**Отчёт по лабораторной работе №25-26**

по курсу «Языки и методы программирования».

Выполнил студент группы **М8О-114БВ-24**: **Дылдин Сергей Владиславович** № по списку **10**.

Контакты: [**dsergey010206@gmail.com**](mailto:dsergey010206@gmail.com)

Работа выполнена: «28» мая 2025 г.

Преподаватель: **каф. 806 Никулин Сергей Петрович**

Входной контроль знаний с оценкой: **\_\_\_\_\_**

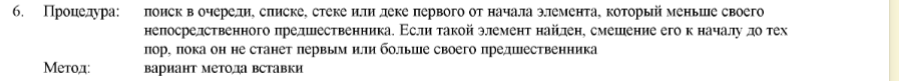
Отчет сдан «30» мая 2025 г.

Итоговая оценка: \_\_\_\_\_

Подпись преподавателя: \_\_\_\_\_\_\_\_\_\_

1. **Тема:** динамические структуры данных, обработка деревьев.

2. **Цель работы:** составить программу на языке Си для построения и обработки списка, содержащего элементы типа int и создать Makefile.

3. **Задание: 4.Линейный список** .

4. **Оборудование**: *Оборудование ПЭВМ студента, если использовалось:*

Процессор **Intel core i5 12400H**, ОП **16** ГБ, SSD **1**TB.

5. **Программное обеспечение:** *Программное обеспечение ПЭВМ студента, если использовалось****:***

Операционная система семейства **Ubuntu**, наименование версия **VirtualBox Ubuntu 20.04.3 LTS**,

интерпретатор команд **bash** версия **5.0.17**. Система программирования **C**.Редактор текстов **VI** версия **8.1**

6**. Идея, метод, алгоритм** *решения задачи [в формах: словесной, псевдокода, графической (блок-схема, диаграмма, рисунок, таблица) или формальные спецификации с пред- и постусловиями]:*

*Составление программы на Си, выполняющей следующие задачи:*Инициализация списка:  
 При запуске программы вызывается функция list\_init, которая устанавливает указатель на начало списка (head) в NULL, тем самым создавая пустой список.

**Метод решения**:

- Использование односвязного списка для хранения данных. Каждый узел (node) содержит значение и указатель на следующий элемент.

- Инкапсуляция логики списка в структуру list, которая хранит указатель на первый узел (header) и размер списка.

- Реализация операций через функции, работающие с указателями.

**Алгоритмы ключевых операций**:

1. Добавление элемента (push):

- Если индекс = 0, элемент становится новым заголовком.

- Для других индексов находится предыдущий узел, и элемент вставляется после него.

- Проверка корректности индекса (0 ≤ индекс ≤ размер списка).

2. Удаление элемента (pop\_front, pop\_back, pop\_by\_ind)

- Корректировка указателей соседних узлов и освобождение памяти.

- Для pop\_back используется поиск предпоследнего узла.

3. Сортировка (insertion\_sort):

- Алгоритм сортировки вставками. Создается новый отсортированный список (sorted).

- Каждый элемент исходного списка последовательно вставляется в sorted на правильную позицию.

- После завершения обновляется заголовок списка (header) на начало sorted.

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

linked\_list.h

#ifndef \_\_list\_H\_\_

#define \_\_list\_H\_\_

struct node {

int value;

struct node \*next;

};

typedef struct node node;

struct list {

node \*header;

int size;

};

typedef struct list list;

void init\_list(list \*cur\_list);

int empty(list \*cur\_list);

node \*find\_by\_index(node \*cur, int need\_ind);

void push(list \*cur\_list, int val, int ind);

void pop\_front(list \*cur\_list);

void pop\_back(list \*cur\_list);

void pop\_by\_ind(list \*cur\_list, int ind);

int size(list \*cur\_list);

void el\_by\_ind(list \*cur\_list, int ind);

void clear(list \*cur\_list);

void print\_list(list \*cur\_list);

int size(list \*cur\_list);

void insertion\_sort(list \*cur\_list);

#endif

**oper.c**

#include "linked\_list.h"

