пустоту
        {
            puts("List is empty");
            puts("To return to the menu, press any key");
            getchar();
            return;
        }
       puts("|  N |    Name of animal    |    Name of area    |        Money
per day        |   Count animal   |");
         puts("-----------------------------------------------------------
---------------------------------------");
        while (cur) // пока есть записи вывводим записи
        {
              printf("|%3d |%-22.22s|%-20.20s|%30.2f|%18.d|\n", ++k, cur-
>individual.NameAnimal, cur->individual.NameArea, cur->individual.MoneyDay,
cur->individual.count);
            cur = cur->next;
```

```
        }
         puts("-------------------------------------------------------------
----------------------------------------");
        puts("To return to the menu, press any key");
        getchar();
    }
    // запись односвязного списка в бинарный файл
    int WriteFile(char *filename, list begin)
    {
        FILE *f;
        if ((f = fopen(filename, "wb")) == NULL)
        {
            perror("Error create file");
            getchar();
            return 0;
        }
        while (begin) // пока есть записи записываем в файл
        {
            if (fwrite(&begin->individual, sizeof(DataType), 1, f))
                begin = begin->next;
            else
                return 0;
        }
        return 1;
    }
    // чтение записей из файла в односвязный список
    list ReadFile(char *filename)
    {
        FILE *f;
        DataType individual;
        list begin = NULL, cur;
        if ((f = fopen(filename, "rb")) == NULL)
        {
            perror("Erro open file");
            getchar();
            return begin;
        }
        if (fread(&individual, sizeof(individual), 1, f)) // если это первая
запись
            begin = NewNode(NULL, individual);
        else
            return NULL;
        cur = begin; // присваиваем указателю cur указатель на начало списка
        while (fread(&individual, sizeof(individual), 1, f))
        {
            cur->next = NewNode(NULL, individual); // создаем узел и цепляем
его к списку
             cur = cur->next;                          // переносим указатель
на следующий элемент
        }
        fclose(f);
        return begin;
    }
    // функция очищения памяти и удаления списка
    void DeleteList(list begin)
    {
        list temp = begin;
        while (temp) // пока есть элементы в списке
```

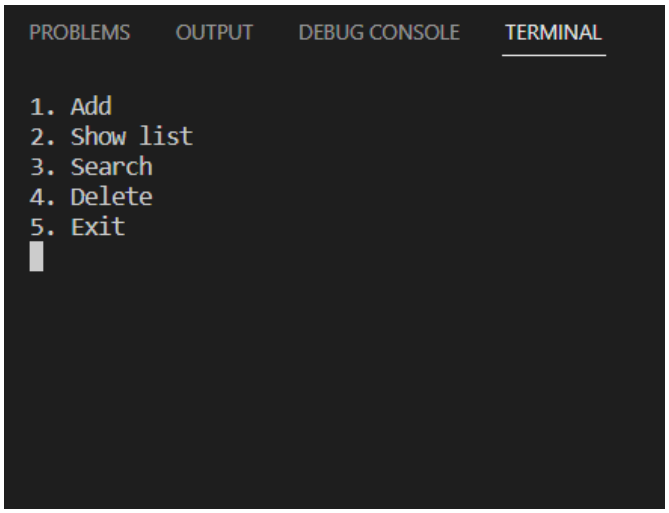
```
        {
            begin = temp->next;
            free(temp);
            temp = begin;
        }
    }
    // функция удаления записи с начала
    list DeleteNode(list begin)
    {
        system("cls");
        struct Node *temp;
        if (begin)
        {
             temp = begin;         // запоминаем адресс начала односвязного
списка
             begin = begin->next; // переносим указатель начала списка сна
следующий элемент
            free(temp);          // удаляем элемент
            puts("First element deleted");
            puts("To return to the menu, press any key");
            getchar();
        }
        return begin;
    }
    // поиск записей в односвязном списке(или поиск количества животных
