Отчёт по лабораторной работе № VIII

по курсу 1 Практикум на ЭВМ

студента группы М8О-102Б-21 Кажекин Денис,

№ по списку 9

Адреса www, e-mail, jabber, skype deniskazhekin@mail.ru

Работа выполнена: "____" _____ 20____г.

Преподаватель: доцент каф. 806 Никулин С.П.

Входной контроль знаний с оценкой _____

Отчёт сдан "____" _____ 20____г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема**: Линейный списки
2. **Цель работы**: Составить программу на языке Си для обработки линейного списка заданной организации с отображением списка на динамические структуры.
3. **Задание** (*Номер по списку № 9)*

   Тип элемента списка: перечислимый.

   Вид списка: линейный двунаправленный.

   Нестандартное действие: добавить k экземпляров последнего элемента в начало списка

4. **Оборудование** (*лабораторное*):
   Оборудование ПЭВМ студента, если использовалось:
   Процессор: Apple M1, с ОП 8192 Мб, НМД 262144 Мб. Монитор: Retina 13,3; IPS 2560 x 1600.
5. **Программное обеспечение** (*лабораторное*):
   Операционная система семейства Linux, наименование Ubuntu, версия ____20.04.3 LTS____интерпретатор команд _bash_ версия __5.0.17(1)__
   Редактор текстов GNU emacs, версия 27.2
   Прикладные системы и программы
6. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями)

Идея:

Программа работает со списком и может выполнять несколько функций:

1) Печать списка
2) Добавить новый элемент
3) Удалить элемент из списка по индексу
4) Печать длины списка
5) Выполнить нестандартное действие
6) Выход

7.Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты, либо соображения по тестированию].

Функции

typedef struct Node, typedef struct List – структуры списка и элементов
void Lappend(List* list, Node* new_elem) – добавляет элемент списка в список
void add_to_the_beginning(List* list, Node* node) – добавляет элемент в начало списка

void print_list(List *list) – выводит список
void delete_element(List* list, int index) – удалить элемент списка
List* create_list() – создаёт список
Node* create_node(double zn) – создаёт элемент списка
int menu() – меню
int main(int argc, char* argv[]) – главная функция
void delete_list(List *list) – удалить список
void action(List* list) – выполнить действие
int Llength(List *list) – вычисляет длину списка

*Пункты 1-7 отчёта составляются **строго до** начала лабораторной работы.*

Допущен к выполнению работы. Подпись преподавателя _____

7. **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

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

enum Number{zero, one, two, three, four, five};


int menuchisel() {
   printf("%s\n", "0. Ноль");
   printf("%s\n", "1. Один");
   printf("%s\n", "2. Два");
   printf("%s\n", "3. Три");
   printf("%s\n", "4. Четыре");
   printf("%s\n", "5. Пять");
   int k;
   scanf("%d", &k);
   return k;
}

typedef struct Node
{
   enum Number number;
   struct Node* next;
   struct Node* prev;
} Node;

int menu() {
   printf("%s\n", "1. Печать списка");
   printf("%s\n", "2. Добавить новый элемент");
   printf("%s\n", "3. Удалить элемент");
   printf("%s\n", "4. Печать длины списка");
   printf("%s\n", "5. Нестандартное действие");
   printf("%s\n", "6. Выход");
   int k;
   scanf("%d", &k);
   return k;
}

typedef struct List
{
   Node* head;
} List;

void print_list(List *list){
   Node *now = list->head;
   if(now == NULL){
      printf("Список пуст\n");
      return;
   }
   while(now){
      printf("%d ", now->number);
      now = now->next;
   }
   printf("\n");
}

List* create_list()
```

```
{
    return (List*)calloc(1, sizeof(List));
}

Node* create_node(enum Number number)
{
    Node* node = (Node*)malloc(sizeof(Node));
    node->number = number;
    node->next = NULL;
    node->prev = NULL;
    return node;
}
void delete_list(List *list) {
    Node *ptr = list->head, *ptr_prev;
    while (ptr) {
        ptr_prev = ptr;
        ptr = ptr->next;
        free(ptr_prev);
    }
    free(list);
}



