Лабораторная работа №3.
Часть 1. *Разработка подпрограмм с параметром-массивом*

Целью данной работы является закрепление навыков работы с динамическими массивами и создания подпрограмм. В задачах этого раздела требуется логически независимые или повторяющиеся последовательности действий оформить в виде отдельных подпрограмм, обращение к которым происходить из основной программы. При разработке программы составить алгоритм по отдельности, как для процедуры, так и для основной части программы. Формальными параметрами процедуры является два массива $B_1,B_2,...,B_n$; $C_1,C_2,...,C_m$, либо один из них, или матрица A из n строк и n столбцов. Составленную процедуру использовать в основной части программы для обработки конкретных (фактических) массивов или матрицы.

Реализовать программу на языке C++.

Номер решаемой задачи соответствует номеру варианта.

Пример решения задачи

Массив $P_1,P_2,...,P_n$; P_i(i=1,2,...,n) формируется по правилу: P_i=1, если в i-й строке матрицы A нет элементов, которые меньше полусуммы максимального и минимального элементов матрицы, иначе P_i=0.

Программа на языке Си.

```
#include<stdio.h>
#include<stdlib.h>
void vvod(int **, int, int, char);
int kolzam(int **, int, int);

void vvod(int **matr, int n, int m, char namematr)
{       int i, j;
        for(i=0; i<n; i++)
                for(j=0; j<m; j++)
                {       printf("\n %c[%d][%d]=",namematr,i+1,j+1);
                        scanf("%d", &matr[i][j]);
                }
}

void  f2(int **matr, int *mas, int n)
{       int  i, j, kol, min, max;
        float  r;
        min= matr[0][0];
        max= matr[0][0];
        for(i=0; i<n; i++)
                for(j=1; j<n; j++)
                {       if (matr[i][j] > max)
```

```
                max= matr[i][j];
            if (matr[i][j] < min)
                min= matr[i][j];
        }
    r= (float)(max + min)/2.;
    for(i=0; i<n; i++)
    {   j= 0;
        kol= 0;
        while ((kol ==0) && (j < n))
        {   if (matr[i][j] < r)
                kol= 1;
            j++;
        }
        if (kol == 1)
            mas[i]= 0;
        else
            mas[i]= 1;
    }
    return ;
}
void main()
{   int  **a,*p, n, i, j;
    printf("Введите размерность матрицы A: ");
    scanf("%d", &n);
    b=(int**)malloc(n*sizeof(int*));
    p=(int*)malloc(n*sizeof(int));
    for(i=0; i<n; i++)
        a[i]=(int*)malloc(n*sizeof(int));
    vvod(a, n, n, 'A');
    f2(a, p, n);
    for(i=0; i<n; i++)
        printf("\np[%d]= %d", i+1, p[i]);
    for(i=0; i<n1; i++)
        free(a[i]);
    free(a);
    free(p);
    return;
}
```

Задание на программирование

1. Коэффициенты многочлена, являющегося суммой многочленов $B_1X^n+B_2X^{n-1}+...+B_nX$; $C_1X^m+C_2X^{m-1}+...+C_mX$.

2. Массив $P_1,P_2,...,P_m$, получаемый из $C_1,C_2,..,C_m$ по правилу: заменить на 0

все элементы до первого отрицательного, который заменить на 1, а все остальные элементы заменить их модулями.

3. Множество значений, которые имеются и в массиве B_1,B_2, ..., B_n, и в массиве $C_1,C_2,...,C_m$.

4. Номера всех элементов массива $B_1,B_2,...,B_n$, которые имеют наибольшее значение (оно может повторяться).

5. Массив $D_1,D_2,...,D_m$ такой, что $D_1=C_m$, $D_2=C_{m-1}$, ..., $D_m=C_1$.

6. Коэффициенты 1-й производной многочлена $C_1X^m + C_2X^{m-1} + ... + C_mX$.

7. Значения трех наибольших элементов среди $C_1,C_2,...,C_m$.

8. Наибольший среди отрицательных элементов $B_1,B_2,...,B_n$ и среднее арифметическое всех элементов.

9. Расстояния между 1-ой точкой двумерного пространства и всеми прочими точками, каждая из которых задана парой координат $(B_i;C_i)$, i=1,2,...,n; n=m.

10. Массив, полученный перемещением отрицательных элементов массива $C_1,C_2,...,C_m$ в его начало, а остальных - в его конец.

11. Массив $P_1,P_2,...,P_m$, полученный из массива $C_1,C_2,...,C_m$ по правилу $P_k \sum_{i=1}^{k} C_i$, k=1,2,...,m.

