**ЛР7**- 2-й сем

**Структура данных «список».**

**Цель работы:** «Изучить структуру данных «список». Рассмотреть внутреннюю структуру, виды списков, основные операции над списками. Разобрать приведенные примеры и выполнить самостоятельно один или более вариантов заданий».

**Теоретические сведения.**

**1. Основные определения.**

**Список** – это совокупность объектов или элементов, в котором каждый объект содержит информацию о местоположении связанного с ним объекта.

Если список располагается в оперативной памяти, то, как правило, информация для поиска следующего объекта – это указатель, адрес памяти. Если связный список хранится на диске в файле, то информация о следующем элементе может включать смещение элемента от начала файла к положению указателя записи или считывания файла, ключ записи и любую другую информацию, позволяющую однозначно отыскать следующий элемент списка.

В списке элементы связаны друг с другом логически. Логический порядок следования элементов списка опре­деляется с помощью указателей. Подчеркнем, что логи­ческий порядок следования элементов списка может не совпадать с физическим порядком их расположения в памяти ЭВМ.

Списки бывают линейными и кольцевыми, односвязными и двусвязными.

Как правило, элемент списка представляет собой структурную переменную, содержащую указатель или указатели на следующий элемент и любое число других полей, называемых информационными.

Если движение от элемента к элементу списка возможно только в одном направлении и список имеет начальную точку такого движения, говорят об односвязном списке. Элемент односвязного списка включает только указатель на следующий элемент. Сам список характеризуется указателем на начало списка (см. рис.12.1).

Двусвязный список позволяет выполнять «движение» от элемента к элементу в обоих направлениях. В этом случае элемент включает два указателя: на предыдущий и последующий элементы списка. А так как список имеет и начало, и конец, описываются еще два указателя – начала и конца списка (см. рис. 12.2).

Указатель на начало списка

NULL

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

данные

Указатель на второй элемент

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

данные

Указатель на второй элемент

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

данные

**…**

Рис. 12.1 Модель односвязного линейного списка

*Указатель на начало списка*

NULL

данные

*Указатель на предыдущий элемент*

Указатель на третий элемент

данные

*Указатель на первый элемент*

Указатель на второй элемент

данные

*NULL*

**…**

**…**

*Указатель на конец списка*

Рис. 12.2 Модель двусвязного линейного списка

Список, в котором последний элемент не связан с первым, называется линейным. Соответственно, кольцевым называется список, у которого последний элемент указывает на первый.

В последнем элементе односвязного и двусвязного линейного списка указатель на следующий элемент равен нулю. В первом элементе двусвязного списка указатель на предыдущий элемент равен нулю.

**2. Операции над списками.**

Основными операциями, которые выполняются над списками, являются перебор элементов списка, поиск заданного элемента, вставка в список нового элемента, удаление элемента из списка. Выполнение этих операций основано на изменении указателей.

2.1. Перебор элементов списка.

Эта операция выполняется для линейных списков очень часто и состоит в последовательном доступе к элементам списка – ко всем до конца списка либо до нахождения искомого элемента. Для каждого из перебираемых элементов осуществляется некоторая обработка его информационной части: сравнение с образцом, печать, модификация и прочее.

2.2. Вставка элемента в список.

Схематически выполнение этой операции представлено на рис. 12.3.

Мы видим, что указатель элемента эл2 теперь указывает не на элемент эл3, а на элемент эл5. Указатель элемента эл5 указывает на элемент эл3. Логическая последовательность элементов будет эл1, эл2, эл5, эл3, эл4. Подчеркнем еще раз, что в списке новый элемент эл5 физически не обязательно помещается за элементом эл2. Достаточно, чтобы он следовал за ним логически.

previous

Эл4

Эл3

Эл2

Эл1

current

Эл5

Рис. 12.3 Вставка элемента в середину односвязного списка

Вставка элемента в начало односвязного списка показана на рис. 12.4. В данном случае переустанавливается указатель на начало списка на вновь вставленный элемент Эл3, а элемент Эл3 указывает на бывший первый элемент списка.

head – указатель на начало

Эл2

Эл1

current

Эл3

Рис. 12.4 Вставка элемента в начало односвязного списка

2.3. Удаление элемента из списка.

При удалении элемента Эл3 из списка, прежде всего, выполняется поиск этого элемента в списке, а затем его удаление. Результат удаления показан на рис. 12.5.

Эл1

Эл2

Эл5

Эл3

Эл4

previous

delete

Рис. 12.5 Удаление элемента из списка

Теперь за элементом эл5 следует элемент эл4. Соответствующим образом изменился указатель элемента эл5. Элемент эл3 в списке теперь отсутствует, так как при просмотре списка, переходя от элемента к элементу в соответствии с указателями, мы на элемент эл3 не попадаем. Следует отметить, что удаление элемента из списка и удаление из памяти – это не одно и то же. Элемент Эл3 будет находиться в памяти до тех пор, пока мы не удалим его явно с помощью оператора delete.

