**Отчет по курсовому проекту №8** по курсу фундаментальная информатика

Студентка группы М8О-113Б-22, № по списку 1, Астахова Анастасия Сергеевна

Контакты astakhovaanastasia0201@gmail.com

Работа выполнена: « » \_\_\_\_\_\_\_\_ 2023 г.

Преподаватель: доцент каф. 806 Никулин Сергей Петрович

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_2023 г., итоговая оценка \_\_\_\_\_

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

* 1. **Тема:** Линейные списки

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Цель работы:** Составить и отладить программу на языке Си для обработки линейного списка заданной организации с отображением на динамические структуры.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Задание (тип элемента списка №6, вид списка №1):** написать кольцевой однонаправленный список, состоящий из комплексных чисел. Функции: печать списка, вставка нового элемента в список, удаление элемента из списка, подсчет длины списка, удаление всего списка, если в нем есть заданный элемент. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. **Оборудование** (лабораторное):

ЭВМ \_\_\_\_\_\_\_\_\_, процессор \_\_\_\_ \_\_, имя узла сети \_\_\_\_\_\_ с ОП \_\_\_\_\_\_\_\_ Мб, НМД \_\_\_\_\_\_\_ Мб. Терминал \_\_\_\_\_ адрес \_\_\_\_\_\_\_\_\_\_. Принтер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Другие устройства \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Оборудование ПЭВМ студента, если использовалось:*

Ноутбук: Процессор Intel Core i5-10210U с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь Bloody R80

*Компьютер: Процессор Ryzen 5 3600 с ОП 16 ГБ, SSD 512 ГБ. Другие устройства: мышь* Razer BASILISK X HYPERSPEED, [*Logitech G910 (920-008019) Orion Spectrum RGB Mechanical Gaming*](https://www.komus.ru/katalog/tekhnika/kompyuternaya-tekhnika/klaviatury-i-myshi/klaviatury-provodnye/klaviatura-logitech-g910-920-008019-orion-spectrum-rgb-mechanical-gaming/p/659916/?from=n-13776), монитор [*LG UltraGear 27GP750*](https://www.lg.com/ru/monitors/lg-27gp750-b)

1. **Программное обеспечение (лабораторное):**

Операционная система семейства \_\_\_Unix\_\_\_\_, наименование \_\_\_\_\_Ubuntu\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_4.15.0\_\_\_\_\_\_

интерпретатор команд \_\_\_\_bash\_\_\_\_\_\_\_\_ версия \_\_\_4.4.20\_\_\_\_

Система программирования \_\_С\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_последняя\_\_\_\_\_\_\_\_\_

Редактор текстов \_\_\_gedit\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_25.2.2 \_\_\_\_\_\_\_\_\_

Утилиты операционной системы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Прикладные системы и программы \_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных\_\_stud/208104 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Ноутбук: Операционная система семейства Ubuntu, наименование версия Ubuntu 20.04.3 LTS,

*интерпретатор команд bash версия 5.0.17. Система программирования C. Редактор текстов gedit версия 28.2. Утилиты: стандартные утилиты UNIX. Программное обеспечение: стандартное программное обеспечение системы UNIX.*

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

Будет написана структура комплексного числа, функция, которая принимая действительную и мнимую часть, будет возвращать число, комплексного типа, а также функция, печатающая комплексное число.

Лист:

Будут созданы две структуры: элемент списка и сам список. Элемент списка будет хранить указатель на следующтй элемент и ключ. Лист будет хранить свой размер и указатель на последний элемент.

Будут написаны функции для взаимодействия со списком: инициализация, итератор (возвращает указатель на элемент из списка, который находится под указанным индексом, функция, которая возвращает размер списка, пуст ли список, функция добавления элемента на указанное место, в конец, функция, которая удаляет элемент из указанного места, функция которая, удаляет список, функция, которая печатает список, а также функция, которая удалит список, если в нем есть указанный элемент.

1. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию].

PS D:\c\_labs\cp\_8> ./list

Input your command

Create list: 1 r i (r is real part, i is complex)

Add elem after another elem, with inputed index: 2 ind r i

Push back an elem: 3 r i

Del elem, that has inputed index: 4 ind

Length of list: 5

Clear list, if in contains inputed number: 6 r i

Print list: 7

Exit: 0

1 1 2

3 2 3

3 3 4

3 4 -5

7

1+2i; 2+3i; 3+4i; 4-5i;

2 0 2 2

7

1+2i; 2+2i; 2+3i; 3+4i; 4-5i;

4 3

7

1+2i; 2+2i; 2+3i; 4-5i;

5

The size of an elem is 4

6 7 7

7

1+2i; 2+2i; 2+3i; 4-5i;

6 1 2

7

List is empty!

0

PS D:\c\_labs\cp\_8>

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

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

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами,подписанныйпреподавателем).