одного вида или затраты на корм для животных определенной природной зоны)
    void Search(list cur)
    {
        char name[30], area[30];
        float money;
        int count = 0, k = 0, menu;
        DataType individual;
        system("cls");
        if (cur == NULL) // проверка списка на пустоту
        {
            puts("List is empty");
            puts("To return to the menu, press Enterr");
            while (getchar() != '\n')
                ;
            return;
        }
        do
        {
            puts("1. Find out the number of animals of the same species");
             puts("2. Find out the cost of keeping animals of a certain
natural area for a month");
            puts("3. Exit to the menu");
            scanf("%d%*c", &menu);
            switch (menu)
            {
            case 1:
            {
                  puts("Input name of animal to search"); // ввод имени
животного для поиска
                fgets(name, 30, stdin);
                {
                    char *ptmr = strchr(name, '\n');
                    if (ptmr)
```

```
                        *ptmr = '\0';
                    else
                        while (getchar() != '\n')
                            ;
                    name[0] = toupper(name[0]);
                }
                while (cur) // пока есть элементы
                {
                    if (strcmp(name, cur->individual.NameAnimal) == 0)
                        k = k + 1 * cur->individual.count;
                    cur = cur->next;
                }
                if (k > 0) // если найдены животные
                     printf("The number of animals of this species: %d\n",
k);
                else
                    puts("There is no such animal");
                puts("To return to the menu, press Enter");
                while (getchar() != '\n')
                    ;
                return;
            }
            case 2:
            {
                 puts("Enter the name of the natural area to calculate the
cost of animals per month");
                fgets(area, 30, stdin);
                {
                    char *ptmr = strchr(area, '\n');
                    if (ptmr)
                        *ptmr = '\0';
                    else
                        while (getchar() != '\n')
                            ;
                    area[0] = toupper(area[0]);
                }
                while (cur)
                {
                    if (strcmp(area, cur->individual.NameArea) == 0)
                            money += cur->individual.MoneyDay * 30 * cur-
>individual.count;
                    cur = cur->next;
                }
                if (money > 0) // если нашлось
                   printf("Feed costs for all animals in this natural area:
%.2f\n", money);
                else
                    puts("There is no such natural area");
                puts("To return to the menu, press Enter");
                while (getchar() != '\n')
                    ;
                return;
            }
            case 3: // выход
            {
                return;
            }
            }
```

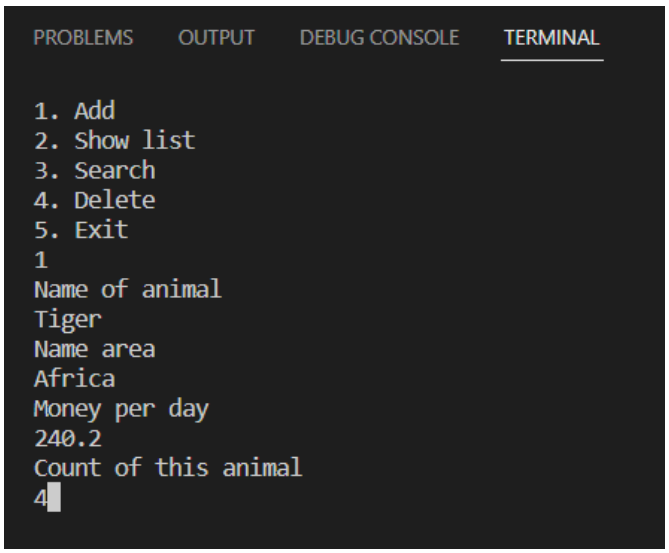
```
    } while (menu != 3);
}
```

Результаты работы программы:

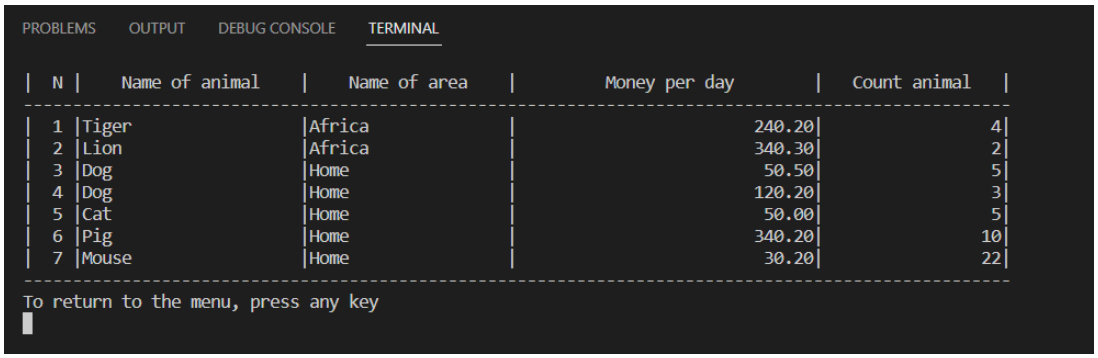
Основное меню.



