Практическое занятие № 6 «Программирование алгоритмов поиска в двумерных массивах на языке С++»

Учебные цели:

- получение практических умений и навыков кодирования алгоритмов нахождения заданных элементов в двумерных массивах. Сформировать компетенции анализа и синтеза при решении задач циклической обработки последовательности значений.

Воспитательные цели:

- формировать диалектико-материалистическое мировоззрение;
- формировать навыки самостоятельности и дисциплинированности;
- стимулировать активную познавательную деятельность обучаемых, способствовать формированию у них творческого мышления.

Категория слушателей: 2,3 курс.

Время: 90 мин.

Место проведения: компьютерный класс.

Материально-техническое обеспечение:

1) персональный компьютер *IBM PC* с операционной системой Windows XP; 2) среда разработки приложений *Visual C++.NET.*

ПЛАН ПРАКТИЧЕСКОГО ЗАНЯТИЯ

Учебные вопросы	Время, мин
Вступительная часть	5
1. Поиск заданных элементов двумерного массива в С++ . .	15
2. Выполнение индивидуального задания	65
Заключительная часть	5

Элементы теории

Сортировка данных − это процесс изменения порядка расположения элементов в некоторых упорядоченных структурах данных таким образом, чтобы обеспечить возрастание или убывание числового значения элемента данных или определенного числового параметра, связанного с каждым элементом данных (ключа), при переходе от предыдущего элемента к последующему. То есть для любой пары чисел определены отношения «больше» или «меньше».

При разработке программы можно воспользоваться различными алгоритмами. Наиболее известными являются следующие:

- метод сортировки обменами («пузырьковая» сортировка);

- метод сортировки вставками;
- метод сортировки выбором элемента;
- метод разделения (алгоритм «быстрой» сортировки метод Хоора);
- метод «пирамиды» (метод Уильямса-Флойда);
- метод «счетчика».

Главным показателем качества алгоритма внутренней сортировки является скорость сортировки. Алгоритмы «пирамиды», «счетчика» и «быстрой» сортировки обеспечивают высокую скорость сортировки и находят широкое практическое применение. Недостатком «быстрой» сортировки является возможность резкого увеличения трудоемкости при «неблагоприятном» исходном порядке элементов в массиве. С другой стороны, метод «пирамиды» в целом отстает по скорости от «быстрой» сортировки. Метод «счетчика» является быстрым, однако не рационально использует оперативную память компьютера, поскольку требует введения дополнительного массива.

Выбор в пользу того или иного алгоритма может быть сделан при условии тщательного статистического анализа реальной задачи и это является достаточно важной проблемой в программировании.

Алгоритм сортировки обменами («пузырьковая» сортировка). Метод «пузырька» один из самых простых методов внутренней сортировки. Суть алгоритма состоит в последовательном просмотре массива от конца к началу или от начала к концу и сравнении каждой пары элементов между собой. При этом «неправильное» расположение элементов устраняется путем их перестановки. Процесс просмотра и сравнения элементов повторяется до просмотра всего массива. При сортировке по возрастанию «легкие» элементы с меньшим значением как бы «всплывают» к началу массива подобно тому, как это делают пузырьки воздуха в стакане с водой − отсюда и происходит популярное название алгоритма.

«Пузырьковая» сортировка имеет очень плохие временные характеристики. Она имеет только учебно-исторический интерес и не может быть рекомендована для практического использования.

Схема описанного алгоритма приведена на рис. 1. Обратите внимание на то, что для перестановки элементов (блок 4) используется буферная переменная *B*, в которой временно хранится значение элемента, подлежащего замене.

Приведём ниже программную реализацию данного алгоритма на языке C++.

```
/*функция пузырьковой сортировки элементов массива по
возрастанию*/
void sort_puzyr (int *A, int n, int *per, int *pr, int *sr)
{
int temp;
for (int j = 0; j<n-1; j++)
     {
          (*pr)++; // подсчет проходов исходного массива
```

```
            temp = 0;
            for (int i=0; i<n-j-1; i++)
            { (*sr)++; // подсчет сравнений элементов массива
                 if (A[i]>A[i+1])
                 {
                  temp = A[i+1];
                  A[i+1] = A[i];
                  A[i] = temp;
                  (*per)++; // подсчет перестановок
                 }
            }
      }
}
```

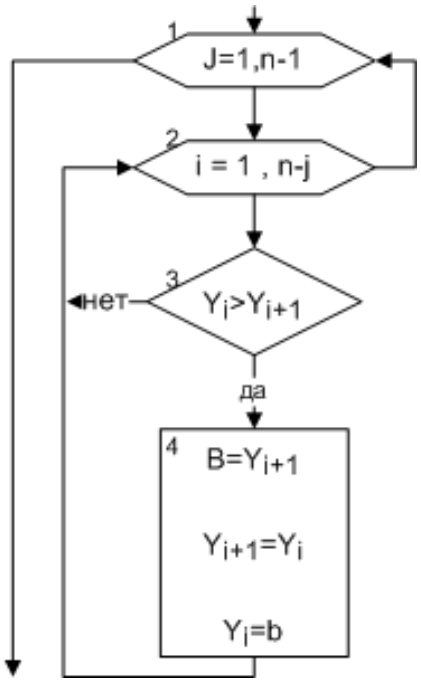

Рис. 1. Сортировка массива пузырьковым методом

Вопросы для самопроверки

1. Дайте определение массива.
2. Какими свойствами обладает массив
3. Многомерные массивы: инициализация и обращение к элементам массива.
4. Как определить количество элементов массива?
5. Напишите программу для ввода 10 элементов двумерного массива целочисленного типа.
6. Напишите программу для вывода 10 элементов двумерного массива целочисленного типа (массив инициализировать при объявлении).
7. Напишите программу для ввода элементов двумерного массива типа double размером 3*2.

