Week 05

=====================================================

Problem 501:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, answer = 1001;

cin >> N >> M;

int\*\* massiv = new int\* [N];

for (int count = 0; count < N; count++)

massiv[count] = new int[M];

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

// find minimum element in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

if (answer > massiv[i][j])

answer = massiv[i][j];

cout << answer;

return 0;

}

=====================================================

Problem 502:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, number, d\_right, d\_left, i, j;

cin >> N;

int\*\* massiv = new int\* [N];

for (int count = 0; count < N; count++)

massiv[count] = new int[N];

d\_left = 0;

d\_right = 0;

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

cin >> massiv[i][j];

// find the sum diferent diagonals

for (i = 0; i < N; i++)

{

for (j = 0; j < N; j++)

{

if (i == j)

if (i + j == N - 1)

{

d\_left += massiv[i][j];

d\_right += massiv[i][j];

}

else

d\_left += massiv[i][j];

else if (i + j == N - 1)

d\_right += massiv[i][j];

}

}

cout << d\_left << " " << d\_right << endl;

return 0;

}

=====================================================

Problem 503:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, number, row, column, i, j;

cin >> N >> M;

int\*\* massiv = new int\* [N];

for (int count = 0; count < N; count++)

massiv[count] = new int[M];

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

// find index in array of number 0

for (i = 0; i < N; i++)

{

for (j = 0; j < M; j++)

{

if (massiv[i][j] == 0)

{

row = i;

column = j;

}

}

}

cout << row + 1 << " " << column + 1 << endl;

return 0;

}

=====================================================

Problem 504:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, number, row, column, i, j, answer;

cin >> N >> M;

int\*\* massiv = new int\* [N];

for (int count = 0; count < N; count++)

massiv[count] = new int[M];

bool x = false;

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

for (i = 0; i < N; i++)

{

for (j = 0; j < M; j++)

{

if (massiv[i][j] == -1)

{

row = i;

column = j;

x = true;

}

}

}

//if x equal false we output -1 -1 else output index of number -1

if (x == false)

cout << -1 << " " << -1 << endl;

else

cout << row + 1 << " " << column + 1 << endl;

return 0;

}

=====================================================

Problem 505:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and first ans second Two-dimensional dynamic array

int N, number, i, j, count;

cin >> N;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[N];

int\*\* massiv\_2 = new int\* [N];

for (count = 0; count < N; count++)

massiv\_2[count] = new int[N];

// input elements in first array

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

cin >> massiv[i][j];

// input elements in second array

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

cin >> massiv\_2[i][j];

//first array + second array

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

massiv[i][j] = massiv[i][j] + massiv\_2[i][j];

for (i = 0; i < N; i++)

{

for (j = 0; j < N; j++)

cout << massiv[i][j] << " ";

cout << endl;

}

return 0;

}

=====================================================

Problem 506:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, number, i, j, count;

cin >> N;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[N];

// input elements in array

bool x = true;

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

cin >> massiv[i][j];

// matrix symmetry check

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

if (massiv[i][j] != massiv[j][i])

{

x = false;

break;

}

if (x == true)

cout << "YES";

else

cout << "NO";

return 0;

}

=====================================================

Problem 507:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int answer, i, j, N, count;

cin >> N;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[N];

// input elements in array

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

cin >> massiv[i][j];

//find the sum of the numbers under the main diagonal

for (answer = i = 0; i < N; i++)

for (j = 0; j < N; j++)

if (j < i)

answer += massiv[i][j];

cout << answer << endl;

return 0;

}

=====================================================

Problem 508:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int answer, i, j, N, count;

cin >> N;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[N];

// input elements in array

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

cin >> massiv[i][j];

//find the sum of the numbers above the main diagonal

for (answer = i = 0; i < N; i++)

for (j = 0; j < N; j++)

if (j > i)

answer += massiv[i][j];

cout << answer << endl;

return 0;

}

=====================================================

Problem 509:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, count, total, index;

cin >> N >> M;

int\*\* massiv = new int\* [N + M];

for (count = 0; count < N + M; count++)

massiv[count] = new int[M + N];

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

// find the column number(index) that have maximum sum of it's elements.

total = -10000;

index = -1;

for (int i = 0; i < M; i++)

{

int answer = 0;

for (int j = 0; j < N; j++)

answer += massiv[j][i];

if (answer > total)

{

total = answer;

index = i;

