void findMin (int a[], int n)

```
#include <iostream>
void findMin (int a[], int n)
{
	int min = a[0];
	for (int i = 1; i < n; i++)
	{
		if (a[i] < min)
		{
			min = a[i];
		}
	}
	 std ::cout << "The minimun element is" << min << std::endl;
}
```

Arrays/findIndex

```
#include <iostream>
void findIn dex(int a[], int n, int key)
{
	int start = -1, end = -1;
	for (int i = 0; i < n; i++)
	{
		 if(a[i] == key)
		{
			start = i;
			break;
		}
	}
	for (int j = n - 1; j >= 0; j--)
	{
		 if(a[j] == key)
		{
			end = j;
			 break;
		}
```

Arrays/findIndex (cont)

```
>    }
    if (start == -1 || end == -1)
    {
        std::cout << -1;
    }
    else
    {
        std::cout << start << " " << end;
    }
}
int main()
{
    int n;
    std::cin >> n;
    int a[n];
    for (int i = 0; i < n; i++)
    {
        std::cin >> a[i];
    }
    int key;
    std::cin >> key;
    findIndex(a, n, key);
    return 0;
}
```

đếm số lần xuất hiện của giá trị x

```
#include <iostream>
int countO ccu rre nce s(int a[], int n, int x)
{
        int count = 0;
        for (int i = 0; i < n; ++i)
        {
                 if(a[i] == x)
                {
```

đếm số lần xuất hiện của giá trị x (cont)

```
>         ++count;
      }
    }
    return count;
}
```

kiểm tra mảng có tăng dần không

```
#include <iostream>
bool isAsce ndi ngO rde r(int a[], int n)
{
        for (int i = 0; i < n - 1; ++i)
        {
                 if(a[i] > a[i + 1])
                {
                         return false;
                }
        }
         return true;
}
```

concatTwoArrays

```
#include <iostream>
void concat Two Arr ays(int a[], int n, int b[], int m, int c[])
{
        int k = m + n;
        int j = n;
        for (int i = 0; i < n; ++i)
        {
                c[i] = a[i];
        }
        for (int g = 0; j < k; ++g)
        {
              c[j] = b[g];
              ++j;
```

concatTwoArrays (cont)

```
>  }
}
```

xuất chuỗi theo thứ tự ngược

```
#include <iostream>
void revers eCh arArray (char a[], int n)
{
	char tmp;
	for (int i = 0; i < n / 2; ++i)
	{
		tmp = a[i];
		a[i] = a[n - i - 1];
		a[n - i - 1] = tmp;
	}
}
```

kiểm tra chuỗi đối xứng

```
#include <iostream>
bool checkS ymm etr icA rra y(char a[], int n)
{
	for (int i = 0; i < n / 2; i++)
	{
		 if(a[i] != a[n - i -1])
		{
			return false;
		}
	}
	 return true;
}
```

đếm số lần xuất hiện của kí tự c

```
int countOccurrencesChar(char a[], int n, char c)
{
	int count = 0;
	for (int i = 0; i < n; ++i)
```

đếm số lần xuất hiện của kí tự c (cont)

```
>   {
      if (a[i] == c)
      {
         ++count;
      }
   }
   return count;
}
```

Hàm nhập mảng m dòng n cột

```
void set2dArray(int a[][MAX], int &m, int &n)
{
         for(int i = 0; i < m; i++)
        {
                  for(int j = 0; j < n; j++)
                 {
                           std ::cin >> a[i][j];
                 }
        }
}
```

h tổng các phần tử của mảng 2d

```
int sumArray(int a[][MAX], int m, int n)
{
        int sum = 0;
        for (int i = 0; i < m; i++)
        {
                for (int j = 0; j < n; j++)
                {
                        sum = sum + a[i][j];
                }
        }
         return sum;
}
```

Hàm tính tổng đường chéo chính

```
#include <iostream>
```

Hàm tính tổng đường chéo chính (cont)

```
> int sumDiagonal (int a[][MAX], int m, int n)
{
   int sum = 0;
   for(int i = 0; i < m; i++)
   {
      sum = sum + a[i][i];
   }
   return sum;
}
```

đếm mảng có bao nhiêu số nguyên tố

```
#include <iostream>
#include <cm ath>
bool checkP rim e(int n)
{
        if (n <= 1)
        {
                 return false;
        }
        else if (n == 2)
        {
                 return true;
        }
        else if (n % 2 == 0)
        {
                 return false;
        }
        else if (n > 2)
        {
                for (int i = 3; i <= sqrt(n); i += 2)
                {
                        if (n % i == 0)
                         return false;
                }
```

đếm mảng có bao nhiêu số nguyên tố (cont)

```
>   }
    return true;
}
int countNumberofPrime (int a[], int n)
{
    int count = 0;
    for(int i = 0; i < n; i++)
    {
        if(checkPrime(a[i]))
        {
            ++count;
        }
    }
    return count;
}
```

longerSyl(word, len, syllable)

```
#include<iostream>
#inclu de< str ing>
bool isVowel (char t)
{
return (t =='a' || t =='e' || t =='i' || t =='o' || t =='u' || t =='y' ||
t =='A' || t =='E' || t =='I' || t =='O' || t =='U' || t =='Y');
}
std::s tring findLo nge stS yl( std ::s tring senten ces){
// sentences = "hoc mon nhap mon lap trinh rat vui"
// return " inh "

int i = 0, max_index, max_len = 0, curren t_i ndex, curren t_len = 0;
while (!isVo wel (se nte nce s[i])) i ++;
max_index = i;
// Assign max to first syllabus
while( sen ten ces[i] != ' ' && senten ces[i] != 0)
{
i++;
```

longerSyl(word, len, syllable) (cont)

```
> max_len++;
}
while(sentences[i] != 0)
{
current_len = 0;
while(!isVowel(sentences[i])) i++;
current_index = i;
while((sentences[i] != ' ' && sentences[i] != 0))
{
i++;
current_len++;
}
if(current_len > max_len)
{
max_len = current_len;
max_index = current_index;
}
}
// Copy to string
char sys[20] = {0};
for(int j = 0; j < max_len; j++)
{
sys[j] = sentences[j + max_index];
}
return (std::string) sys;
}
int main()
{
char sentences[] = "hooooooc mon nhap mon lap trinh rat vuaWQWVViqw";
std::string output = findLongestSyl(sentences);
std::cout << output;
}
```

findext lam dai dfskdhfkjsfh

```
#include <iostream>
```

findext lam dai dfskdhfkjsfh (cont)

```
> #include <string>
#include <cstring>
void FindExt( std::string paths[], int nPath, std::string extName, std::string results[], int &nRes)
{
   for (int i = 0; i < nPath; i++)
   {
      if(extName == paths[i])
      {
         for (int j = 0; j < nPath; j++)
         {
            results[j] = paths[i];
         }
         nRes++;
      }
   }
}
```