12. Множество всех значений, имеющихся в массивах $B_1,B_2,...,B_n$; $C_1,C_2,...,C_m$, без повторения значений.

13. Массив $P_1,P_2,...,P_m$, полученный заменой нулей в массиве $C_1,C_2,...,C_m$ полусуммой соседних элементов (прочие элементы не изменяются); если 0 стоит на первом или последнем месте, то он заменяется значением соседнего элемента.

14. Массив элементов, каждый из которых встречается в массиве $B_1,B_2,...,B_n$ не более 1-го раза.

15. Массив, каждый элемент которого равен наибольшему из двух элементов с таким же номером в исходных массивах (n=m).

16. Измененный массив $C_1,C_2,...,C_m$: каждый элемент, предшествующий

минимальному, помножен на него, а все следующие за минимальным уменьшены на 1.

17. Массив $P_1,P_2,...,P_n$; $P_i=\sum_{k=1}^{n} C_k\sqrt{B_i}$.

18. Номера элементов - локальных минимумов в массиве C_1, $C_2,...,C_m$ и их количество.

19. Значения наименьших элементов строк матрицы A.

20. Средние арифметические значения: --
элементов первого столбца матрицы A;
--совокупности элементов 1-го и 2-го столбцов матрицы A;
--совокупности элементов трех первых столбцов матрицы A.

21. Значение наибольшего элемента 1-й строки матрицы A, наименьшего элемента 2-й строки и среднее арифметическое всех элементов матрицы.

22. Массив $P_1,P_2,...,P_n$. P_i =1, если в i-й строке матрицы A положительных элементов больше, чем отрицательных, иначе P_i=0.

23. Суммы элементов, расположенных на главной диагонали матрицы A и всех нижележащих диагоналях (отдельная сумма для каждой из диагоналей), на главной диагонали номер строки равен номеру столбца.

24. Количество и координаты (номер строки и столбца) локальных минимумов матрицы A - элементов A_{ij}, удовлетворяющих одновременно следующим неравенствам: $A_{ij-1} > A_{ij} < A_{ij+1}$ и $A_{i-1j} > A_{ij} < A_{i+1j}$.

25. Массив $P_1,P_2,...,P_n$; P_i=0, если i-ый столбец матрицы A не содержит элементов, абсолютная величина которых больше 1, иначе P_i=1.

26. Значения наименьших элементов во всевозможных квадратах матрицы A, левый верхний угол которых совпадает с элементом A_{11}.

27. Новое содержание матрицы A, полученное замещением 1-й строки 2-й строкой, 2-й строки – 3-й строкой и т.д., в последней строке должно оказаться исходное содержание 1-й строки.

28. Множество элементов матрицы A - натуральных чисел, являющихся простыми числами (не делятся на меньшие натуральные, кроме 1).

29. Матрица D, каждый элемент которой D_{ij} (i=1,2,...,n); (j=1,2,...,m) равен

$$(\sum_{k=1}^{i} B_k) \cdot (\sum_{i=j}^{m} C_i).$$

Часть 2. *Динамические структуры данных*

Цель данной работы – получение навыков работы с динамическими структурами данных, способных увеличиваться и уменьшаться в размерах в процессе выполнения программы. Каждая структура данных взаимосвязью элементов, набором типовых операций над этой структурой. В разделе приведены задания на реализацию таких динамических структур, как: очередь, стек, кольцевой список, бинарное дерево.

Номер решаемой задачи соответствует номеру варианта, выданному преподавателем.

Общее задание: реализовать собственную динамическую структуру и используя ее выполнить задание на языке С++.

Пример решения задачи

Вводятся фамилии абитуриентов. С использованием бинарного дерева распечатать их в алфавитном порядке с указанием количества повторений каждой фамилии.

В программе будет использована рекурсивная функция *der()*, которая строит дерево фамилий, а также рекурсивная функция для печати дерева *print_der()*, в которой реализован первый способ обхода дерева.