8. Напишите программу для вывода элементов двумерного массива типа double размером 4*3.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОТРАБОТКЕ УЧЕБНЫХ ВОПРОСОВ

Практическое задание

Задание 1. Составить программу для следующих задач обработки двумерных массивов:

1. Дана матрица A(n,m). Найти максимум среди наименьших элементов столбцов.
2. Дана матрица A(n,m). Найти минимум среди максимальных элементов строк.
3. Дана матрица A(n,m). Найти порядковые номера первого отрицательного и последнего положительного элемента (если таковые имеются). Значение элементов и их порядковые номера вывести на экран или выдать соответствующее сообщение.
4. Дана матрица A(n,m). Найти строку с максимальной суммой элементов.
5. Дана матрица A(n,m). Найти столбец с минимальным произведением элементов.
6. Дана матрица A(n,m). Найти максимальное из чисел встречающееся более одного раза.
7. Дана матрица A(n,m). Найти минимальное из чисел встречающееся более одного раза.
8. Дана матрица A(n,m). Найти номер строки, в которой находится самая длинная серия одинаковых элементов.
9. Дана матрица A(n,m). Найти номер столбца, в котором находится самая длинная серия одинаковых элементов.
10. Дана матрица A(n,m). Сформировать одномерный массив B(n), в который записать номера столбцов минимальных элементов каждой строки.

Задание 2. Составить программу для следующих задач обработки двумерных массивов:

1. Дана матрица A(n,n). Найти максимум среди элементов, лежащих выше побочной диагонали.
2. Дана матрица A(n,n). Найти минимум среди элементов, лежащих ниже побочной диагонали.
3. Дана матрица A(n,n). Найти максимум среди элементов, лежащих выше главной диагонали.
4. Дана матрица A(n,n). Найти минимум среди элементов, лежащих ниже главной диагонали.
5. Дана матрица A(n,n). Найти минимум среди диагональных элементов.

6. Дана матрица A(n,n). Найти максимум среди элементов побочной диагонали.

7. Дана матрица A(n,n). Вычислить сумму элементов, лежащих выше и ниже главной диагонали матрицы, и ответить какая сумма больше.

8. Дана матрица A(n,n). Вычислить сумму тех столбцов матрицы, последний элемент которых равен элементу, стоящему на главной диагонали.

9. Дана матрица A(n,n). Вычислить произведение элементов, лежащих выше и ниже побочной диагонали матрицы, и ответить какая сумма больше.

10. Дана матрица A(n,n). Вычислить сумму элементов, лежащих на побочной и главной диагонали матрицы, и ответить какая сумма больше.

Листинг П. 2
Выбрать все строки матрицы (*n*x*m*), среднее арифметическое которых отрицательно.

```
// Laboratornay rabota N 11
/*Vypolnil stydent 11 uchebnoy gruppy
Ivanov P.S. */

#include "stdafx.h"
#include <iostream>

int main()
{
     using namespace std;
     int t,s,i,j,k,l,M[10][15],M_new[10][15];
     cout << "Vvedite kolichestvo strok massiva M k=";
     cin >> k;
     cout << "Vvedite kolichestvo stolbcov massiva M l=";
     cin >> l;
     for (i=0;i<=k-1;i++)
     {
    for (j=0;j<=l-1;j++)
         {cout << "Vvedite M["<<i+1<<","<<j+1<<"]="; // Vvod
dvumernogo
          cin >> M[i][j];            // massiva
         }
     }
     cout << endl;

     for (i=0;i<=k-1;i++)
     {
    for (j=0;j<=l-1;j++) // Vyvod dvumernogo massiva
         cout << M[i][j]<<"\t";
     cout << endl;
     }
     cout << endl;
     t=0;
     for (i=0;i<=k-1;i++)
     {
          s=0;
     for (j=0;j<=l-1;j++)
```

```
            s+=M[i][j];
            if (s<0)
            {
             ++t;
            for (j=0;j<=l-1;j++) M_new[t-1][j]=M[i][j];
            }
      }
      cout << "Obrabotanny massiv" << endl;

      if (t==0) cout << "Net massiva" << endl;
      else
            for (i=0;i<=t-1;i++)
            {
            for (j=0;j<=l-1;j++)     // Vyvod dvumernogo massiva
           cout << M_new[i][j]<<"\t";
      cout << endl;
            }
            return 0;
}
```

Контрольные вопросы

1. Напишите фрагмент программы поиска минимального элемента двумерного массива.
2. Напишите фрагмент программы поиска максимального элемента двумерного массива.
3. Как поменять местами два элемента массива.
4. Напишите фрагмент программы поиска количества положительных элементов двумерного массива.
5. Напишите фрагмент программы поиска места расположения элемента массива равного числу, введенному с клавиатуры.