---

poe

```
#include <iostream>
#include <string>
void FindExt(std::string paths[], int nPath, std::string extName, std::string results[], int& nRes)
{
   for (int i = 0; i < nPath; i++)
   {
      if (paths[i].find(extName) != std::string::npos)
      {
         results[nRes] = paths[i];
         nRes++;
      }
   }
}
int main()
{
```

findext lam dai dfskdhfkjsfh (cont)

```
>    std::string paths[] = { "D:/Others/us/b.pdf", "C:/Users/c.txt", "C:/Users/adm/a.txt", "E:/temp/d.exe" };
    int nPath = 4;
    std::string results[4];
    std::string extName = "txt";
    int nRes = 0;
    FindExt(paths, nPath, extName, results, nRes);
    for (int i = 0; i < nRes; i++)
    {
        std::cout << results[i] << " ";
    }
    return 0;
}
```

void RemovePhone

```
#include <iostream>
struct Phone
{
int ID;
int version;
int year;
};
void Remove Pho ne( Phone listPh one[], int &n Phone, int curYear)
{
        int elemen tsD eleted = 0;
        for (int i = 0; i < nPhone; i++)
        {
                if (curYear - listPh one [i].year > 4)
                {
                         for(int j = i; j < nPhone; j++)
                        {
                         lis tPh one[j] = listPh one[j + 1];
                        }
                        i--;
                         nPh one--;
```

void RemovePhone (cont)

```
>     }
   }
}
int main()
{
   Phone listPhone[100] = {{10, 9, 2019}, {20, 10, 2020}, {21, 11, 2021}, {22, 12, 2022}, {9, 8, 2018}};
   int nPhone = 5;
   int curYear = 2024;
   RemovePhone(listPhone, nPhone, curYear);
   // In ra ID cua cac dien thoai con lai
   for (int i = 0; i < nPhone; i++)
   std::cout << listPhone[i].ID << " ";
   return 0;
}
```

vd cua convert char array to string

```
#include <iostream>
#include <st rin g>
int main()
{
       char cstr[] = " clc 03";
        std ::s tring s(cstr);
        std ::cout << s;
        return 0;
}
```

Hàm xoá các phần tử có giá trị x

```
#include <iostream>
void delete Items (int a[], int &n, int x)
{
       for (int i = 0; i < n; i++)
       {
               if (a[i] == x)
               {
```

Hàm xoá các phần tử có giá trị x (cont)

```
>          for (int j = i; j < n; j++)
        {
           a[j] = a[j + 1];
        }
        i--;
        n--;
     }
   }
}
```

void deleteItemsFromIndex

```
#include <iostream>
void delete Ite msF rom Ind ex(int a[], int &n, int index, int no_items)
{
         for(int i = index ; i < index + no_items; ++i)
        {
                if (i + no_items < n)
                {
                        a[i] = a[i + no_items];
                }
                else
                {
                         break;
                }
        }
        n = n - std::m in( no_ items, std::max(n - index, 0));
}
```

mảng đánh dấu

```
//hãy đếm xem có bao nhiêu giá trị khác nhau trong mảng;
int main() {
        int count[ 100 0001] = {0};
        int n;
        cin >> n;
```

mảng đánh dấu (cont)

```
>     int a[n];
    for(int i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    for (int i = 0; i < n; i++)
    {
        count[a[i]] = 1;
    }
    int ans = 0;
    for (int i = 0; i <= 1000000; i++)
    {
        if(count[i] == 1) ++ans;
    }
    cout << ans;
    return 0;
}
// hãy liệt kê các giá trị xuất hiện trong mảng theo thứ tự từ nhỏ đến lớn kèm theo tần suất của nó
    int cnt[1000001];
int main()
{
    int n;
    cin >> n;
    int a[n];
    for (int i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    for (int i = 0; i < n; i++)
    {
        cnt[a[i]]++;
    }
    for (int i = 0; i <= 1000000; i++)
```

mảng đánh dấu (cont)

```
>   {
      if(cnt[i] != 0) cout << i << " " << cnt[i] << endl;
   }
   return 0;
}
```

mò 1d arra

```
//liệt kê
for(int i = 0; i < n; i++)
{
bool check = true;
for(int j = 0; j <= n - 1; j++)
{
if(a[j] == a[i])
{
check = false;
break;
}
}
if(check) cout << a[i] << " ";
}
// đếm tần suất
for (int i = 0; i < n; i++)
{
bool check = true;
for (int j = 0; j <= i - 1; j++)
{
if(a[j] == a[i])
{
check = false;
break;
}
}
if (check)
```

mở 1d arra (cont)

```
> {
int frequency = 1;
for (int j = i + 1; j < n; j++)
{
if(a[i] == a[j]) ++frequency;
}
cout << a[i] << " " << frequency << endl;
}
}
ool checkdx(int a[], int n)
{
for(int i = 0; i < n / 2; i++)
{
if(a[i] != a[n - i - 1]) return false;
}
return true;
}
//lật mảng: duyệt các element đối xứng nhau và swap nó là được
void latmang(int a[], int n)
{
for(int i = 0; i < n /2; i++)
{
swap(a[i], a[n - 1 - i]);
}
}
// xóa phần tử tại k
void xoa(int a[], int &n, int k)
{
for(int i = k; i < n - 1; i++)
{
a[i] = a[i + 1];
}
--n;
```