Node *get_i (List *list, int index)
{
    Node* now = list->head;
    for (int i = 0; i < index; i++)
        now = now->next;
    return now;
}

int Llength(List *list)
{
    int kolvo = 0;
    Node* now = list->head;
    while(now){
        now = now->next;
        kolvo++;
    }
    return kolvo;
}



void delete_element(List* list, int index)
{
    Node* node = get_i(list, index);
    if(list->head == node){
        list->head = node->next;
        node->next->prev = NULL;
        free(node);
    }
    else {
        if (node->prev)
            node->prev->next = node->next;
        if (node->next != NULL)
            node->next->prev = node->prev;
        free(node);
    }
}


void add_to_the_beginning(List* list, Node* node){
    list->head->prev = node;
    node->next = list->head;
    list->head = node;
}


void action(List* list){
    Node* now = list->head;
```

```
    enum Number number;
    int k;
    printf("Сколько раз добавить последний элемент в начало списка:");
    scanf("%d", &k);
    if(k == 0)
        return;
    while (now->next != NULL)
        now = now->next;
    number = now->number;
    for(int i = 0; i < k; i++){
        add_to_the_beginning(list, create_node(number));
    }
}


void Lappend(List* list, Node* new_elem, int index)
{
    if(index < 0){
        printf("Error");
        exit(1);
    }
    else if (list->head == NULL)
    {
        list->head = new_elem;
        new_elem->next = NULL;
        new_elem->prev = NULL;
        return;
    }
    else if(index == 0)
    {
        Node* node = get_i(list, index);
        node->prev = new_elem;
        new_elem->next = node;
        new_elem->prev = NULL;
        list->head = new_elem;
    }
    else if(index > Llength(list)){
        printf("Error");
        exit(1);
    }
    else if(index == Llength(list)){
        Node *node_prev = get_i(list, index - 1);
        new_elem->prev = node_prev;
        node_prev->next = new_elem;
    }
    else{
        Node *node = get_i(list, index);
        Node *node_prev = get_i(list, index - 1);
        new_elem->next = node;
        node->prev = new_elem;
        new_elem->prev = node_prev;
        node_prev->next = new_elem;
    }
}

int main(int argc, char* argv[]) {
    int znach;
    char z;
    int index;
    List* list = create_list();

    int k = 0;
    while (k != 6) {
        k = menu();
        switch (k)
        {
            case 1:
            {
                printf("List:\n");
                print_list(list);
                break;
            }
```

```
                case 2:
                {
                    printf("Введите число, которое хотите добавить:\n");
                    int l = menuchisel();
                    printf("Введите номер позиции, на которую поставить число:");
                    scanf("%d", &index);
                    Lappend(list, create_node(l), index);
                    break;
                }
                case 3:
                {
                    printf("Введите индекс элемента, который хотите удалить:");
                    scanf("%d", &index);
                    delete_element(list, index);
                    break;
                }
                case 4:
                {
                    printf("Длина списка:");
                    printf("%d\n", Llength(list));
                    break;
                }
                case 5:
                {
                    action(list);
                    break;
                }
                case 6:
                {
                    break;
                }
                default:
                    printf("%s\n", "Попробуйте снова");
            }
        }

        delete_list(list);
        return 0;
    }
deniskazhekin@MacBook-Air-Denis ~ % gcc main.c -o main.out
deniskazhekin@MacBook-Air-Denis ~ % ./main.out
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
Список пуст
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
4
Длина списка:0
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
```

0
Введите номер позиции, на которую поставить число:0
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
0
Введите номер позиции, на которую поставить число:1
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
3
Введите номер позиции, на которую поставить число:2
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
4
Введите номер позиции, на которую поставить число:3
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
5
Сколько раз добавить последний элемент в начало списка:2
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
4 4 0 0 3 4
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие

6. Выход
3
Введите индекс элемента, который хотите удалить:0
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
3
Введите индекс элемента, который хотите удалить:1
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
3
Введите индекс элемента, который хотите удалить:2
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
4 0 4
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
3
Введите индекс элемента, который хотите удалить:1
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
4 4
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
4
Введите номер позиции, на которую поставить число:2
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
4 4 4
1. Печать списка
2. Добавить новый элемент

3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
4
Введите номер позиции, на которую поставить число:3
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
2
Введите число, которое хотите добавить:
0. Ноль
1. Один
2. Два
3. Три
4. Четыре
5. Пять
5
Введите номер позиции, на которую поставить число:4
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
4 4 4 4 5
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
5
Сколько раз добавить последний элемент в начало списка:2
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
1
List:
5 5 4 4 4 4 5
1. Печать списка
2. Добавить новый элемент
3. Удалить элемент
4. Печать длины списка
5. Нестандартное действие
6. Выход
6

9.Дневник отладки должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или	Дата	Время	Событие	Действие по исправлению	Примечание

	дом.					

10.Замечание автора по существу работы _____

11.Выводы Я выполнил работу и научился работать со списком

_____

Недочеты, допущенные при выполнении задания, могут быть устранены следующим образом

Подпись студента _____