Отчёт по лабораторной работе № VIII

по курсу: ЯМП

студента группы: Чурилов С. Э. М8О-103Б-20, № по списку: 29

Адреса www, e-mail, jabber, skype churilov.ser1204@gmail.com

Работа выполнена: "___" _____ 20____ г.

Преподаватель: каф. 806 В. К. Титов

Входной контроль знаний с оценкой _____

Отчёт сдан "___" _____ 20____ г., итоговая оценка _____

Подпись преподавателя _____

1 **Тема**: Линейные списки

2 **Цель работы**: Составить и отладить программу на языке С для обработки линейного списка заданной организации с отображением списка на динамические структуры

3 **Задание** (*вариант №* 29): Тип элемента списка — строковый/вид списка - линейный однонаправленный с барьерным элементом/переставить первую и вторую половины списка

4 **Оборудование** (*лабораторное*):
ЭВМ MSI GT70 0ND 447-RU , процессор Intel Celeron 2.4 GhZ , имя узла сети cameron с ОП 15 ГБ
НМД _____ ГБ. Терминал _____ адрес _____. Принтер _____
Другие устройства _____

Оборудование ПЭВМ студента, если использовалось:
Процессор Ryzen 3 3200U @ 4x 2.6GHz , ОП 8192 МБ, НМД _____ ГБ. Монитор: встроенный
Другие устройства _____

5 **Программное обеспечение** (*лабораторное*):
Операционная система семейства UNIX, наименование: Ubuntu версия 18.04.5 LTS
Интерпретатор команд: bash версия _____
Система программирования: C версия _____
Редактор текстов: Emax версия _____
Утилиты операционной системы: _____
Прикладные системы и программы: _____
Местонахождения и имена файлов программ и данных: _____

Программное обеспечение ЭВМ студента, если использовалось:
Операционная система семейства UNIX, наименование Ubuntu версия 20.04 LTS
Интерпретатор команд: bash версия _____
Система программирования: C версия _____
Редактор текстов: Emax версия _____
Утилиты операционной системы: _____
Прикладные системы и программы: _____
Местонахождения и имена файлов программ и данных: _____

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

Принцип работы программы:

1. Создается бесконечный цикл, внутри которого запрашиваются команды от пользователя и параметры к командам

2. Далее каждая команда обрабатывается своей функцией

Функции:

1. add_first(str) — функция добавления элемента в начало списка принцип работы: если первый элемент равен барьерному элементу, значит список пустой, тогда новый элемент равен первому и последнему и указывающий на барьерный элемент, иначе новый элемент будет указывать на старый первый элемент

2. add_back(str) — функция добавления элемента в конец списка принцип работы: если первый элемент равен барьерному элементу, значит список пустой, тогда новый элемент равен первому и последнему и указывающий на барьерный элемент, иначе старый последний элемент ссылается на новый элемент, который ссылается на барьерный элемент

3. remove_element(str) — перегруженная функция удаления первого встретившегося элемента с заданным значением принцип работы: необходимо пройтись по всем элементам, если элемент не был найден, то вывести соответствующее сообщение, иначе необходимо ссылавшемуся на него элементу определить новою ссылку на следующий элемент за найдены.

4. move() - перестановка первой и второй половины списка

5. print_lenght() — вывод длины списка

6. print_list() — вывод элементов списка

7. str_cmp(str, str) — сравнение строк на равенство принцип работы: необходимо пройтись по элементам строк, если элементы строк под одними индексами не равны, вернуть что строки не равны, иначе идти дальше пока не встретится элемент, соответствующий концу строки, тогда, если оба символа равны, то вернуть, что строки равны, иначе что не равны

8. print_menu() — вывести список команд

9. insert_element(str, str) — вставка элемента в список после заданного элемента принцип работы: пройтись по всем элементам, если не был найден, после которого нужно добавить новый, то вывести соответствующее сообщение, иначе для нового элемента определить ссылку на элемент после которого нужно вставить, и определить на него ссылку.

10. erase_list() — очистка списка

11. generate_list(int) — генерация списка с заданным числом элементов

12. get_random_str(int) — генерация случайной строки

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

План работы:
1. Придумать способ тестирования
2. Создать ядро программы
3. Создать программу
4. Провести рефакторинг
5. Отладить основную программу.

Для тестирования программы использована функция генерации списка (generate_list(int))

generate_list(int) — создает заданное кол-во элементов списка. Для создания элементов используется функция get_random_str(int), которая создает строку заданной длины. Из generate_list функция get_random_str вызывается со значениями от 3 до 13.