mò 1d arra (cont)

```
> }
// --n để giảm kích thước của mảng
//chèn phần tử
void chen(int a[], int &n, int k, int x)
{
for(int i = n; i > k; i--) //không phải n - 1 vì
{
a[i] = a[i - 1];
}
a[k] = x;
++n;
//++n dê tăng kích thước của mảng
}
// fibo
bool check(long long n)
{
long long f[100];
f[0] = 0;
f[1] = 1;
for (int i = 2; i <= 92; i++)
{

f[i] = f[i - 1] + f[i - 2];
}
for (int i = 0; i <= 92; i++)
{
if(f[i] == n) return true;
}
return false;
}
// trộn 2 mảng tăng dần
int a[] = {1, 2, 5, 6};
int b[] = {4, 8, 9, 10, 11, 29};
int n = 4 //số elements của a;
```

mở 1d arra (cont)

```
> int m = 6 //só elements của b;
int i = 0, j = 0;
while(i < n && j < m)
{
if(a[i] <= b[j])
{
cout << a[i] << " ";
i++;
}
else
{
cout << b[j] << " ";
j++;
}
}
while (i < n)
{
cout << a[i] << " ";
i++;
}
while (j < m)
{
cout << b[j] << " ";
j++;
}
//tìm phần tử đầu tiên lớn hơn nó nằm ở bên phải, đối với phần tử không có phần tử lớn hơn bên phải thì in ra -1, ví dụ mảng A[] là {3, 8, 9, 1, 4, 2, 5} thì kết quả sẽ là 8, 9, -1, 4, 5, 5, -1.
 int main()
{
int n;
cin >> n;
int a[n];
for (int i = 0; i < n; i++)
{
```

mò 1d arra (cont)

```
> cin >> a[i];
}
for (int i = 0; i < n; i++)
{
bool check = true;
for(int j = i + 1; j < n; j++)
{
if(a[i] < a[j])
{
check = false;
cout << a[j] << " ";
break;
}
}
if(check)
{
cout << -1 << " ";
}
}
}
```

fibo matrix but it's ai created

```
#include <iostream>
using namespace std;
void print_ spi ral _fi bo(int n) {
       int fib[n*n] = {0, 1};
       for (int i = 2; i < n*n; i++) {
                fib[i] = fib[i-1] + fib[i-2];
       }
       int start_row = 0, start_col = 0, end_row = n-1, end_col = n-1;
        while (start_row <= end_row && start_col <= end_col) {
               // Print the first row from the remaining rows
               for (int i = start_col; i <= end_col; i++) {
                       cout << fib[st art_row * n + i] << " ";
```

fibo matrix but it's ai created (cont)

```
>       }
        start_row++;
        // Print the last column from the remaining columns
        for (int i = start_row; i <= end_row; i++) {
            cout << fib[i * n + end_col] << " ";
        }
        end_col--;
        // Print the last row from the remaining rows
        if (start_row <= end_row) {
            for (int i = end_col; i >= start_col; i--) {
                cout << fib[end_row * n + i] << " ";
            }
            end_row--;
        }
        // Print the first column from the remaining columns
        if (start_col <= end_col) {
            for (int i = end_row; i >= start_row; i--) {
                cout << fib[i * n + start_col] << " ";
            }
            start_col++;
        }
    }
}
int main() {
    int n;
    cout << "Enter the size of the spiral matrix: ";
    cin >> n;
    print_spiral_fibo(n);
    return 0;
}
```

This function first generates an array of Fibonacci numbers up to `

Is this conversation helpful so far?

give me some test cases

fibo matrix but it's ai created (cont)

> Edit

Sources: geeksforgeeks.org (1) programiz.com (2) geeksforgeeks.org (3) javatpoint.com (4)

Sure, here are some test cases for the spiral matrix Fibonacci C++ program:

Test Case n Output

1 3 0 1 1 2 3 5

2 4 0 1 1 2 3 5 8 13

3 5 0 1 1 2 3 5 8 13 21 34

4 6 0 1 1 2 3 5 8 13 21 34 55 89

5 7 0 1 1 2 3 5 8 13 21 34 55 89 144 233

And here is the C++ code to print a spiral matrix whose elements are Fibonacci numbers:

cpp

Edit

Full Screen

Copy code

```
#include <iostream>
using namespace std;
void printSpiral(int n) {
    int fib[n*n] = {0, 1};
    for (int i = 2; i < n*n; i++) {
        fib[i] = fib[i-1] + fib[i-2];
    }
    int start_row = 0, start_col = 0, end_row = n-1, end_col = n-1;
    while (start_row <= end_row && start_col <= end_col) {
        // Print the first row from the remaining rows
        for (int i = start_col; i <= end_col; i++) {
            cout << fib[start_row * n + i] << " ";
        }
        start_row++;
        // Print the last column from the remaining columns
        for (int i = start_row; i <= end_row; i++) {
            cout << fib[i * n + end_col] << " ";
        }
        end_col--;
```

fibo matrix but it's ai created (cont)

```
>       // Print the last row from the remaining rows
      if (start_row <= end_row) {
        for (int i
```

spiral matrix (ngược chiều dượng

```
// C++ Program to print a matrix spirally

#include <bi ts/ std c++.h>
using namespace std;
#define R 4
#define C 4

void spiral Pri nt(int m, int n, int a[R][C])
{
       int i, k = 0, l = 0;

       /* k - starting row index
               m - ending row index
               l - starting column index
               n - ending column index
               i - iterator
       */

        while (k < m && l < n) {
               /* Print the first row from
                              the remaining rows */
               for (i = l; i < n; ++i) {
                       cout << a[k][i] << " ";
               }
               k++;

               /* Print the last column
                 from the remaining columns */
               for (i = k; i < m; ++i) {
                       cout << a[i][n - 1] << " ";
               }
               n--;

               /* Print the last row from
                               the remaining rows */
```

spiral matrix (ngược chiều dượng (cont)

```
>       if (k < m) {
          for (i = n - 1; i >= l; --i) {
             cout << a[m - 1][i] << " ";
          }
          m--;
       }

       /* Print the first column from
             the remaining columns */
       if (l < n) {
          for (i = m - 1; i >= k; --i) {
             cout << a[i][l] << " ";
          }
          l++;
       }
    }
}
```