Запишем программу в операторах языка Си.

```
#include<alloc.h>
#include<stdio.h>
#define TREE  struct  der
TREE
{       char *w;
        int c;
        TREE *1, *r;
};
TREE*  der(TREE *kr,   char *word)
{       int sr;
        if(kr == NULL)
        {       kr = (TREE  *)malloc(sizeof (TREE)) ;
                kr -> w = word;
                kr -> c = 1;
                kr -> l = kr -> r = NULL;
        }
        else
                if ((sr=strcmp(word, kr -> w))==0)
                        kr -> c++;
                else
                        if( sr<0 )
```

```
                        kr -> l = der(kr -> l, word);
                else
                        kr -> r = der(kr -> r, word);
        return    kr;
}

void   print_der(TREE  *kr)
{       if ( kr )
        {       print_der (kr -> l) ;
                printf ("слово - %-20s \tкол-во повтор.- %d\n",  kr -> w,  kr -> c);
                print_der(kr -> r);
        }
}

void    main()
{       int i;
        TREE  *kr;
        static   char  word[40][21];
        kr=NULL;
        i=0;
        puts("Введите <40 фамилий длинной <20 каждая");
        scanf("%s",  word[i]);
        while ( word[i][0]!='\0')
        {       kr = der(kr, word[i]);
                scanf("%s",  word[++i]);
        }
        print_der(kr);
}
```

Задание на программирование

1. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя список вычислить значение многочлена в целочисленной точке x.
2. Бинарное дерево. Реализовать проверку всех элементов динамической структуры и определить: для целочисленных элементов – число отрицательных.
3. Кольцевой список. Перегруппировать элементы так, чтобы они образовывали сектора: положительных, отрицательных и нулевых элементов, определить процентную долю каждого сектора.
4. Используя стек осуществить перевод числа, представленного строкой, из двоичной системы счисления в десятичную.
5. Используя стек осуществить перевод числа, представленного строкой, из двоичной системы счисления в шестнадцатеричную и обратно.
6. Очередь. Определить произведение элементов, расположенных между максимальным и минимальным элементами.
7. Использовать стек для отыскания пути выхода из лабиринта. Лабиринт задается матрицей из 1 и 0, 1 – есть проход, 0 – нет прохода. Путь определяется виде набора квадратов, квадрат имеет координаты ячеек матрицы.
8. Бинарное дерево. Реализовать считывание текста из файла, найти самое длинное слово и число его повторений.
9. Два бинарных дерева подобны, если оба пусты и если подобны их правые и левые поддеревья. Проверить являются ли два дерева подобными.
10. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя проверить на равенство два многочлена.
11. Используя стек распечатать слова текстового файла в обратном порядке.
12. Очередь. Реализовать игру «считалка», начав отсчет от первого, удалять каждый *k*-ый элемент (*k* задано). Выводить удаляемые элементы.
13. Используя стек, переписать построчно текстовый файл, а именно переписать только встречающиеся цифры в обратном порядке.
14. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя список сложить два многочлена (нулевые слагаемые исключить из результирующего списка).
15. Очередь. С использованием заданной структуры за один просмотр файла, содержащего целые числа, распечатать файл в следующем виде: сначала все числа меньшие A, а затем остальные.
16. Кольцевой список. Реализовать схему кольцевого маршрута и определять кратчайший путь между двумя заданными остановками.

17. В файле находится текст программы на языке C++. Написать препроцессор, используя стек, проверяющий правильность вложений циклов в этой программе.
18. Используя кольцевой список необходимо сложить два длинных целых числа, представленных виде строк.
19. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя список привести подобные члены и расположить их по убыванию степеней x.
20. Два бинарных дерева зеркально подобны, если оба пусты и если левое поддерево подобно правому, а правое левому. Проверить являются ли два дерева зеркально подобными.
21. Кольцевой список. Реализовать проверку всех элементов динамической структуры и определить: для целочисленных элементов – число нулевых элементов и разделить на него все положительные элементы.
22. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя список по многочлену построить многочлен его производной.
23. Вставлять элементы в очередь в порядке возрастания элементов, если такой элемент уже в очереди есть, не вставляя, сообщить об этом.
24. Кольцевой список. Реализовать игру «считалка», начав отсчет от первого (обход осуществлять слева на право), удалять каждый *k*-ый элемент (*k* задано). Выводить удаляемые элементы.
25. ,Двунаправленный список. Реализовать добавление заданного элемента после k, k+5, …, k+i*5, где (i=0,1,…n, n – задается). Вывести все элементы.
26. Используя стек, проверить, соблюден ли баланс скобок в тексте. Текст находится в файле.
27. Бинарное дерево. Реализовать русско-английский словарь с возможностью его заполнения из файла или с клавиатуры.
28. Кольцевой список. Графическое построение многогранника, вершины которого заданы декартовыми координатами.
29. Список. В узле списка хранятся коэффициенты многочлена (коэффициент при x, степень x). Используя проверить на равенство два многочлена.