Лабораторная работа №4
Динамические массивы

Пример 1. Напишите программу, которая запрашивает число **n** с консоли, создает одномерный динамический массив размера **n** и заполняет его единицами. Далее необходимо вывести на печать все элементы массива.

```
#include <stdio.h>
int main( ){
      int n;      //определяем переменную для хранения количества элементов массива
      printf("Enter n: \n "); //запрашиваем n
      scanf("%i",&n);         //считываем n с консоли
      int *m = new int[n];    //заводим динамический массив
      for(int i=0; i<n; i++)
            m[i]=1;           //присваиваем всем элементам массива значение 1
      for(int i=0;i<n;i++)
            printf("m[%i]=%i\n",i,m[i]);//выводим элементы массива на печать
  return 0;
}
```

Пример 2. Написать программу, которая, узнав от пользователя количество строк и столбцов двумерного динамического массива, выделяет место в памяти и заполняет его идущими подряд натуральными числами, начиная с 1. Далее необходимо, по номеру строки, заданному пользователем, вывести на печать элементы, делящиеся без остатка на 3, расположенные в этой строке. Проход по элементам массива должен осуществляться с помощью указателя.

```
#include <stdio.h>
int main(){
      int n,m;//определяем переменные для хранения количества элементов массива
      printf("Enter n: ");    //запрашиваем n - количество строк
      scanf("%i",&n);               //считываем n с консоли
      printf("Enter m: ");    //запрашиваем m
      scanf("%i",&m);               //считываем m с консоли
      //заводим динамический массив
      int** mass = new int *[n];
      for(int i=0; i<n; i++)
            mass[i] = new int [m];
      int k = 1;
      // присваиваем значения элементам массива,  *(*(mass+i)+j) то же, что и mass[i][j]
      for(int i=0; i<n; i++){
            for(int j=0; j<m; j++){
                  *(*(mass+i)+j)=k;
                  k++;
            }
      }
      int a;
      printf("Enter a: ");    //запрашиваем a
      scanf("%i",&a);         //считываем a с консоли
      for(int i=0;i<m;i++){
            if(*(*(mass+(a-1))+i)%3 == 0)
                  printf("%i ",*(*(mass+(a-1))+i));
      }
      //освобождение памяти - самостоятельно
      return 0;
}
```

Задание 4.

1. Все используемые массивы должны быть реализованы как динамические.
2. Доступ к элементам массива должен осуществляться через указатели.
3. Массив заполняется пользователем с клавиатуры.
4. Каждый логически законченный фрагмент должен быть оформлен в виде отдельной функции. Все необходимые данные для функции должны передаваться в качестве параметров (глобальные переменные запрещены).
5. Реализовать возможность считывания (записи) исходных данных из файла (в файл).
6. Имя файла должно запрашиваться у пользователя.
7. Программа, исходя из содержимого файла, должна выдавать сообщение об ошибке в случае, если данные не удовлетворяют условиям задачи.

4.1. Дана квадратная матрица A порядка n. Получить матрицу $A+A^2$.
4.2. Дана квадратная матрица A порядка n. Получить матрицу A^2-2A.
4.3. Дана квадратная матрица A порядка n. Получить матрицу $E+A^2$, где E – единичная матрица порядка n.
4.4. Дана квадратная матрица A порядка n. Получить матрицу $(E+A)^2$, где E – единичная матрица порядка n.
4.5. Дана квадратная матрица A порядка n. Получить матрицу $(A-3E)^2$, где E – единичная матрица порядка n.
4.6. Дана квадратная матрица A порядка n. Получить матрицу AB; элементы матрицы B вычисляются по формуле: $b_{ij} = \frac{1}{i+j+1}$.
4.7. Дана квадратная матрица A порядка n. Получить матрицу BA; элементы матрицы B вычисляются по формуле: $b_{ij} = \frac{1}{|i-j|+1}$.
4.8. Дана квадратная матрица A порядка n. Получить матрицу AB; элементы матрицы B вычисляются по формуле: $b_{ij} = \begin{cases} \frac{1}{i+j+1}, i \le j \\ \frac{1}{i+j+3}, else \end{cases}$
4.9. Дана квадратная матрица A порядка n. Получить матрицу BA; элементы матрицы B вычисляются по формуле: $b_{ij} = \begin{cases} \frac{1}{i+j+1}, i \ge j \\ \frac{1}{i+j+3}, else \end{cases}$
4.10. Дана квадратная матрица A порядка n. Получить матрицу AB; элементы матрицы B вычисляются по формуле: $b_{ij} = \begin{cases} \frac{1}{i+j+1}, i < j \\ 0, i = j \\ -\frac{1}{i+j+1}, else \end{cases}$
4.11. Дана квадратная матрица A порядка n. Получить матрицу BA; элементы матрицы B вычисляются по формуле: $b_{ij} = \begin{cases} -\frac{1}{i+j+1}, i < j \\ 0, i = j \\ \frac{1}{i+j+3}, else \end{cases}$
4.12. Дана квадратная матрица A порядка n. Получить матрицу b^TAb; элементы вектора b вычисляются по формуле: $b_i = \frac{1}{i^2+2}$.
3.13. Дана квадратная матрица A порядка n. Получить матрицу b^TAb; элементы вектора b вычисляются по формуле: $b_i = \begin{cases} \frac{1}{i^2+2}, i\text{-}\textit{четное} \\ \frac{1}{i+1}, else \end{cases}$
3.14. Дана квадратная матрица A порядка n. Получить матрицу b^TA^2b; элементы вектора b вычисляются по формуле: $b_i = \begin{cases} \frac{1}{i^2+2}, i\text{-}\textit{нечетное} \\ \frac{1}{i+1}, else \end{cases}$
4.15. Дана квадратная матрица A порядка n. Получить матрицу $b^T(A-E)b$, где E – единичная матрица порядка n; элементы вектора b вычисляются по формуле: $b_i = \frac{1}{i^3+2}$.
4.16. Дана квадратная матрица A порядка n. Получить матрицу A(A-E)+C, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{1}{i+j+1}$.
4.17. Дана квадратная матрица A порядка n. Получить матрицу 2(A+E)+C, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{1}{i+j+1}$.