tượng

```
int countSquares(int row, int column)
{

        // Count top left squares
        int topLeft = min(row, column) - 1;

        // Count bottom right squares
        int bottom Right = 8 - max(row, column);

        // Count top right squares
        int topRight = min(row, 9 - column) - 1;

        // Count bottom left squares
        int bottomLeft = 8 - max(row, 9 - column);

        // Return total count
         return (topLeft + topRight + bottom Right + bottom Left);
}
```

quân mã

```
int findPossibleMoves(int mat[n][m], int p, int q)
```

quân mã (cont)

```
> {
    // All possible moves of a knight
    int X[8] = { 2, 1, -1, -2, -2, -1, 1, 2 };
    int Y[8] = { 1, 2, 2, 1, -1, -2, -2, -1 };

    int count = 0;

    // Check if each possible move is valid or not
    for (int i = 0; i < 8; i++) {

        // Position of knight after move
        int x = p + X[i];
        int y = q + Y[i];

        // count valid moves
        if (x >= 0 && y >= 0 && x < n && y < m
            && mat[x][y] == 0)
            count++;
    }

    // Return number of possible moves
    return count;
}
```

lớn nhất và lớn thứ 2

```
int main() {
        long long m;
        cin >> m;
        long long a[m];
         for (long long i = 0; i < m; i++)
        {
                cin >> a[i];
        }
        long long max1 = 0;
        long long max2 = 0;
        int index1 = 0;
         for(int i = 0; i < m; i++)
        {
                 if(a[i] > max1)
```

lớn nhất và lớn thứ 2 (cont)

```
>      {
         max1 = a[i];
         index1 = i;
      }
   }
   for(int i = 0; i < m; i++)
   {
      if(a[i] > max2 && i != index1)
      {
         max2 = a[i];
      }
   }
   cout << max1 << " " << max2;
   return 0;
}
```

lớn nhất và lớn thứ 2

```
int main() {
        long long m;
        cin >> m;
        long long a[m];
         for (long long i = 0; i < m; i++)
        {
                cin >> a[i];
        }
        long long max1 = 0;
        long long max2 = 0;
        int index1 = 0;
         for(int i = 0; i < m; i++)
        {
                 if(a[i] > max1)
                {
                        max1 = a[i];
                         index1 = i;
```

lớn nhất và lớn thứ 2 (cont)

```
>     }
   }
   for(int i = 0; i < m; i++)
   {
      if(a[i] > max2 && i != index1)
      {
         max2 = a[i];
      }
   }
   cout << max1 << " " << max2;
   return 0;
}
```

lớn nhất và lớn thứ 2

```
int main() {
        long long m;
        cin >> m;
        long long a[m];
         for (long long i = 0; i < m; i++)
        {
                cin >> a[i];
        }
        long long max1 = 0;
        long long max2 = 0;
        int index1 = 0;
         for(int i = 0; i < m; i++)
        {
                  if(a[i] > max1)
                {
                        max1 = a[i];
                         index1 = i;
                }
        }
         for(int i = 0; i < m; i++)
```

lớn nhất và lớn thứ 2 (cont)

```
>   {
      if(a[i] > max2 && i != index1)
      {
         max2 = a[i];
      }
   }
   cout << max1 << " " << max2;
   return 0;
}
```

gcd in arrays

```
#include <cmath>
#include <cs tdi o>
#include <ve cto r>
#include <io str eam>
#include <al gor ith m>
using namespace std;
long long gcd(long long a, long long b)
{
        while (b!= 0)
       {
               long long r = a % b;
               a = b;
               b = r;
       }
        return a;
}
int main() {
       long long n;
       cin >> n;
       long long a[n];
       for (int i = 0; i < n; i++)
       {
               cin >> a[i];
```

gcd in arrays (cont)

```
>   }
    long long gcd1 = gcd(a[0], a[1]);
    for(int i = 0; i < n; i++)
    {
        gcd1 = gcd(gcd1, a[i]);
    }
    cout << gcd1;
    return 0;
}
```

fibo

```
bool isFibo(long long n)
{
        long long f[93];
        if (n == 1 || n == 0) return true;
        f[0] = 0;
        f[1] = 1;
        for (int i = 2; i < 93; i++)
        {
                f[i] = f[i - 1] + f[i - 2];
        }
        for (int i = 0; i < 93; i++)
        {
                 if(f[i] == n)
                {
                        return true;
                }
        }
         return false;
}
```

week5

```
#include <iostream>
using namespace std;
```

week5 (cont)

```
> #define MAX 100
void set2DArray(int a[][MAX], int &m, int &n) {
cin >> m >> n;

for (int i = 0; i < m; i++)
for (int j = 0; j < n; j++)
cin >> a[i][j];
}
void print2DArr(int a[][MAX], int m, int n) {
for (int i = 0; i < m; i++) {
for (int j = 0; j < n; j++)
cout << a[i][j] << ' ';
cout << endl;
}
cout << endl;
}
int sum2DArray(int a[][MAX], int m, int n) {
int sum = 0;
for (int i = 0; i < m; i++)
for (int j = 0; j < n; j++)
sum += a[i][j];
return sum;
}
int sumDiagonal(int a[][MAX], int m, int n) {
int sum = 0;
// for (int i = 0; i < m; i++) {
// for(int j = 0; j < n; j++) {
// if(i == j)
// sum += a[i][j];
// }
// }

for (int i = 0; i < m; i++)
sum += a[i][i];

return sum;
```

week5 (cont)

```
> }
int findMin(int a[][MAX], int m, int n) {
int min = a[0][0];
for(int i = 0; i < m; i++) {
for(int j = 0; j < n; j++) {
if (a[i][j] < min)
min = a[i][j];
}
}
return min;
}
bool checkPrime(int n)
{
if (n < 2)
return false;
if (n == 2)
return true;
if (n % 2 == 0)
return false;
for (int i = 3; i <= n / 2; i = i + 2)
{
if (n % i == 0)
return false;
}

return true;
}
int countNumberOfPrime(int a[][MAX], int m, int n)
{
int count = 0;
for (int i = 0; i < m; ++i)
{
for (int j = 0; j < n; ++j)
{
```