}

}

cout << index + 1;

return 0;

}

=====================================================

Problem 510:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array and array where will be positive numbers

int N, M, count, x, massiv\_2[10000]{};

cin >> N >> M;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[M];

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

//if the number in the matrix is positive then add to the array

int z = 0;

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

if (massiv[i][j] > 0)

{

massiv\_2[z] = massiv[i][j];

z++;

}

// bubble sort

for (int i = 0; i < z; i++)

{

for (int j = 0; j < z; j++)

{

if (massiv\_2[i] <= massiv\_2[j])

{

x = massiv\_2[i];

massiv\_2[i] = massiv\_2[j];

massiv\_2[j] = x;

}

}

}

cout << z << endl;

// output positive numbers

for (int i = 0; i < N \* M; i++)

{

if (massiv\_2[i] == 0)

break;

else

cout << massiv\_2[i] << " ";

}

return 0;

=====================================================

Problem 511:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, count;

cin >> N;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[N];

//Fill matrix from N^2 to 0

int x = N \* N;

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

{

massiv[i][j] = x;

x--;

}

// output matrix

for (int i = 0; i < N; i++)

{

for (int j = 0; j < N; j++)

cout << massiv[j][i] << " ";

cout << endl;

}

return 0;

}

=====================================================

Problem 512:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, count;

cin >> N >> M;

int\*\* massiv = new int\* [N + M];

for (count = 0; count < N + M; count++)

massiv[count] = new int[M + N];

// input elements in array

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

cin >> massiv[i][j];

//rotate given matrix on 90 degree counterclockwise.

for (int i = M - 1; i >= 0; i--)

{

for (int j = 0; j < N; j++)

cout << massiv[j][i] << " ";

cout << endl;

}

return 0;

}

=====================================================

Problem 513:

=====================================================

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int i;

char name[80];

char list[100][100] = { "Tom", "555-3322","Mary", "555-8976", "Jon", "555-1037","Rachel", "555-1400","Sherry", "555-8873" };

//find a number, you enter the name. The number is displayed. Otherwise output "Not found"

cin >> name;

for (i = 0; i < 10; i += 2)

if (strcmp(name, list[i]) == 0) {

cout << list[i + 1];

break;

}

if (i == 10)

cout << "Not found";

return 0;

}

=====================================================

Problem 514:

=====================================================

#include <iostream>

using namespace std;

//create function which find sum of two numbers

long Pascal(int N, int K)

{

double total = 1;

for (int i = 1; i <= K; i++)

total = total \* (N - i + 1) / float(i);

return total;

}

int main()

{

int N;

cin >> N;

// output numbers

for (int j = 0; j < N; j++)

{

for (int i = 0; i <= j; i++)

cout << Pascal(j, i) << " ";

cout << endl;

}

return 0;

}

=====================================================

Problem 515:

=====================================================

#include <iostream>

using namespace std;

int main() {

// create and input variables

int N, k, index = 0;

cin >> N >> k;

//find the survivor among N people

for (int i = 2; i <= N; i++)

index = (index + k) % i;

cout << index + 1;

return 0;

}

=====================================================

Problem 516:

=====================================================

#include <iostream>

using namespace std;

int main()

{

//create variables and Two-dimensional dynamic array

int N, M, count;

cin >> N >> M;

int\*\* massiv = new int\* [N];

for (count = 0; count < N; count++)

massiv[count] = new int[M];

// Fill matrix from 0 to N^2

int x = 0;

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++)

{

massiv[i][j] = x;

x++;

}

//output matrix with numbers in a snake form

for (int i = 0; i < N; i++)

{

for (int j = 0; j < M; j++)

{

if ((i + 2) % 2 != 0)

cout << massiv[i][M - 1 - j] << " ";

else

cout << massiv[i][j] << " ";

}

cout << endl;

}

return 0;

}

=====================================================

Problem 517:

=====================================================

#include <iostream>

#include <math.h>

using namespace std;

int main()

{

//create variables and Two-dimensional array

int k, index, total\_row, total\_column;

int massiv[10][10];

// input elements in array

for (int i = 0; i < 10; i++)

for (int j = 0; j < 10; j++)

cin >> massiv[i][j];

//find maximum horizontal length

index = 0;

total\_row = 0;

k = 0;

for (int i = 0; i < 10; i++)

{

k = 0;

index = 0;

for (int j = 0; j < 10; j++)