Удаление элемента из начала списка показано на рис.12.6.

delete

head

Эл4

Эл3

Рис. 12.6 Удаление элемента из начала списка

**3. Пример реализации односвязного списка с помощью массива структур.**

В рассматриваемом примере используется следующая структура

struct list

{ char book;

list \*next; };

Предполагается, что память, отводимая под элемент списка, выделяется динамически, поэтому при реализации списков его длина ограничивается только доступным объёмом памяти. Информационное поле представляет собой символьную переменную по имени book. Признаком последующего элемента списка является наличие поля next, а признаком последнего элемента списка является равенство на NULL этого поля.

Следующая программа реализует операции:

* создает пустой список;
* добавляет элементы в список в алфавитном порядке;
* удаляет элементы из списка;
* проверяет, является ли список пустым;
* выводит список на экран.

Функция main организует запуск программы instructions для ввода кода необходимой операции над списком, и вызывает функцию, выполняющую эту операцию.

Операцию добавления элемента в список в алфавитном порядке выполняет функция insert, операцию удаления – функция del, операцию проверки, является ли список пустым – IsEmpty, вывод списка на экран – printList.

Пример 12.1

#include "stdafx.h"

# include <stdio.h>

# include <stdlib.h>

struct list

{

char book;

struct list \*next;

};

typedef struct list ListNode;

typedef ListNode \*ListNodePtr;

void insert (ListNodePtr\*, char);

char del (ListNodePtr\*, char);

int IsEmpty (ListNodePtr);

void printList (ListNodePtr);

void instructions (void);

int main ()

{

ListNodePtr start = NULL;

int choice;

char elem;

instructions(); // вывести меню

printf("?");

scanf("%d", &choice);

while (choice !=3)

{

switch (choice)

{

case 1: /\*Добавить значение в список\*/

printf ("Enter an character: ");

scanf("\n%c", &elem);

insert(&start, elem);

printList (start);

break;

case 2: /\*Удалить значение из списка\*/

if (!IsEmpty(start))

{

printf("Enter character to be deleted:");

scanf("\n%c", &elem);

if (del(&start,elem))

{ printf("%c deleted.\n", elem);

printList(start); }

else

printf("%c not found.\n", elem);

}

else

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

break;

default:

printf ("Invalid choice.\n");

instructions();

break;

}

printf("?");

scanf("%d", &choice);

}

printf("End of run.\n");

return 0;

}

/\*Вывести инструкции\*/

void instructions(void)

{

printf("Enter choice:\n"

"1 insert an element into the list.\n"

"2 delete an element from the list.\n"

"3 end program.\n");

}

/\*Вставить в список новое значение в алфавитном порядке \*/

void insert (ListNodePtr \*s, char value)

{

ListNodePtr newP, previous, current;

NewP = (ListNodePtr) malloc (sizeof(ListNode));

if (newP!=NULL)

{ /\*есть ли место ?\*/

newP->book =value;

newP->next=NULL;

previous=NULL;

current=\*s;

while (current!=NULL && value >current->book)

{ previous=current;

current=current->next; }

if(previous==NULL)

{ newP->next=\*s;

\*s=newP; }

else

{ previous->next = newP;

newP->next = current; }

}

else

printf("%c not inserted. No memory available.\n", value);

}

/\*Удалить элемент списка\*/

char del (ListNodePtr \*s, char value)

{

ListNodePtr previous, current, temp;

if (value == (\*s)->book)

{

temp=\*s;

\*s=(\*s)->next; // отсоединить узел

free(temp); // освободить отсоединенный узел

return value;

}

else

{ previous=\*s;

current=(\*s)->next;

while (current!=NULL && current->book!= value)

{

previous=current; /\*перейти.......\*/

current=current->next; /\*...к следующему\*/

}

if (current!=NULL)

{ temp=current;

previous->next=current->next;

free(temp);

return value; }

}

return '\0';

}

/\*Возвратить 1, если список пуст, 0 в противном случае\*/

int IsEmpty(ListNodePtr s)

{

return s==NULL;

}

/\*Распечатать список\*/

void printList (ListNodePtr current)

{

if (current==NULL)

printf("The list is empty.\n");

else

{ printf("The list is :\n");

while (current!=NULL)

{ printf("%c-->",current->book);

current=current->next; }

printf("NULL\n");

}

}

**Варианты индивидуальных заданий.**

***На 4:***

1. Создать список из введенных значений. Вывести его на экран.