week5 (cont)

```
> count += checkPrime(a[i][j]);
}
}

return count;
}
bool isUnappear(int a[], int n, int x){
for(int i = 0; i < n; i++){
if(a[i] == x)
return false;
}
return true;
}
int countNumberOfUniquePrime(int a[][MAX], int m, int n){
int temp[MAX*MAX];
int l = 0;
for(int i = 0; i < m; i++){
for(int j = 0; j < n; j++){
if(checkPrime(a[i][j]) && isUnappear(temp, l, a[i][j])){
temp[l] = a[i][j];
l++;
}
}
}
return l;
}
int countOccurrences(int a[][MAX], int m, int n, int x) {
int count = 0;
for( int i = 0; i < m; i++)
for (int j = 0; j < n; j++)
if (a[i][j] == x)
count++;

return count;


}
```

week5 (cont)

```
> void delete_row(int a[][MAX], int &m, int n, int idx)
{
for (int i = idx; i < m - 1; i++)
{
for (int j = 0; j < n; j++)
{
a[i][j] = a[i + 1][j];
}
}
m--;
}
void delete_col(int a[][MAX], int m, int &n, int idx)
{
for (int i = 0; i < m; i++)
{
for (int j = idx; j < n - 1; j++)
{
a[i][j] = a[i][j + 1];
}
}

// for (int j = idx; j < n - 1; j++)
// for (int i = 0; i < m; i++)
// a[i][j] = a[i][j + 1];

n--;
}
void deleteItem(int a[][MAX], int &m, int &n, int x){
for(int i = 0; i < m; i++){
for(int j = 0; j < m; j++){
if(a[i][j] == x){
delete_row(a, m, n, i);
delete_col(a, m, n, j);
j--;
}
```

week5 (cont)

```
>}
}
}
```

struct sinhvien

```
#include <iostream>
#include <cs tri ng>
#include <io man ip>
#pragma pack(1)
struct SINHVIEN
{
       char MSSV[9];
       char Ho_va_ ten _lo t[31];
       char Ten[9];
        float mon1 = 0;
        float mon2 = 0;
        float mon3 = 0;
};
void print( SIN HVIEN sv[], int n)
{
        std ::cout << " MSS V" << std::s etw(10) << " HO" << std::s etw(10) << " Ten " << std::s etw(10)
<< " D1" << std::s etw(10) << " D2" << std::s etw(10) << " D3 \n";
        for(int i = 0; i < n; i++)
                std ::cout << sv[i].MSSV << std::s etw(10) << sv[i].H o_ va_ ten_lot << std::s etw(10) <<
sv[i].Ten << std::s etw(10) << sv[i].mon1 << std::s etw(10) << sv[i].mon2 << std::s etw(10) << sv[i].mon3
<< " \n";
}
void sortBy Nam e(S INHVIEN sv[], int n)
{
        for(int i = 0; i < n; i++) {
                for(int j = n - 1; j > i; j--) {
                        if( str cmp (sv [i].Ten, sv[j].Ten) > 0)
                                std ::s wap (sv[i], sv[j]);
               }
       }
}
int main()
```

struct sinhvien (cont)

```
> {
   SINHVIEN sv[]={{"23127489", "Nguyen Ngoc Minh", "Thu", 5.0, 9.0, 6.5}, {"23127237", "Le Truong Bao", "Ngoc", 7.0, 4.5, 8.7}, {"23127525", "Tran Ngoc", "Dieu", 9.7, 1.2, 5.6}};
   int n = sizeof(sv)/sizeof(SINHVIEN);
   std::cout << "Danh sách ban đầu: \n";
   print(sv, n);
   sortByName(sv, n);
   std::cout << "Danh sách sau khi sắp xếp theo tên: \n";
   print(sv, n);
   return 0;
}
```

tìm min của mảng 2d

```
#include <iostream>
int findMi n(int a[][MAX], int m, int n)
{
        int min = a[0][0];
        for (int i = 0; i < m; i++)
        {
                for (int j = 0; j < n; j++)
                {
                          if( a[i][j] < min)
                         min = a[i][j];
                }
        }
         return min;
}
```

deleteItems

```
void deleteItems(int a[][MAX], int& m, int& n, int x)
{
        int elemen tsD eleted = 0; // Variable to keep track of the number of elements deleted
        for (int i = 0; i < m; i++)
        {
                for (int j = 0; j < n; j++)
```

deleteItems (cont)

```
>       {
         if (a[i][j] == x)
         {
            // Shift elements to the left starting from the next position
            for (int k = j; k < n - 1; k++)
            {
               a[i][k] = a[i][k + 1];
            }
            elementsDeleted++;
            // Decrement the number of columns
            n--;
            // Decrement j to recheck the new element at the current position
            j--;
         }
      }
   }
   // Update the number of rows after deletion
   m -= elementsDeleted;
}
```

final lab

```
#include <iostream>
#include <st rin g>
#include <fs tre am>
/* The member function cin.ge tline() works with C strings (i.e. arrays of char)
whereas the free function std::g etl ine() works with C++ strings (i.e. std::s tring.)
You should not be using C strings at all when learning C++, which means
you should not be using cin.ge tli ne().*/
using namespace std;
void FindEx t(s td: :string paths[], int nPath, std::s tring extName, std::s tring results[], int &nRes)
{
nRes = 0;
for(int i = 0; i < nPath; i++)
{
```

final lab (cont)

```
> string tmp = paths[i].substr(paths[i].find('.') + 1);
if(tmp == extName)
{
++nRes;
for(int j = 0; j < nRes; j++)
{
results[j] = paths[i];
}
}
}
}
void Find(string paths[], int nPath, string curDir, string results[], int &nRes)
{

}
int main()
{
string paths[] = {"D:/Others/us/b.pdf", "C:/Users/c.txt", "C:/Users/adm/a.txt", "E:/temp/d.exe"};
string extName = "txt";
string results[1000];
int nRes;
FindExt(paths, 4, extName, results, nRes);
for(int i = 0; i < nRes; i++)
{
cout << results[i] << " ";
}
cout << endl;
cout << nRes;
return 0;
}
#include <iostream>
using namespace std;
void doMin(int a[][3], int m, int n)
{
```