4.18. Дана квадратная матрица A порядка n. Получить матрицу $(A-E)^2+C$, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{1}{i+j+1}$.

4.19. Дана квадратная матрица A порядка n. Получить матрицу A-3C+E, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{1}{(i-j)^2+1}$.

4.20. Дана квадратная матрица A порядка n. Получить матрицу A^2+C, где элементы матрицы C вычисляются по формуле $c_{ij} = \mathrm{sgn}(i-j)\cdot 2$.

4.21. Дана квадратная матрица A порядка n. Получить матрицу A^2+CA, где элементы матрицы C вычисляются по формуле $c_{ij} = |i-j|$.

4.22. Дана квадратная матрица A порядка n. Получить матрицу $(A-E)^2+C^2$, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{\mathrm{sgn}(i-j)}{i+j+1}$.

4.23. Дана квадратная матрица A порядка n. Получить матрицу $b^TA^2b^2$; элементы вектора b вычисляются по формуле: $b_i = \begin{cases} \frac{1}{i^2+2}, & i\text{-}\textit{нечетное} \\ -\frac{1}{i+1}, & \textit{else} \end{cases}$

4.24. Дана квадратная матрица A порядка n. Получить матрицу A(A-E)+C-E, где E – единичная матрица порядка n, а элементы матрицы C вычисляются по формуле $c_{ij} = \frac{i-j}{i+j}$.

Пример 1. Сортировка целочисленного массива методом выбора. **Добейтесь успешной компиляции.**

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
//==========функция заполнения массива==========
int RandVector(int size, int *massiv, int max_rand){
    srand(time(0));
    //рандомизация - установка начального псевдослучайного числа
    for(int i = 0; i < size; i++){
        massiv[i] = 1 + (rand() % max_rand);
        //случайное число в диапазоне от 1 до 10
    }
    return 0;
}
//==========функция вывода массива на консоль==========
int PrintVector(int size, int *massiv){
    for(int i = 0; i < size; i++){
        printf("%i ",massiv[i]);
    }
    printf("\n");
    return 0;
}
//==========функция сортировки==========
int SortVector(int size, int *massiv){
    int i_min, a;
    for(int i = 0; i < size-1; i++){
        i_min = i; //наименьший - текущий
        for(int j = i+1; j < size; j++){
            if(massiv[j] < massiv[i_min])
                i_min = j; //минимальный из еще неупорядоченных
        }
        //меняем местами элементы i и i_min
        a = massiv[i];
        massiv[i] = massiv[i_min];
        massiv[i_min] = a;
    }
    return 0;
}
//==========функция main()==========
int main(){
    int n;
    printf("Enter number of elements: ");
    scanf("%i",&n);
    //==========выделение памяти==========
    int *vector = new int [n];
    //==========заполнение массива==========
    RandVector(n,vector,10);
    //==========проверка заполнения==========
    PrintVector(n,vector);
    //==========сортировка==========
    SortVector(n,vector);
    //==========вывод упорядоченного массива==========
    PrintVector(n,vector);
    //==========освобождение памяти==========
    delete [] vector;
    return 0;
}
```

Пример 2. Число четных элементов в двумерном динамическом массиве. **Измените код так, чтобы каждый пункт обрабатывался отдельной функцией (глобальные переменные запрещены!!!).**

```
    int n = 5, m = 5;
1.
    int **matr = new int *[n];
    // выделяется память под массив указателей на строки массива
    for(int i=0; i<n; i++){
        matr[i] = new int [m];
            // выделяется память под m элементов типа int
    }
2.
    // заполненеие
    srand(time(0));
    for(int i=0; i<n; i++)
        for(int j=0; j<m; j++)
            matr[i][j] = (rand() % 11) - 5;
                  // диапазон значений [-5,5]
3.
    // проверка заполнения
    for(int i=0; i<n; i++){
        for(int j=0; j<m; j++){
                printf("%2i ", matr[i][j]);
        }
        printf("\n");
    }
4.
    // подсчет числа четных элементов
    int col_odd = 0; // помните про вырожденные случаи!
    for(int i=0; i<n; i++){
        for(int j=0; j<m; j++){
            if (matr[i][j] % 2 == 0)
                col_odd++;
        }
    }
    printf("The sum of even elements: %i",col_odd);
5.
    // освобождение памяти
    for(int i=0; i<n; i++){
        delete [] matr[i];
    }
    delete [] matr;
```