***На 5-6:***

1. Создать список с помощью массива целых чисел. Элементы списка в обратном порядке вывести на экран.
2. Создать список с помощью массива целых чисел. Все четные элементы списка вывести на экран.
3. Создать список с помощью массива целых чисел. Все нечетные элементы списка вывести на экран.
4. Создать односвязный список с помощью массива целых чисел. Отсортировать элементы списка по возрастанию, задавая порядок чисел массивом индексов следующих элементов (next). В результате массив чисел остается без изменений, массив индексов переупорядочивается. Результирующий список вывести на экран.
5. Создать односвязный список с помощью массива целых чисел. Расположить в начале списка все четные элементы списка, задавая порядок чисел массивом индексов следующих элементов (next). В результате массив чисел остается без изменений, массив индексов переупорядочивается. Результирующий список вывести на экран.
6. Создать односвязный список с помощью массива целых чисел. Исключить из списка все нулевые элементы, задавая порядок чисел массивом индексов следующих элементов (next). В результате массив чисел остается без изменений, массив индексов переупорядочивается. Результирующий список вывести на экран.
7. Создать односвязный список с помощью массива целых чисел. Исключить из списка все нулевые элементы, задавая порядок чисел массивом индексов следующих элементов (next). В результате массив чисел остается без изменений, массив индексов переупорядочивается. Найти сумму все четных элементов списка. Результирующий список и сумму вывести на экран.
8. Создать односвязный список с помощью массива целых чисел. Ввести с клавиатуры число и поместить его за пятым элементом списка, задавая порядок чисел массивом индексов следующих элементов (next). Результирующий список вывести на экран.
9. Создать односвязный список с помощью массива целых чисел. Ввести с клавиатуры число и поместить его перед тем элементом списка, который больше него. Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
10. Создать односвязный список с помощью массива целых чисел. Ввести с клавиатуры число, найти это число в списке и удалить. Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
11. Создать односвязный список с помощью массива целых чисел. Ввести с клавиатуры число, найти все элементы в этим числом в списке и удалить. Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
12. Создать односвязный список с помощью массива целых чисел. Поменять местами четные и нечетные элементы списка (рядом стоящие). Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
13. Создать односвязный список с помощью массива целых чисел. Сформировать новый список, в котором элементы следуют от конца к началу (последний элемент станет первым, предпоследний – вторым и т.д.). Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
14. Создать односвязный список с помощью массива целых чисел. Продублировать в списке первый, третий и пятый элементы. Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
15. Создать односвязный список с помощью массива целых чисел. Удалить в списке первый, третий и пятый элементы.. Результирующий список вывести на экран. Порядок чисел в списке задается массивом индексов следующих элементов (next).
16. Создать список с помощью массива структур. Элементы списка в обратном порядке вывести на экран.
17. Создать список с помощью массива структур. Все четные элементы списка вывести на экран.
18. Создать список с помощью массива структур. Все нечетные элементы списка вывести на экран.
19. Создать односвязный список с помощью массива структур. Отсортировать элементы списка по возрастанию. Результирующий список вывести на экран.
20. Создать односвязный список с помощью массива структур. Один из элементов структуры – целое число. Расположить в начале списка все элементы списка с четными целыми числами. Результирующий список вывести на экран.
21. Создать односвязный список с помощью массива структур. Исключить из списка все элементы с нулевым целым числом. Результирующий список вывести на экран.
22. Создать односвязный список с помощью массива структур. Исключить из списка все элементы с нулевым целым числом. Найти сумму тех целочисленных элементов списка, которые являются четными. Результирующий список и сумму вывести на экран.
23. Создать односвязный список с помощью массива структур. Создать новый элемент списка, ввести с клавиатуры число и поместить его в целочисленное поле нового элемента. Разместить новый элемент за пятым элементом списка. Результирующий список вывести на экран.
24. Создать односвязный список с помощью массива структур. Создать новый элемент списка, ввести с клавиатуры число и поместить его в целочисленное поле нового элемента. Разместить новый элемент перед тем элементом списка, целочисленное поле которого имеет большее значение. Результирующий список вывести на экран.
25. Создать односвязный список с помощью массива структур. Ввести с клавиатуры число, найти это число в списке (в целочисленном поле) и удалить соответствующий элемент списка. Результирующий список вывести на экран.
26. Создать односвязный список с помощью массива структур. Ввести с клавиатуры число, найти все элементы в этим числом (в целочисленном поле) в списке и удалить. Результирующий список вывести на экран.
27. Создать односвязный список с помощью массива структур. Поменять местами четные и нечетные элементы списка (рядом стоящие). Результирующий список вывести на экран.
28. Создать односвязный список с помощью массива структур. Сформировать новый список, в котором элементы следуют от конца к началу (последний элемент станет первым, предпоследний – вторым и т.д.).). Результирующий список вывести на экран.
29. Создать односвязный список с помощью массива структур. Продублировать в списке первый, третий и пятый элементы. Результирующий список вывести на экран.
30. Создать односвязный список с помощью массива структур. Удалить в списке первый, третий и пятый элементы.. Результирующий список вывести на экран.