#include <stdio.h>

#include <stdlib.h>

void init\_list(list \*cur\_list) {

cur\_list->header = NULL;

cur\_list->size = 0;

}

int empty(list \*cur\_list) {

return cur\_list->size == 0;

}

node \*find\_by\_index(node \*cur, int need\_ind) {

for (int i = 0; cur != NULL && i < need\_ind; i++) {

cur = cur->next;

}

return cur;

}

void push(list \*cur\_list, int val, int ind) {

if (ind < 0 || ind > cur\_list->size) {

printf("Неверный индекс\n");

return;

}

node \*new\_node = malloc(sizeof(node));

new\_node->value = val;

new\_node->next = NULL;

if (ind == 0) {

new\_node->next = cur\_list->header;

cur\_list->header = new\_node;

} else {

node \*prev = find\_by\_index(cur\_list->header, ind - 1);

if (prev == NULL) {

free(new\_node);

return;

}

new\_node->next = prev->next;

prev->next = new\_node;

}

cur\_list->size++;

}

int size(list \*cur\_list) {

return cur\_list->size;

}

void pop\_front(list \*cur\_list) {

if (empty(cur\_list)) return;

node \*temp = cur\_list->header;

cur\_list->header = temp->next;

free(temp);

cur\_list->size--;

}

void pop\_back(list \*cur\_list) {

if (empty(cur\_list)) return;

if (cur\_list->size == 1) {

pop\_front(cur\_list);

return;

}

node \*prev = find\_by\_index(cur\_list->header, cur\_list->size - 2);

free(prev->next);

prev->next = NULL;

cur\_list->size--;

}

void pop\_by\_ind(list \*cur\_list, int ind) {

if (ind < 0 || ind >= cur\_list->size) return;

if (ind == 0) {

pop\_front(cur\_list);

return;

}

node \*prev = find\_by\_index(cur\_list->header, ind - 1);

node \*temp = prev->next;

prev->next = temp->next;

free(temp);

cur\_list->size--;

}

void clear(list \*cur\_list) {

while (!empty(cur\_list)) {

pop\_front(cur\_list);

}

}

void print\_list(list \*cur\_list) {

node \*current = cur\_list->header;

while (current != NULL) {

printf("%d ", current->value);

current = current->next;

}

printf("\n");

}

void el\_by\_ind(list \*cur\_list, int ind) {

if (ind < 0 || ind >= cur\_list->size) {

printf("Неверный индекс\n");

return;

}

node \*n = find\_by\_index(cur\_list->header, ind);

printf("Значение %d на индексе %d\n", n->value, ind);

}

void insertion\_sort(list \*cur\_list) {

if (cur\_list->size < 2) return;

node \*sorted = NULL; // начало нового отсортированного списка

node \*current = cur\_list->header;

while (current != NULL) {

node \*next = current->next;

// вставляем current в нужное место в отсортированном списке

if (sorted == NULL || current->value < sorted->value) {

current->next = sorted;

sorted = current;

} else {

node \*temp = sorted;

while (temp->next != NULL && temp->next->value < current->value) {

temp = temp->next;

}

current->next = temp->next;

temp->next = current;

}

current = next;

}

cur\_list->header = sorted;

}

**main.c**

#include "linked\_list.h"

#include <stdio.h>

void display\_menu() {

printf("\nМеню:\n");

printf("1. Добавить элемент в начало\n");

printf("2. Добавить элемент в конец\n");

printf("3. Добавить элемент по индексу\n");

printf("4. Удалить первый элемент\n");

printf("5. Удалить последний элемент\n");

printf("6. Удалить элемент по индексу\n");

printf("7. Вывести список\n");

printf("8. Отсортировать список\n");

printf("9. Выход\n");

printf("Выберите действие: ");

