Национальный исследовательский университет «Московский авиационный институт»
Факультет №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

КУРСОВОЙ ПРОЕКТ

ПО КУРСУ “ПРАКТИКУМ НА ЭВМ”
1 СЕМЕСТР ЗАДАНИЕ №4
“ЛИНЕЙНЫЕ СПИСКИ”

Выполнил студент	Мохляков Павел Александрович
Группа	М80-108Б-19
Преподаватель:	Поповкин Александр Викторович
Дата	
Оценка	

Москва
2020

СОДЕРЖАНИЕ

ЗАДАНИЕ

Состаить и отладить программу на языке Си для обработки линейного списка заданной организации с отображением списка на динамиеские структуры. Навигацию по списку следует реализовать с применением итераторов. Предусмотреть выполнени одного нестандартного и четырех стандартных действий:

1. Печать списка.
2. Вставка нового элимента в список.
3. Удаление элимента из списка.
4. Подсчет длины списка.

ВАРИАНТ 16

Тип элемента списка:целый. Вид списка: линейный однонаправленный с барьерным элиментом. Нестандартное действие: удалить из списка все элименты, предшествующие и последующие заданному значению.

ОСНОВНОЙ МЕТОД РЕШЕНИЯ

Программа создает пустой указатель на список, и входит в бесконечный список интерфейса, выход из которого сделан условием. Далее можно выполнять лубое из действий указанное в задании.

При добавлении нового элимента, если указатель на список пуст, то он создается, далее введенный элимент помещается на указанную позицию, если позиция больше длины списка, то номер позиции меняется на номер на единицу больший предъидущего.

Также есть удаление списка по номеру. В таком случае программа доходит по списку до нужного элимента, меняет указатели в обход него и удаляет его. Еще есть удаление по значению, тогда если элмент с таким значением существует, то удаляется первый встретившийся элимент с данным значением.

ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

Таблица А.1 - Общие сведение о программе

Аппаратное обеспечение	Ноутбук на базу Intel Core i5
Операционная система	Manjaro 5.4.36
Язык и система программирования	GNU C
Число строк	220
Компиляция программы в терминале	Zsh 5.8

ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

Программы предназначены для записи, чтения и поиска в структуре списка на языке Си. Программа поиска работает с временной сложностью алгоритма x.

ОПИСАНИЕ ПРОГРАММЫ

АЛГОРИТМЫ РАБОТЫ

1. Подключаем необходимые библиотеки
2. Создаем служебные функции
3. Создаем структуру данных
4. Ввод, вывод и удаление данных

ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Таблица А.2 - Функции файла main.c

Название	Аргументы и их тип	Описание функции
int main()		Создает указатель на список, создает интерфейс и вызывает вункции

Таблица А.3 - Функции файла list.c

Название	Аргументы и их тип	Описание функции
void Print_list()	struct list *top	Выводит список
struct list* Create_list()	struct list *top	Создает список
int Size_list()	struct list *top	Возвращает длину списка
struct list* Push_list()	struct list *top,int data,int pos	Добавляет элимент на заданную позицию в списке
struct list* Delete_list_pos()	struct list *top,int pos	Удаляет элимент в заданной позиции
int Find_list_data()	struct list *top,int data	Возвращает позицию элимента в заданным значением
struct list* Delete_list_data()	struct list *top,int data	Удаляет первый элимент в введенным значением
struct list* Clear_list()	struct list *top	Очищает список

ИСПОЛЬЗУЕМЫЕ ПЕРЕМЕНЫЕ

Таблица А.4 - Общие переменные

Имя переменной	Начальное значение	Тип	Назначение
top		struct list*	Указатель на вершину списка
data		int	Данные
pos		int	Положение в списке
size		int	Длина списка