**На 7-9:**

1. Написать программу, содержащую процедуру, которая меняет местами первый и второй элементы непустого списка. Если элементы не найдены, то выдать на экран соответствующие сообщение.
2. Написать программу, содержащую процедуру, которая меняет местами первый и пятый элементы непустого списка. Если элементы не найдены, то выдать на экран соответствующие сообщение.
3. Написать программу, содержащую процедуру, которая меняет местами первый и последний элементы непустого списка. Если элементы не найдены, то выдать на экран соответствующие сообщение.
4. Написать программу, содержащую процедуру, которая вставляет новый элемент перед каждым вхождением заданного элемента. Если элементы не найдены, то выдать на экран соответствующие сообщение.
5. Написать программу, содержащую процедуру, которая вставляет новый элемент за каждым вхождением заданного элемента. Если элементы не найдены, то выдать на экран соответствующие сообщение.
6. Написать программу, содержащую подпрограмму, которая проверяет на равенство списки М1 и М2.
7. Написать программу, содержащую функцию, которая определяет, входит ли список М1 в список М2. Предполагается, что списки существуют.
8. Написать программу, содержащую подпрограмму, которая копирует в конец непустого списка М его первый элемент. Если элементы не найдены, то выдать на экран соответствующие сообщение.
9. Написать программу, содержащую подпрограмму, которая копирует в начало непустого списка М его последний элемент. Если элементы не найдены, то выдать на экран соответствующие сообщение.
10. Написать программу, содержащую процедуру, которая копирует в список М за каждым вхождением заданного элемента все элемента списка М1.
11. Написать программу, содержащую процедуру, которая объединяет два упорядоченных по не убыванию списка М1 и М2 в один упорядоченный по не убыванию список, построив новый список М.
12. Написать программу, содержащую процедуру, которая объединяет два упорядоченных по не убыванию списка М1 и М2 в один упорядоченный по не убыванию список, сменив соответствующим образом ссылки в М1 и М2.
13. Написать программу, содержащую функцию, которая проверяет, упорядочены ли элементы списка по алфавиту.
14. Напишите программу сортировки существующего списка по алфавиту. В программе используйте подпрограммы.
15. Напишите программу, которая создавала бы файл целых чисел, а затем формировала список целых чисел файла. Создайте в конце списка элемент, содержащий сумму всех чисел файла. В программе используйте подпрограммы.
16. Напишите программу, которая создавала бы файл целых чисел, а затем формировала список целых чисел файла. Создайте список чисел, являющихся суммой соседних элементов. В программе используйте подпрограммы.
17. Напишите программу, которая создавала бы текстовый файл, а затем формировала список строк файла. Создайте список обратных строк. В программе используйте подпрограммы.
18. Напишите программу, которая создавала бы текстовый файл, а затем формировала список строк файла. Создайте отсортированный список строк. В программе используйте подпрограммы.
19. Напишите программу, которая создавала бы файл комбинированного типа, а затем формировала список, используя какое-либо поле записи. Создайте отсортированный список. В программе используйте подпрограммы.
20. Напишите программу, которая создавала бы файл комбинированного типа, а затем формировала список элементов файла. Создайте отсортированный по какому-либо полю список. В программе используйте подпрограммы.
21. Составить программу, которая вводит с клавиатуры названия городов, динамически отводит место в памяти под каждое название и строит из них связанный список, упорядоченный по алфавиту. По окончании формирования список городов вывести на экран монитора.
22. Составить список учебной группы, содержащий не менее 15 студентов. Указать для каждого студента оценки, полученные на последних четырех экзаменах. Разработать программу, которая вводит данные с клавиатуры о каждом студенте, строит односвязный список, а затем удаляет из списка элементы, относящиеся к неуспевающим студентам.
23. Информацию о величине экспорта и соответствующий номер контракта записать в двухсвязный кольцевой список. Затем переместить данную информацию в двумерный динамический массив и осуществить поиск максимальной величины экспорта. На экран вывести искомую информацию.
24. Создать двухсвязный список, содержащий следующую информацию: год и соответствующую численность населения. Программу организовать таким образом, чтобы на экран выводилась информация, численность населения в которой была больше введенного с клавиатуры значения.
25. Ввести с клавиатуры строку символов, формируя из ее элементов двусвязный список. Написать программу, которая формирует двусвязный список из входной строки. В поле данных каждого элемента списка записывается отдельный символ. В программе производится анализ первого символа входной строки: если это буква ***'А'***, то в конец списка добавляется еще одна буква ***'А'***, иначе из списка исключаются все буквы ***'А'***. Полученный результат выводится на экран.