Допущен к выполнению работы. Подпись преподавателя _____

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

```
leninware@leninware-VirtualBox:~/LRVIII$ date
Пт 30 апр 2021 14:05:30 MSK
leninware@leninware-VirtualBox:~/LRVIII$ pwd
/home/leninware/LRVIII
leninware@leninware-VirtualBox:~/LRVIII$ ls
about about~ main main.cpp
leninware@leninware-VirtualBox:~/LRVIII$ cat about
**********
ФИО:Чурилов Сергей Эдуардович
Группа:М8О-103Б-20
E-mail: churilov.ser1204@gmail.com
ЛР: VIII
Номер по списку: 29
**********
leninware@leninware-VirtualBox:~/LRVIII$ cat main.cpp
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
struct List;

typedef List* Link_List;
typedef char* str;

struct List{
	Link_List next;
	str body;
};

str Alphabet =
(str)"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789\0";
Link_List First, Last, Barrier;
int Lenght;

void add_first(str);
void add_back(str);
void remove_element(str);
void remove_element(int);
void print_lenght();
void print_list();
bool str_cmp(str, str);
void print_menu();
void insert_element(str, str);
void erase_list();
void generate_list(int);
str get_random_str(int);

int main()
{
	Barrier = new List;
	Barrier->body = NULL;
	Barrier->next = NULL;
	First = Last = Barrier;

	int cmd, id, len;
	str line = new char[20], line2 = new char[20];
	print_menu();
	for(;;)
	{
		printf("\nEnter the command: ");
```

```
		scanf("%d", &cmd);
		switch(cmd)
		{
			case 0:
				return 0;
			case 1:
				printf("Enter a new item: ");
				scanf("%s", line);
				add_first(line);
				line = new char[20];
				break;
			case 2:
				printf("Enter a new item: ");
				scanf("%s", line);
				add_back(line);
				line = new char[20];
				break;
			case 3:
				printf("Enter the element after which you want to insert: ");
				scanf("%s", line);
				printf("Enter a new item: ");
				scanf("%s", line2);
				insert_element(line, line2);
				line = new char[20];
				break;
			case 4:
				printf("Enter the item you want to delete: ");
				scanf("%s", line);
				remove_element(line);
				break;
			case 5:
				move();
				break;
			case 6:
				erase_list();
				break;
			case 7:
				print_lenght();
				break;
			case 8:
				print_list();
				break;
			case 9:
				print_menu();
				break;
			case 10:
				printf("Enter the length of the new list: ");
				scanf("%d", &len);
				generate_list(len);
				break;
			default:
				printf("Unknown command!\n");
				break;
		}
	}
}

str get_random_str(int lenght)
{
	str s = new char [lenght+1];
	for(int i = 0; i < lenght; i++)
	{
		//srand(time(NULL));
```

```
		s[i] = Alphabet[rand() % 62];
	}
	s[lenght] = '\0';
	return s;
}

void generate_list(int n)
{
	erase_list();
	for(int i = 0; i < n; i++)
	{
		add_first(get_random_str(rand()%10 + 3));
	}
}

void add_first(str element)
{
	Link_List tmp = First;
	First = new List;
	First->body = element;
	First->next = tmp;
	if(Last == Barrier)
		Last = First;
	Lenght++;
}

void add_back(str element)
{
	if(Last == Barrier)
	{
		add_first(element);
		return;
	}
	Link_List tmp = new List;
	Last->next = tmp;
	tmp->body = element;
	tmp->next = Barrier;
	Last = tmp;
	Lenght++;
}

void remove_element(str element)
{
	if(First == Barrier)
	{
		printf("List is empty!");
		return;
	}
	Link_List prev = NULL, cur = First;
	while(cur != Barrier)
	{
		if(str_cmp(cur->body, element))
		{
			if(!prev)
				First = cur->next;
			else
				prev->next = cur->next;
			delete cur;
			Lenght--;
			return;
		}
		prev = cur;
		cur = cur->next;
```

```
        }
        printf("\nItem with this value not exists!\n");
}

void move()
{
        if (Lenght<=1)
                return;
        Link_List tmp = First;
        Link_List mid = First;
        for(int i = 1; tmp->next != Barrier; i++)
        {
        tmp = tmp->next;
        if (i==Lenght/2-1)
                mid = tmp;
        }

        tmp->next = First;
        First = mid->next;
        mid->next = Barrier;
}

void erase_list()
{
        Link_List tmp, cur = First;
        while(cur != Barrier)
        {
                tmp = cur->next;
                delete cur;
                cur = tmp;
        }
        First = Last = Barrier;
        Lenght = 0;
}

void insert_element(str old, str frash)