final lab (cont)

```
> int res[100][100] = {0};

int toaDoX[8] = {-1, -1, -1, 0, 0, 1, 1, 1};
int toaDoY[8] = {-1, 0, 1, -1, 1, -1, 0, 1};

for(int i = 0; i < m; ++i)
{
for(int j = 0; j < n; ++j)
{
if(a[i][j] == 1)
{
for(int k = 0; k < 8; ++k)
{
int newX = i + toaDoX[k];
int newY = j + toaDoY[k];
if(newX >= 0 && newY >= 0 && newX < m && newY < n)
{
++res[newX][newY];
}
}
}
}
}
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
a[i][j] = res[i][j];
}
}
}
int main()
{
int a[3][3] = {
1, 0, 0,
```

final lab (cont)

```
> 0, 1, 0,
0, 0, 0};
int m = 3;
int n = 3;
doMin(a, m, n);
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
cout << a[i][j] << " ";
}
cout << endl;
}
return 0;
}
#include <iostream>
#include <fstream>
using namespace std;
void ReadMap(int map[][100], int &m, int &n)
{
ifstream fin;
fin.open("map.txt");
if(!fin.is_open())
{
cout << "failed to open file!";
return;
}
fin >> m >> n;
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
fin >> map[i][j];
```

final lab (cont)

```
> }
}
fin.close();
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
cout << map[i][j] << " ";
}
cout << endl;
}
}
int main()
{
int map[100][100];
int m, n;
ReadMap(map, m, n);
return 0;
}
//các hàm
c_str: string to char
char array to string
// Demonstrates conversion
// from character array to string

#include <bits/stdc++.h>
using namespace std;

// converts character array
// to string and returns it
string convertToString(char* a, int size)
{
    int i;
    string s = "";
    for (i = 0; i < size; i++) {
```

final lab (cont)

```
>       s = s + a[i];
    }
    return s;
}

// Driver code
int main()
{
    char a[] = { 'C', 'O', 'D', 'E' };
    char b[] = "geeksforgeeks";

    int a_size = sizeof(a) / sizeof(char);
    int b_size = sizeof(b) / sizeof(char);

    string s_a = convertToString(a, a_size);
    string s_b = convertToString(b, b_size);

    cout << s_a << endl;
    cout << s_b << endl;

    return 0;
}
// C++ implementation of above approach
#include <bits/stdc++.h>
using namespace std;

#define N 4

// Function to rotate the matrix 90 degree clockwise
void rotate90Clockwise(int a[N][N])
{

    // Traverse each cycle
    for (int i = 0; i < N / 2; i++) {
        for (int j = i; j < N - i - 1; j++) {

            // Swap elements of each cycle
            // in clockwise direction
            int temp = a[i][j];
            a[i][j] = a[N - 1 - j][i];
            a[N - 1 - j][i] = a[N - 1 - i][N - 1 - j];
            a[N - 1 - i][N - 1 - j] = a[j][N - 1 - i];
```

final lab (cont)

```
>            a[j][N - 1 - i] = temp;
      }
   }
}

// Function for print matrix
void printMatrix(int arr[N][N])
{
   for (int i = 0; i < N; i++) {
      for (int j = 0; j < N; j++)
         cout << arr[i][j] << " ";
      cout << '\n';
   }
}

// Driver code
int main()
{
   int arr[N][N] = { { 1, 2, 3, 4 },
                     { 5, 6, 7, 8 },
                     { 9, 10, 11, 12 },
                     { 13, 14, 15, 16 } };
   rotate90Clockwise(arr);
   printMatrix(arr);
   return 0;
}
//anticlockwise
void rotateMatrix(int mat[][N])
{
   // Consider all squares one by one
   for (int x = 0; x < N / 2; x++) {
      // Consider elements in group
      // of 4 in current square
      for (int y = x; y < N - x - 1; y++) {
         // Store current cell in
```

final lab (cont)

```
>            // temp variable
            int temp = mat[x][y];

            // Move values from right to top
            mat[x][y] = mat[y][N - 1 - x];

            // Move values from bottom to right
            mat[y][N - 1 - x] = mat[N - 1 - x][N - 1 - y];

            // Move values from left to bottom
            mat[N - 1 - x][N - 1 - y] = mat[N - 1 - y][x];

            // Assign temp to left
            mat[N - 1 - y][x] = temp;
        }
    }
}

// Function to print the matrix
void displayMatrix(int mat[N][N])
{
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            cout << mat[i][j] << " ";
        }
        cout << endl;
    }
    cout << endl;
}

/ Driver code /
int main()
{
    // Test Case 1
    int mat[N][N] = { { 1, 2, 3, 4 },
                      { 5, 6, 7, 8 },
                      { 9, 10, 11, 12 },
                      { 13, 14, 15, 16 } };

    // Function call
```

final lab (cont)

```
>     rotateMatrix(mat);

    // Print rotated matrix
    displayMatrix(mat);

    return 0;
}
int binaryToDecimal(int n)
{
    int num = n;
    int dec_value = 0;

    // Initializing base value to 1, i.e 2^0
    int base = 1;

    int temp = num;
    while (temp) {
        int last_digit = temp % 10;
        temp = temp / 10;

        dec_value += last_digit * base;

        base = base * 2;
    }

    return dec_value;
}
void decToBinary(int n)
{
    // array to store binary number
    int binaryNum[32];
    // counter for binary array
    int i = 0;
    while (n > 0) {
        // storing remainder in binary array
        binaryNum[i] = n % 2;
        n = n / 2;
        i++;
    }
    // printing binary array in reverse order
```

final lab (cont)

```
>     for (int j = i - 1; j >= 0; j--)
       cout << binaryNum[j];
}
```

final lab

```
#include <iostream>
#include <st rin g>
#include <fs tre am>
/* The member function cin.ge tline() works with C strings (i.e. arrays of char)
whereas the free function std::g etl ine() works with C++ strings (i.e. std::s tring.)
You should not be using C strings at all when learning C++, which means
you should not be using cin.ge tli ne().*/
using namespace std;
void FindEx t(s td: :string paths[], int nPath, std::s tring extName, std::s tring results[], int &nRes)
{
nRes = 0;
for(int i = 0; i < nPath; i++)
{
string tmp = paths[ i].s ub str (pa ths [i].fi nd('.') + 1);
if(tmp == extName)
{
++nRes;
for(int j = 0; j < nRes; j++)
{
results[j] = paths[i];
}
}
}
}
void Find(s tring paths[], int nPath, string curDir, string results[], int &nRes)
{


}
int main()
{
```