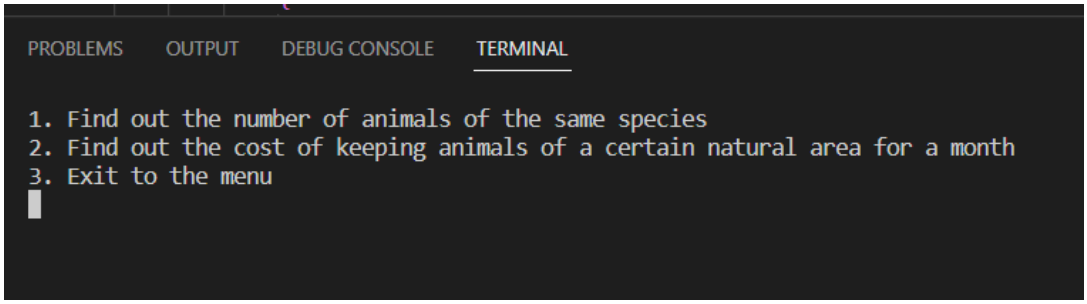
Добавление записи.



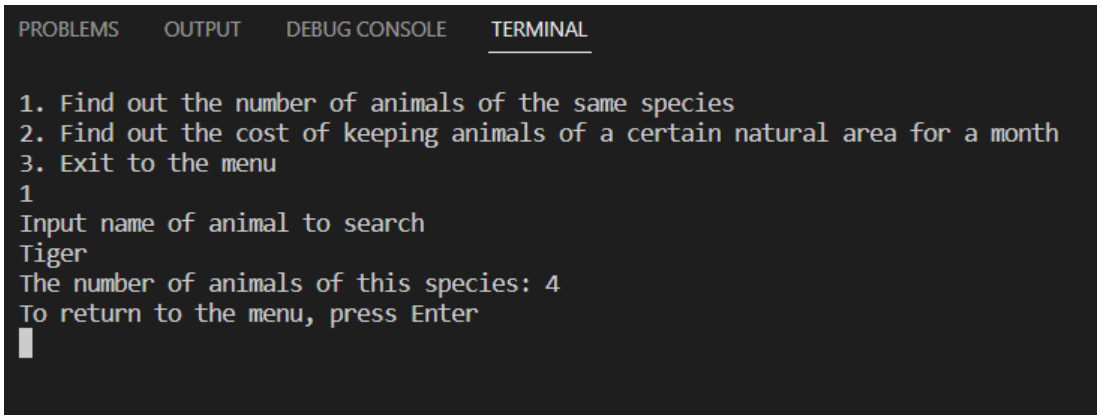
Показ записей.



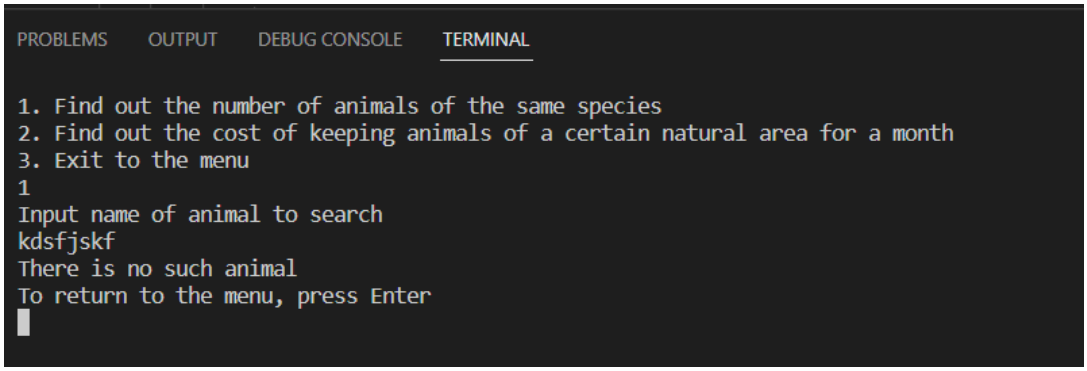
Меню поиска записей.