{
        if(First == Barrier)
        {
                printf("List is empty!");
                return;
        }
        Link_List cur = First;
        while(cur != Barrier)
        {
        if(str_cmp(cur->body, old))
        {
                Link_List f = new List;
                f->body = frash;
                f->next = cur->next;
                cur->next = f;
                Lenght++;
                return;
        }
        cur = cur->next;
        }
        printf("\nItem with this value not exists!\n");
}

void print_lenght()
{
        printf("List lenght is %d\n", Lenght);
}
```

```
bool str_cmp(str s1, str s2)
{
	int i;
	for(i = 0; s1[i] != '\0' || s2[i] != '\0'; i++)
		if(s1[i] != s2[i])
			return false;
	return s1[i] == s2[i];
}

void print_list()
{
	if(First == Barrier)
	{
		printf("List is empty\n");
		return;
	}
	printf("List:\n");
	Link_List tmp = First;
	for(int i = 1; tmp != Barrier; i++)
	{
		printf("(%d) %s ", i, tmp->body);
		tmp = tmp->next;
	}
	printf("\n");
}

void print_menu()
{
	printf("available commands:\n");
	printf("0 --- exit programm\n");
	printf("1 --- Add an item to the top of the list.\n");
	printf("2 --- Add an item to the end of the list\n");
	printf("3 --- Insert an item in the list after another one\n");

	printf("4 --- Remove an item from the list by value\n");
	printf("5 --- Rearrange the first and second half of the list\n");
	printf("6 --- Erase list\n");
	printf("7 --- Print the list length\n");
	printf("8 --- Print list\n");
	printf("9 --- Print this menu\n");
	printf("10 --- Generation list by length\n");
	printf("\n");
}

leninware@leninware-VirtualBox:~/LRVIII$ g++ main.cpp -o main
leninware@leninware-VirtualBox:~/LRVIII$ ./main
available commands:
0 --- exit programm
1 --- Add an item to the top of the list.
2 --- Add an item to the end of the list
3 --- Insert an item in the list after another one
4 --- Remove an item from the list by value
5 --- Rearrange the first and second half of the list
6 --- Erase list
7 --- Print the list length
8 --- Print list
9 --- Print this menu
10 --- Generation list by length

Enter the command: 10
Enter the length of the new list: 7
```

```
Enter the command: 8
List:
(1) i5Ug1YHC3UAV (2) w397 (3) sb1fE (4) Tt9rW (5) 9k88EmLgN7cC (6) R18N2 (7) kDHTxm

Enter the command: 5
List:
(1) 9k88EmLgN7cC (2) R18N2 (3) kDHTxm (4) Tt9rW (5) i5Ug1YHC3UAV (6) w397 (7) sb1fE

Enter the command: 5
List:
(1) i5Ug1YHC3UAV (2) w397 (3) sb1fE (4) Tt9rW (5) 9k88EmLgN7cC (6) R18N2 (7) kDHTxm

Enter the command: 4
Enter the item you want to delete: 9k88EmLgN7cC

Enter the command: 8
List:
(1) w397 (2) sb1fE (3) Tt9rW (4) R18N2 (5) kDHTxm

Enter the command: 1
Enter a new item: tyyw

Enter the command: 3
Enter the element after which you want to insert: sb1fE
Enter a new item: retre

Enter the command: 8
List:
(1) tyyw (2) w397 (3) sb1fE (4) retre (5) Tt9rW (6) R18N2 (7) kDHTxm

Enter the command: 7
List lenght is 7

Enter the command: 9
available commands:
0 --- exit programm
1 --- Add an item to the top of the list.
2 --- Add an item to the end of the list
3 --- Insert an item in the list after another one
4 --- Remove an item from the list by value
5 --- Rearrange the first and second half of the list
6 --- Erase list
7 --- Print the list length
8 --- Print list
9 --- Print this menu
10 --- Generation list by length

Enter the command: 2
Enter a new item: qscfd

Enter the command: 8
List:
(1) tyyw (2) w397 (3) sb1fE (4) 123 (5) Tt9rW (6) R18N2 (7) kDHTxm (8)qscfd

Enter the command: 5
List:
(1) Tt9rW (2) R18N2 (3) kDHTxm (4)qscfd (5) tyyw (6) w397 (7) sb1fE (8) 123

Enter the command: 0
leninware@leninware-VirtualBox:~/LRVIII$
```

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 дом.	Дата	Время	Событие	Действие по исправлению	Примечание
1	дом	30.04.2021	23:00	при добавлении элемента изменяются все остальные	необходимо после передачи в функцию добавления элемента создать новый элемент	

10 Замечание автора по существуработы _____

11 Выводы Благодаря выполнению данной лабораторной работы я узнал новую структуру данных и научился ее реализовывать

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

Подпись студента _____