final lab (cont)

```
> string paths[] = {"D:/Others/us/b.pdf", "C:/Users/c.txt", "C:/Users/adm/a.txt", "E:/temp/d.exe"};
string extName = "txt";
string results[1000];
int nRes;
FindExt(paths, 4, extName, results, nRes);
for(int i = 0; i < nRes; i++)
{
cout << results[i] << " ";
}
cout << endl;
cout << nRes;
return 0;
}
#include <iostream>
using namespace std;
void doMin(int a[][3], int m, int n)
{
int res[100][100] = {0};

int toaDoX[8] = {-1, -1, -1, 0, 0, 1, 1, 1};
int toaDoY[8] = {-1, 0, 1, -1, 1, -1, 0, 1};

for(int i = 0; i < m; ++i)
{
for(int j = 0; j < n; ++j)
{
if(a[i][j] == 1)
{
for(int k = 0; k < 8; ++k)
{
int newX = i + toaDoX[k];
int newY = j + toaDoY[k];
if(newX >= 0 && newY >= 0 && newX < m && newY < n)
{
++res[newX][newY];
```

final lab (cont)

```
> }
}
}
}
}
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
a[i][j] = res[i][j];
}
}
}
int main()
{
int a[3][3] = {
1, 0, 0,
0, 1, 0,
0, 0, 0};
int m = 3;
int n = 3;
doMin(a, m, n);
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
cout << a[i][j] << " ";
}
cout << endl;
}
return 0;
}
#include <iostream>
```

final lab (cont)

```
> #include <fstream>
using namespace std;
void ReadMap(int map[][100], int &m, int &n)
{
ifstream fin;
fin.open("map.txt");
if(!fin.is_open())
{
cout << "failed to open file!";
return;
}
fin >> m >> n;
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
fin >> map[i][j];
}
}
fin.close();
for(int i = 0; i < m; i++)
{
for(int j = 0; j < n; j++)
{
cout << map[i][j] << " ";
}
cout << endl;
}
}
int main()
{
int map[100][100];
int m, n;
```

final lab (cont)

```
> ReadMap(map, m, n);
return 0;
}
//các hàm
c_str: string to char
char array to string
// Demonstrates conversion
// from character array to string

#include <bits/stdc++.h>
using namespace std;

// converts character array
// to string and returns it
string convertToString(char* a, int size)
{
    int i;
    string s = "";
    for (i = 0; i < size; i++) {
        s = s + a[i];
    }
    return s;
}

// Driver code
int main()
{
    char a[] = { 'C', 'O', 'D', 'E' };
    char b[] = "geeksforgeeks";

    int a_size = sizeof(a) / sizeof(char);
    int b_size = sizeof(b) / sizeof(char);

    string s_a = convertToString(a, a_size);
    string s_b = convertToString(b, b_size);

    cout << s_a << endl;
    cout << s_b << endl;

    return 0;
```

final lab (cont)

```
> }
// C++ implementation of above approach
#include <bits/stdc++.h>
using namespace std;

#define N 4

// Function to rotate the matrix 90 degree clockwise
void rotate90Clockwise(int a[N][N])
{

    // Traverse each cycle
    for (int i = 0; i < N / 2; i++) {
        for (int j = i; j < N - i - 1; j++) {

            // Swap elements of each cycle
            // in clockwise direction
            int temp = a[i][j];
            a[i][j] = a[N - 1 - j][i];
            a[N - 1 - j][i] = a[N - 1 - i][N - 1 - j];
            a[N - 1 - i][N - 1 - j] = a[j][N - 1 - i];
            a[j][N - 1 - i] = temp;
        }
    }
}

// Function for print matrix
void printMatrix(int arr[N][N])
{
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++)
            cout << arr[i][j] << " ";
        cout << '\n';
    }
}

// Driver code
int main()
{
```

final lab (cont)

```
>    int arr[N][N] = { { 1, 2, 3, 4 },
                       { 5, 6, 7, 8 },
                       { 9, 10, 11, 12 },
                       { 13, 14, 15, 16 } };
    rotate90Clockwise(arr);
    printMatrix(arr);
    return 0;
}
//anticlockwise
void rotateMatrix(int mat[][N])
{
    // Consider all squares one by one
    for (int x = 0; x < N / 2; x++) {
        // Consider elements in group
        // of 4 in current square
        for (int y = x; y < N - x - 1; y++) {
            // Store current cell in
            // temp variable
            int temp = mat[x][y];

            // Move values from right to top
            mat[x][y] = mat[y][N - 1 - x];

            // Move values from bottom to right
            mat[y][N - 1 - x] = mat[N - 1 - x][N - 1 - y];

            // Move values from left to bottom
            mat[N - 1 - x][N - 1 - y] = mat[N - 1 - y][x];

            // Assign temp to left
            mat[N - 1 - y][x] = temp;
        }
    }
}

// Function to print the matrix
void displayMatrix(int mat[N][N])
{
```

final lab (cont)

```
>     for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            cout << mat[i][j] << " ";
        }
        cout << endl;
    }
    cout << endl;
}

/ Driver code /
int main()
{
    // Test Case 1
    int mat[N][N] = { { 1, 2, 3, 4 },
                      { 5, 6, 7, 8 },
                      { 9, 10, 11, 12 },
                      { 13, 14, 15, 16 } };

    // Function call
    rotateMatrix(mat);

    // Print rotated matrix
    displayMatrix(mat);

    return 0;
}
int binaryToDecimal(int n)
{
    int num = n;
    int dec_value = 0;

    // Initializing base value to 1, i.e 2^0
    int base = 1;

    int temp = num;
    while (temp) {
        int last_digit = temp % 10;
        temp = temp / 10;

        dec_value += last_digit * base;
```

final lab (cont)

```
>       base = base * 2;
   }

   return dec_value;
}
void decToBinary(int n)
{
   // array to store binary number
   int binaryNum[32];
   // counter for binary array
   int i = 0;
   while (n > 0) {
      // storing remainder in binary array
      binaryNum[i] = n % 2;
      n = n / 2;
      i++;
   }
   // printing binary array in reverse order
   for (int j = i - 1; j >= 0; j--)
      cout << binaryNum[j];
}
```