Таблица А.5 - Переменные main() main.c

Имя переменной	Начальное значение	Тип	Назначение
k		int	Переменная выбора

ПРОТОКОЛ

```
[redacted] ~/Programs/C/kp8    cat list.c
#include <stdio.h>
#include <stdlib.h>
#include "list.h"

void Print_list(struct list *top)
{
 if(Size_list(top)==0) printf("~");
 while(top->br!=1)
 {
   printf("%d\t",top->data);
   top=top->next;
 }
 printf("\n");
}

struct list* Create_list(struct list *top)
{
 if(top==NULL)
 {
   top=malloc(sizeof(struct list));
   top->data=0;
   top->br=1;
   top->next=NULL;
 }
 return top;
}


int Size_list(struct list *top)
{
 int i=0;
 while(top->br!=1)
 {
   i++;
   top=top->next;
 }
 return i;
}

struct list* Push_list(struct list *top,int data,int pos)
{
 if(top==NULL) top=Create_list(top);
 if(pos>Size_list(top))
 {
   printf("Позиция больше длины списка, элимент записан последним.\n");

   struct list *q;
   q=malloc(sizeof(struct list));
   q->br=0;
   q->data=data;

   if(Size_list(top)==0)
   {
     q->next=top;
     top=q;
   }
   else
   {
     struct list *m=top;
     for(int i=1;i<Size_list(top);i++) m=m->next;
     q->next=m->next;
     m->next=q;
   }
 }
 else
 {
   struct list *q;
   q=malloc(sizeof(struct list));
   q->br=0;
   q->data=data;
   if(pos==1)
   {
     q->next=top;
```

```
    top=q;
   }
   else
   {
    struct list *m=top;
    for(int i=2;i<pos;i++) m=m->next;
    q->next=m->next;
    m->next=q;
   }
 }
 return top;
}

struct list* Delete_list_pos(struct list *top,int pos)
{
 if(top!=NULL)
 {
   struct list *q=top;
   if(pos>Size_list(top)) pos=Size_list(top);
   if(pos!=0)
   {
    if(pos==1)
    {
     top=top->next;
     free(q);
    }
    else
    {
     for(int i=2;i<pos;i++) q=q->next;
     struct list *m=q->next;
     q->next=m->next;
     free(m);
    }
   }
 }
 return top;
}

int Find_list_data(struct list *top,int data)
{
 int pos=0;
 if(top!=NULL)
 {
   while(top->br!=1)
   {
    pos++;
    if(top->data==data) break;
    top=top->next;
   }
   if(top->data!=data) pos=0;
 }
 return pos;
}

struct list* Delete_list_data(struct list *top,int data)
{
 if(top!=NULL)
 {
   int pos=Find_list_data(top,data);
   top=Delete_list_pos(top,pos);
 }
 return top;
}

struct list* Clear_list(struct list *top)
{
 if(top!=NULL)
 {
   while(Size_list(top)!=0)
   {
    top=Delete_list_pos(top,Size_list(top));
   }
 }
 return top;
}
[redacted] ~/Programs/C/kp8    cat list.h
#ifndef _LIST_
#define _LIST_

struct list{
 int data;
```

```
 int br;
 struct list *next;
};

void Print_list(struct list *top);
struct list* Create_list(struct list *top);
int Size_list(struct list *top);
struct list* Push_list(struct list *top,int data,int pos);
struct list* Delete_list_pos(struct list *top,int pos);
int Find_list_data(struct list *top,int data);
struct list* Delete_list_data(struct list *top,int data);
struct list* Clear_list(struct list *top);

#endif
[redacted] ~/Programs/C/kp8    cat main.c
#include <stdio.h>
#include "list.h"

int main()
{
 struct list *l=NULL;
 while(1)
 {
   int k;
   printf("Выберете действие:\n1-Добавить элимент в список\n");
   printf("2-Вывести список\n3-Удалить элимент из списка по позиции\n");
   printf("4-Удалить элимент из списка по значению\n");
   printf("5-Вывести длину списка\n");
   printf("6-Очистить список если в нем присудствует введенный элимент\n");
   printf("7-Выход\n");
   printf("Выбор: ");
   scanf("%d",&k);
   if(k==7)break;
   else
   {
     int d;
     int p;
     switch (k) {
       case 1:
         printf("Введите значение элимента: ");
         scanf("%d",&d);
         printf("Введите позицию элимента: ");
         scanf("%d",&p);
         l=Push_list(l,d,p);
         break;
       case 2:
         Print_list(l);
         break;
       case 3:
         printf("Введите позицию элимента: ");
         scanf("%d",&p);
         l=Delete_list_pos(l,p);
         break;
       case 4:
         printf("Введите значение элимента: ");
         scanf("%d",&d);
         l=Delete_list_data(l,d);
         break;
       case 5:
         printf("Длина списка: %d\n",Size_list(l));
         break;
       case 6:
         printf("Введите значение элимента: ");
         scanf("%d",&d);
         p=Find_list_data(l,d);
         if(p!=0) l=Clear_list(l);
         break;
       default:
         break;
     }
   }
   printf("\n");
 }
 return 0;
}
[redacted] ~/Programs/C/kp8    cat Makefile
CC=gcc

CFLAGS= -c -Wall

all: kp8
```

```
kp8: main.o matrix.o
	$(CC) main.o list.o -o prog

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

matrix.o: list.c
	$(CC) $(CFLAGS) list.c

clean:
	rm -rf *.o prog
```

```
[redacted] ~/Programs/C/kp8  ./prog
Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 1
Введите значение элимента: 12
Введите позицию элимента: 1
Позиция больше длины списка, элимент записан последним.

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 1
Введите значение элимента: 1
Введите позицию элимента: 1

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 1
Введите значение элимента: 3
Введите позицию элимента: 12
Позиция больше длины списка, элимент записан последним.

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 1
Введите значение элимента: 4
Введите позицию элимента: 4
Позиция больше длины списка, элимент записан последним.

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 1
Введите значение элимента: 6
Введите позицию элимента: 3

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
```