{

if (massiv[i][j] == 0 && k < 10)

{

index++;

if (total\_row < index)

total\_row = index;

}

else

{

k += index;

index = 0;

}

}

}

//find maximum vertical length

index = 0;

total\_column = 0;

k = 0;

for (int i = 0; i < 10; i++)

{

k = 0;

index = 0;

for (int j = 0; j < 10; j++)

{

if (massiv[j][i] == 0 && k < 10)

{

index++;

if (total\_column < index)

total\_column = index;

}

else

{

k += index;

index = 0;

}

}

}

//output maximum varibale

cout << max(total\_column, total\_row);

return 0;

}

=====================================================

Problem 518:

=====================================================

const int SIZE = 20;

int masssiv[SIZE][SIZE], masssiv\_2[SIZE][SIZE];

int N, T;

// create a function that counts the number of neighbors

int possition(int x, int y)

{

int ans = 0;

(masssiv[x - 1][y] == 1) ? ans++ : ans;

(masssiv[x - 1][y - 1] == 1) ? ans++ : ans;

(masssiv[x][y - 1] == 1) ? ans++ : ans;

(masssiv[x + 1][y - 1] == 1) ? ans++ : ans;

(masssiv[x + 1][y] == 1) ? ans++ : ans;

(masssiv[x + 1][y + 1] == 1) ? ans++ : ans;

(masssiv[x][y + 1] == 1) ? ans++ : ans;

(masssiv[x - 1][y + 1] == 1) ? ans++ : ans;

return ans;

}

//create a function in which the values are written in massiv\_2 equal massiv

void copy()

{

for (int i = 0; i < N; ++i)

for (int j = 0; j < N; ++j)

masssiv[i][j] = masssiv\_2[i][j];

}

//create function OUTPUt

void output()

{

for (int i = 0; i < N; ++i)

{

for (int j = 0; j < N; ++j)

cout << masssiv[i][j] << " ";

cout << endl;

}

}

int main()

{

cin >> N >> T;

// input elements in array

for (int i = 0; i < N; ++i)

for (int j = 0; j < N; ++j)

{

cin >> masssiv[i][j];

masssiv\_2[i][j] = masssiv[i][j];

}

// change the matrix elements according to the conditions

int total\_possition = 0;

for (int k = 0; k < T; ++k)

{

for (int i = 0; i < N; ++i)

{

for (int j = 0; j < N; ++j)

{

total\_possition = possition(i, j);

if (masssiv[i][j] == 0)

(total\_possition == 3) ? masssiv\_2[i][j] = 1 : masssiv[i][j];

if (masssiv[i][j] == 1)

(total\_possition > 3 || total\_possition < 2) ? masssiv\_2[i][j] = 0 : masssiv[i][j];

}

}

copy();

}

output();

return 0;

}

=====================================================

Problem 519:

=====================================================

#include <iostream>

using namespace std;

bool massiv[50][50];

int main() {

int N;

cin >> N;

//We’ll fill the upper, lower and right edges with 1

for (int i = 0; i < N; i++)

{

massiv[0][i] = 1;

massiv[N - 1][i] = 1;

massiv[i][N - 1] = 1;

}

/\*

We have two parameters: width and height of unit-filled segment.

At each step, reduce the corresponding completed segment by and check that it is greater ,

or end. Throughout the decision, we keep two more parameters:

the coordinates of the point from which we begin the next step.

\*/

int height = N - 2, length = N - 2;

int pointx = N - 1, pointy = 0;

while (true)

{

for (int i = 0; i < height; i++)

massiv[pointx - i][pointy] = 1;

pointx -= height - 1;

height -= 2;

if(height <= 0)

break;

for (int i = 0; i < length; i++)

massiv[pointx][pointy + i] = 1;

pointy += length - 1;

length -= 2;

if (length <= 0)

break;

for (int i = 0; i < height; i++)

massiv[pointx + i][pointy] = 1;

pointx += height - 1;

height -= 2;

if (height <= 0)

break;

for (int i = 0; i < length; i++)

massiv[pointx][pointy - i] = 1;

pointy -= length - 1;

length -= 2;

if (length <= 0)

break;

}

massiv[N / 2][N / 2] = 0;

for (int i = 0; i < N; i++)

{

for (int j = 0; j < N; j++)

cout << massiv[i][j];

cout << endl;

}

return 0;

}