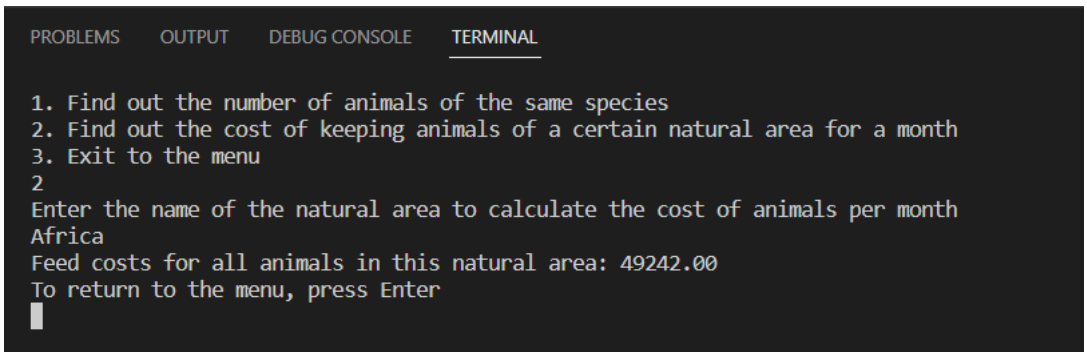
Если пользователь хочет найти количество животных определнного вида.



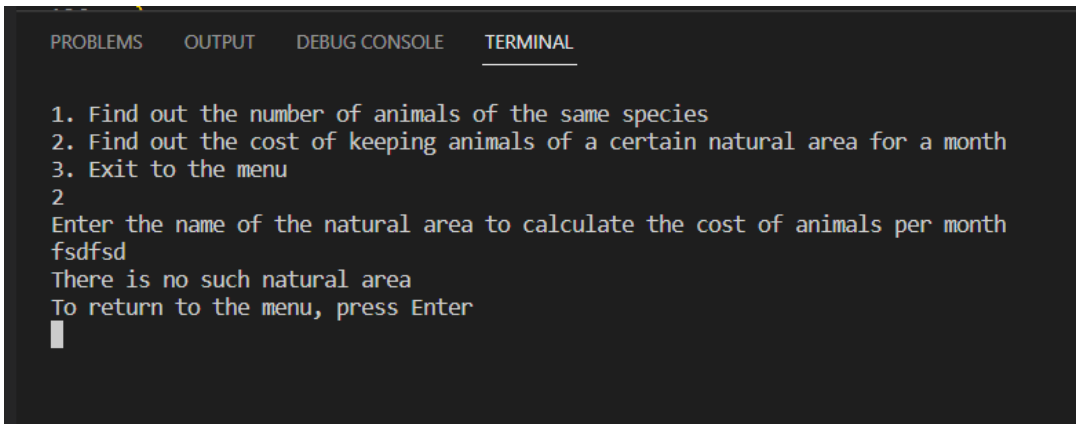
Если животное, которое ввел пользователь не нашлось.



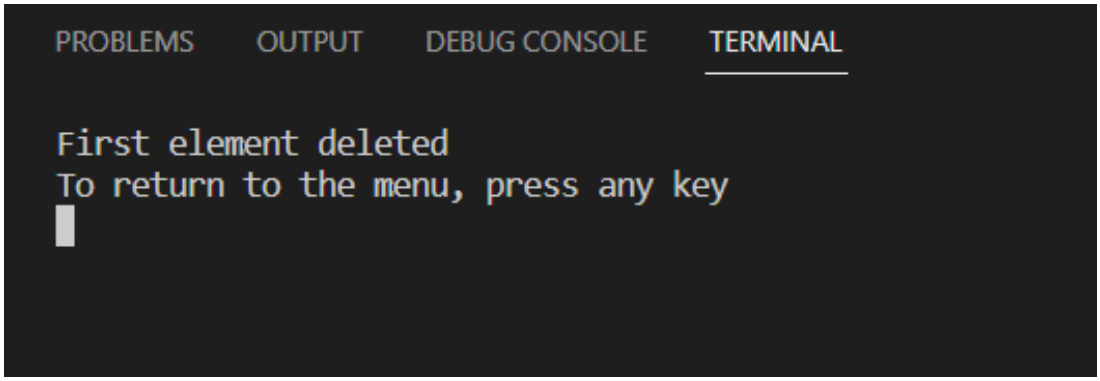
Если пользователь хочет найти затраты на корм животных определенной природной зоны.



Если пользователь ввел природную зону, которой нет.



Удаление первой записи.



Если список пуст.