vd cua convert string to char

```
#include <io str eam>
#include <cs tri ng>
#include <st rin g>
int main()
{
      std ::s tring s = " clc -
3";
     char cstr[50];
      str cpy (cstr, s.c_st -
r());
      std ::cout << cstr;
      return 0;
}
```

phoneNum

phoneNum (cont)

phoneNum (cont)

```
> for(int i = 0; i < n; i++)
{
if(2021 - a[i].birthYear > 18)
{
ofst << a[i].name << " "; //bla bla
}
}
ofs.close();
}
bool soDep(char num[])
{
for(int i = 3; i < strlen(num) - 1; ++i)
{
if(num[i] > num[i + 1])
{
return false;
}
}
return true;
}
```

mò 2.0 array

```
#include <string>
#include <io str eam>
void readFi le( string filename,
Contact a[], int &n)
{
ifstream ifs;
ifs.op en( fil ena me.c _s -
tr());
int n = 0;
string s;
getlin e(i fs,s);
while( !if.eof())
{
getlin e(ifs, s, '|');
strcpy (a[ n].n ame, s.c_st -
r());
getlin e(if,s, '|');
a[n].b irt hYear = atoi(s.c_ -
str());
getlin e(ifs, s, '\n');
strcpy (a[ n].p ho neN umber,
s.c_st r());
n++;
}
ifs.cl ose();
```

```
> //A.I ver
int main() {
    std::ifstream ifs("input.txt");
    std::vector<MyStruct> a;
    std::string line;
    while (getline(ifs, line, '|')) {
        MyStruct temp;
        temp.name = line;
        a.push_back(temp);
    }
}
void chuan_hoa(char name[])
{
for(int i = 0; name[i] != '\0'; i++)
{
if(i == 0 || name[i - 1] == ' ')
{
name[i] == toupper(name[i]);
}
else
{
name[i] = tolower(name[i]);
}
}
}
void writeFile(string filename, Contact a[],
int n)
{
ofstream ofs;
ofs.open(filename.c_str())
{
ofs << "name|Birth|Phone";
}
```

```
void xoayphai(int a[], int n)
{
int tmp = a[n - 1];
for(int i = n - 2; i >= 0; i--)
{
a[i + 1] = a[i];
}
a[0] = tmp;
}
```

mò 2.0 array (cont)

> //khi bắt xoay k lần, để ý là nếu k = n thì sau n lần xoay đó thì vị trí các phần tử cũng quay về lúc ban đầu nên là khi cho k thì cập nhật lại k chia dư với n thôi, và xoay với số lần đã cập nhật đó, tại vì nó giống như đồng hồ á. làm vậy nó tối ưu hơn

```
int k;
k %= n;
for(int i = 0; i < k; i++)
{
xoayphai(a, n);
}
for(int x : a)
{
cout << x << " ";
}
void xoayphai(int a[], int n)
{
tmp = a[0];
for (int i = 1; i < n; i++)
{
a[i] = a[i + 1];
}
a[1] = tmp;
}
//mảng cộng dồn
//xây dựng mảng cộng dồn:
`
int f[n];
for (int i = 0; i < n; i++)
{
if(i == 0) f[0] = a[0];
```

mò 2.0 array (cont)

```
> else f[i] = f[i - 1] + a[i];
}
int q; cin >> q;
while (q--)
{
int l, r; cin >> l >> r;
if (l == 0) cout << f[r] << endl;
else cout << f[r] - f[l -1] << endl;
}
```

//Cho mảng số nguyên A[] gồm N phần tử, hãy tìm vị trí(bắt đầu từ 0) cuối cùng của giá trị nhỏ nhất trong mảng và vị trí đầu tiên của giá trị lớn nhất trong mảng. Tức là nếu có nhiều số có cùng giá trị nhỏ nhất bạn phải in ra ví trí cuối cùng, và có nhiều số có cùng giá trị lớn nhất trong mảng bạn phải in ra vị trí đầu tiên lớn nhất đó

```
int main() {
    int n;
    cin >> n;
    int a[n];
    for (int i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    int max = a[0];
    int min = a[0];
    int vtri_max = 0;
    int vtri_min = 0;
    for (int i = 0; i < n; i++)
    {
```

mò 2.0 array (cont)

```
>         if (a[i] <= min) //dùng dấu = để tìm ra thằng sau cùng
        {
            min = a[i];
            vtri_min = i;
        }
        if (a[i] > max) //không được dùng dấu = vì nếu nó bằng thì vtri_max sẽ bị cập nhật thành the current i, nói chung cái nào bự thì đừng bằng(?)
        {
            max = a[i];
            vtri_max = i;
        }
    }
    cout << vtri_min << " " << vtri_max;


    return 0;
}
```

//đề: Cho mảng số nguyên A[] gồm N phần tử, hãy liệt kê theo thứ tự xuất hiện các số thỏa mãn có ít nhất 1 số trái dấu với nó đứng cạnh nó.

- ý tưởng: ta thấy: chỉ cần a[i] *a[i + 1] < 0 hay a[i]* a[i -1] < 0 thì thỏa mãn

do số liền kề với i là i + 1 và i - 1. xét các trường hợp khi i là 0 và i là n - 1 và các trường hợp (bth) còn lại

```
for (int i = 0; i < n; i++)
    {
        if(i == 0)
        {
            if(a[0] * a[1] < 0) cout << a[0] << " ";
        }
```

mò 2.0 array (cont)

```
>       else if(i == n - 1)
      {
         if(a[n - 1] * a[n - 2] < 0) cout << a[n -
1] << " ";
      }
      else
      {
         if(a[i] a[i - 1] < 0 || a[i] a[i + 1] < 0)
cout << a[i] << " ";
      }
   }
`
```

ato đồ đó

```
atoi -> ASCII to integer.
atol -> ASCII to long.
atof -> ASCII to floating.
stoi -> string to integer.
stol -> string to long.
stoll -> string to long long.
stof -> string to float.
stod -> string to double.
stold -> string to long double.
```