}

int main() {

list our\_list;

init\_list(&our\_list);

int choice, val, ind;

while (1) {

display\_menu();

if (scanf("%d", &choice) != 1) break;

switch (choice) {

case 1:

printf("Введите значение: ");

scanf("%d", &val);

push(&our\_list, val, 0);

print\_list(&our\_list);

break;

case 2:

printf("Введите значение: ");

scanf("%d", &val);

push(&our\_list, val, size(&our\_list)); // Корректно, так как индексы от 0 до size

print\_list(&our\_list);

break;

case 3: // Добавить по индексу

printf("Введите значение и индекс: ");

scanf("%d %d", &val, &ind);

push(&our\_list, val, ind);

print\_list(&our\_list);

break;

case 4:

pop\_front(&our\_list);

print\_list(&our\_list);

break;

case 5:

pop\_back(&our\_list);

print\_list(&our\_list);

break;

case 6:

printf("Введите индекс: ");

scanf("%d", &ind);

pop\_by\_ind(&our\_list, ind);

print\_list(&our\_list);

break;

case 7:

print\_list(&our\_list);

break;

case 8:

insertion\_sort(&our\_list);

break;

case 9:

clear(&our\_list);

return 0;

break;

default:

printf("Неверный выбор!\n");

}

}

clear(&our\_list);

return 0;

}

**Makefile**

CC = gcc

CFLAGS = -Wall -Wextra -std=c11

SRC = main.c oper.c

OBJ = main.o oper.o

EXEC = program.exe

all: $(EXEC)

$(EXEC): $(OBJ)

$(CC) $(CFLAGS) -o $(EXEC) $(OBJ)

main.o: main.c linked\_list.h

$(CC) $(CFLAGS) -c main.c -o main.o

oper.o: oper.c linked\_list.h

$(CC) $(CFLAGS) -c oper.c -o oper.o

clean:

-del /F /Q $(OBJ) $(EXEC) 2> NUL

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

*Допущен к выполнению работы.*  **Подпись преподавателя \_\_\_\_\_\_\_**

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

PS D:\apps\labi\2sem\laba4> make

gcc -Wall -Wextra -std=c11 -c main.c -o main.o

gcc -Wall -Wextra -std=c11 -c oper.c -o oper.o

gcc -Wall -Wextra -std=c11 -o program.exe main.o oper.o

PS D:\apps\labi\2sem\laba4> ls

Directory: D:\apps\labi\2sem\laba4

Mode LastWriteTime Length Name

---- ------------- ------ ----

-a---- 16.03.2025 22:27 997 linked\_list.h

-a---- 30.03.2025 21:25 2537 main.c

-a---- 30.03.2025 22:20 3501 main.o

-a---- 30.03.2025 21:21 363 Makefile

-a---- 16.03.2025 22:25 4021 oper.c

-a---- 30.03.2025 22:20 3092 oper.o

-a---- 30.03.2025 22:20 143571 program.exe

PS D:\apps\labi\2sem\laba4> ./program

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 1

1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 2

2 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 3

3 2 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 4

4 3 2 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 5

5 4 3 2 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 3

Введите значение и индекс: 6 0

6 5 4 3 2 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 3

Введите значение и индекс: 7

5

6 5 4 3 2 7 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 2

Введите значение: 1

6 5 4 3 2 7 1 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 4

5 4 3 2 7 1 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

5 4 3 2 7 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 6

Введите индекс: 2

5 4 2 7 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 8

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 7

1 2 4 5 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 1

Введите значение: 10

10 1 2 4 5 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 3

Введите значение и индекс: 2

1

10 2 1 2 4 5 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 2

Введите значение: 0

10 2 1 2 4 5 7 0

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 7

10 2 1 2 4 5 7 0

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 8

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 7

0 1 2 2 4 5 7 10

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1 2 2 4 5 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1 2 2 4 5

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1 2 2 4

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1 2 2

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1 2

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

0 1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 4

1

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 4

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 5

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 4

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 7

Меню:

1. Добавить элемент в начало

2. Добавить элемент в конец

3. Добавить элемент по индексу

4. Удалить первый элемент

5. Удалить последний элемент

6. Удалить элемент по индексу

7. Вывести список

8. Отсортировать список

9. Выход

Выберите действие: 9

PS D:\apps\labi\2sem\laba4>

9. **Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) 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 дом.** | **Дата** | **Время** | **Событие** | **Действие по исправлению** | **Примечание** |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |

10. **Замечания автора** по существу работы: замечания отсутствуют.

11. **Выводы:** научился работать со структурой данных - линейный список и с сортировкой вставкой

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_