Код лабораторной работы:

#include <stdio.h>

#include <stdbool.h>

#include <stdlib.h>

// #include "list.h"

typedef struct {

int real;

int imaginary;

} complex;

complex cmplx(int r, int im) {

complex c;

c.real = r;

c.imaginary = im;

return c;

}

void cmplx\_print(complex elem) {

if (elem.imaginary < 0) {

printf("%d%di", elem.real, elem.imaginary);

}

else {

printf("%d+%di", elem.real, elem.imaginary);

}

}

typedef struct node{

complex key;

struct node \*ptr\_next;

} node;

typedef struct {

int size;

node \*ptr\_last;

} list;

list \*lst\_init(list \*l, complex elem) {

l = (list\*)malloc(sizeof(list));

l->size = 1;

node \*new\_node;

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

new\_node->key = elem;

new\_node->ptr\_next = new\_node;

l->ptr\_last = new\_node;

//return l;

}

node \*lst\_ptrwind(list \*l, int ind) { //выдает указатель на элемент с введенным индексом

node \*nd = l->ptr\_last;

int sz = l->size;

if (ind >= sz) {

printf("Элемента с таким индексом нет\n");

return NULL;

}

for (int i = 0; i <= ind; i++) {

nd = nd->ptr\_next;

}

return nd;

}

int lst\_size(list \*lst){

return lst->size;

}

bool lst\_isempty(list \*lst) {

return (lst->size == 0) ? true : false;

}

void lst\_addelem(list \*lst, int ind, complex elem) { // осуществляет вставку элемента после того, чей индекс указан

node \*new\_node;

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

new\_node->key = elem;

node \*after = lst\_ptrwind(lst, ind);

node \*tmp = after->ptr\_next;

after->ptr\_next = new\_node;

new\_node->ptr\_next = tmp;

lst->size++;

}

void lst\_pushback(list \*lst, complex elem) {

node \*new\_node;

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

new\_node->key = elem;

if (lst->size == 0) {

lst->ptr\_last = new\_node;

new\_node->ptr\_next = new\_node;

lst->size++;

}

node \*tmp = lst->ptr\_last->ptr\_next;

lst->ptr\_last->ptr\_next = new\_node;

new\_node->ptr\_next = tmp;

lst->ptr\_last = new\_node;

lst->size++;

}

void lst\_delelem(list \*lst, int ind) { // удаляет элемент под заданным индексом

if (lst->size == 1) {

node \*del\_elem;

del\_elem = lst->ptr\_last;

del\_elem = NULL;

lst->size--;

lst->ptr\_last = del\_elem;

return;

}

node \*del\_ptr;

del\_ptr = lst\_ptrwind(lst, ind);

node \*tmp = del\_ptr;

while (tmp->ptr\_next != del\_ptr) {

tmp = tmp->ptr\_next;

}

tmp->ptr\_next = del\_ptr->ptr\_next;

free(del\_ptr);

lst->size--;

}

void lst\_delall(list \*lst) {

int sz = lst->size;

node \*tmp1 = lst->ptr\_last;

node \*tmp2;

node \*tmp3;

for (int i = 1; i < sz; i++) {

tmp2 = tmp1->ptr\_next;

tmp3 = tmp1;

free(tmp3);

tmp1 = tmp2;

}

tmp1 = NULL;

lst->ptr\_last = tmp1;

//tmp1->ptr\_next = NULL;

lst->size = 0;

}

void lst\_custom(list \*lst, complex elem) { // функция удаляет весь список, если в нем есть заданный элемент

int sz = lst->size;

node \*tmp = lst->ptr\_last;

for (int i = 0; i <= sz; i++) {

tmp = tmp->ptr\_next;

if (tmp->key.real == elem.real && tmp->key.imaginary == elem.imaginary) {

lst\_delall(lst);

}

}

}

void lst\_print(list \*lst) {

int sz = lst->size;

if (sz == 0) {

printf("List is empty!\n");

return;

}

node \*tmp = lst->ptr\_last;

for (int i = 0; i < sz; i++) {

tmp = tmp->ptr\_next;

cmplx\_print(tmp->key);

printf("; ");

}

printf("\n");

}

void pr\_menu() {

printf("Input your command\n");

printf("Create list: 1 r i (r is real part, i is complex)\n");

printf("Add elem after another elem, with inputed index: 2 ind r i\n");

printf("Push back an elem: 3 r i\n");

printf("Del elem, that has inputed index: 4 ind\n");

printf("Length of list: 5\n");

printf("Clear list, if in contains inputed number: 6 r i\n");

printf("Print list: 7\n");

printf("Exit: 0\n");

}

int main() {

list \*lst;

pr\_menu();

int q;

while(1) {

scanf("%d", &q);

if (q == 1) {

int r, i;

scanf("%d%d", &r, &i);

complex elem = cmplx(r, i);

lst = lst\_init(lst, elem);

}

if (q == 2) {

int ind, r, i;

scanf("%d%d%d", &ind, &r, &i);

complex elem = cmplx(r, i);

lst\_addelem(lst, ind, elem);

}

if (q == 3) {

int r, i;

scanf("%d%d", &r, &i);

complex elem = cmplx(r, i);

lst\_pushback(lst, elem);

}

if (q == 4) {

int ind;

scanf("%d", &ind);

lst\_delelem(lst, ind);

}

if (q == 5) {

int ans = lst\_size(lst);

printf("The size of an elem is %d\n", ans);

}

if (q == 6) {

int r, i;

scanf("%d%d", &r, &i);

complex elem = cmplx(r, i);

lst\_custom(lst, elem);

}

if (q == 7) {

lst\_print(lst);

}

if (q == 0) {

return 0;

}

}

return 0;

}

1. **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. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  | или |  |  |  |  |  |
|  | дом. |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы:

Замечания отсутствуют.

1. **Выводы**

Я научилась писать свой список с указанной структурной реализацией, работать с ним.

Недочёты при выполнении задания могут быть устранены следующим образом:

Недочеты отсутствуют.

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