5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 2
1 12 6 3 4

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 3
Введите позицию элимента: 4

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 2
1 12 6 4

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 4
Введите значение элимента: 6

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 4
Введите значение элимента: 7

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 2
1 12 4

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 5
Длина списка: 3

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 6

```
Введите значение элимента: 1

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 2
~

Выберете действие:
1-Добавить элимент в список
2-Вывести список
3-Удалить элимент из списка по позиции
4-Удалить элимент из списка по значению
5-Вывести длину списка
6-Очистить список если в нем присудствует введенный элимент
7-Выход
Выбор: 7
[redacted] ~/Programs/C/kp8
```

ЗАКЛЮЧЕНИЕ

В данной работе я изучил работу со списками, создание структуры и функций для работы с ней, и реализовал знания на практике.

Эффективность программы можно увеличить сменив тип структуры данных, например на двухсторонний список с указателем на конец списка, что в ряде случаем может ускорить программу вдвое. Так как мой тип структуры указан в задании я этого не делал.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. РосДиплом, Оформление таблиц в дипломной работе, особенности и требования ГОСТ / Электронный диплом / Режим доступа: https://www.rosdiplom.ru/rd/pubdiplom/view.aspx?id=288
2. Диплом Журнал, Оформление курсовой работы по ГОСТу 2019(образец) / Электронный диплом / Режим доступа: https://journal.duplom.ru/kursovaya/oformlenie-kursov..
3. Vyuchit.work – универсальная методичка / Электронный диплом / Режим доступа: https://vyuchit.work/samorazvitie/sekretyi/oformlenie..
4. Архив вопросов и ответов для программистов / Электронный диплом / Режим доступа: https://qarchive.ru/320864_parametry_gcc__lm__lz__lrt..
5. Компилятор GCC / Электронный диплом / Режим доступа: http://parallel.uran.ru/book/export/html/25
6. Керниган, Брайан У., Ритчи, Деннис М. Язык программирования С, 2-е издание. :Пер. с англ. – М. : Издательский дом «Вильямс», 2009. – 304 с. : ил. –