1 номер 1 уровень

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include <locale.h>

using namespace std;

#define \_CRT\_SECURE\_NO\_WARNINGS

#define maxLength 100

char\* removeChar(char\* inputString, char charToRemove);

int main()

{

setlocale(LC\_ALL, "Rus");

char firstStringBuffer[maxLength];

char secondStringBuffer[maxLength];

char\* firstString, \* secondString;

printf("Введите первую строку (максимальная длина строки - %d):\n", maxLength);

gets\_s(firstStringBuffer);

printf("Введите вторую строку (максимальная длина строки - %d):\n", maxLength);

gets\_s(secondStringBuffer);

firstString = (char\*)malloc(strlen(firstStringBuffer) + 1); /\* + 1 чтобы учитывался 0 в конце строки\*/

secondString = (char\*)malloc(strlen(secondStringBuffer) + 1);

strcpy\_s(firstString, strlen(firstStringBuffer) + 1, firstStringBuffer); /\*1 - куда, 2 - количество, 3 - откуда\*/

strcpy\_s(secondString, strlen(secondStringBuffer) + 1, secondStringBuffer);

printf("\nПервая введённая строка:\n%s\n", firstString);

printf("\nВторая введённая строка:\n%s\n", secondString);

// Удаление символов

char charToRemove;

printf("Введите символ для удаления: ");

scanf\_s(" %c", &charToRemove, 1);

char\* modifiedFirstString = removeChar(firstString, charToRemove);

char\* modifiedSecondString = removeChar(secondString, charToRemove);

// Определение, в какой строке цифр больше

int digitCountFirst = 0, digitCountSecond = 0;

for (int i = 0; i < strlen(modifiedFirstString); ++i) {

if (isdigit(modifiedFirstString[i])) { /\*проверка на число \*/

digitCountFirst++;

}

}

for (int i = 0; i < strlen(modifiedSecondString); ++i) {

if (isdigit(modifiedSecondString[i])) {

digitCountSecond++;

}

}

if (digitCountFirst > digitCountSecond) {

printf("\nИзменённая первая строка:\n%s\n", modifiedFirstString);

}

else {

printf("\nИзменённая вторая строка:\n%s\n", modifiedSecondString);

}

// Освобождение памяти

free(firstString);

free(secondString);

free(modifiedFirstString);

free(modifiedSecondString);

firstString = secondString = modifiedFirstString = modifiedSecondString = NULL;

return 0;

}

char\* removeChar(char\* inputString, char charToRemove)

{

int inputLength = strlen(inputString);

int charCountToRemove = 0;

// Подсчет количества символов для удаления

for (int i = 0; i < inputLength; ++i) {

if (inputString[i] == charToRemove) {

charCountToRemove++;

}

}

// Выделение памяти для новой строки без символов для удаления

char\* resultString = (char\*)malloc(inputLength - charCountToRemove + 1);

int resultIndex = 0;

// Копирование символов в новую строку, исключая символ для удаления

for (int i = 0; i < inputLength; ++i) {

if (inputString[i] != charToRemove) {

resultString[resultIndex++] = inputString[i];

}

}

resultString[resultIndex] = '\0';

return resultString;

}

2 номер 1 уровень

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <malloc.h>

using namespace std;

void printArray(float\*\* a, int N, int M)

{

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

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

printf("%5.1f", a[i][j]);

}

printf("\n");

}

printf("\n");

printf("\n");

}

float Verx(float\*\* a, int N, int M)

{

const int n2 = N / 2;

int i, j;

float summ = 0;

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

{

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

{

if (i < n2 && j < n2 && i <= j || i < n2 && j >= n2 && j < (N - i)) {

summ = summ + a[i][j];

}

}

}

return summ;

}

float Nizz(float\*\* a, int N, int M)

{

const int n2 = N / 2;

int i, j;

float summ = a[N - 1][M - 1];

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

{

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

{

if (i >= n2 && j < n2 && i >= (N - j - 1) || i >= n2 && j >= n2 && i >= j) {

summ = summ + a[i][j];

}

}

}

return summ;

}

int main()

{

setlocale(LC\_ALL, "Rus");

time\_t k;

srand(time(&k));

int M;

int N;

float\*\* a;

scanf("%d", &N);

scanf("%d", &M);

/\*printf("%d %d", N, M);\*/

a = (float\*\*)malloc(N \* sizeof(float\*));

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

a[i] = (float\*)calloc(M, sizeof(float));

}

int i, j;

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

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

/\*a[i][j] = 10.0;\*/

a[i][j] = 10 - (rand() % 200 / 10.0);

}

}

printArray(a, N, M);

float summ1 = Nizz(a, N, M);

float summ2 = Verx(a, N, M);

printf("%5.1f", summ2 + summ1);

return 0;

}

1 номер 2 уровень

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#include <locale.h>

using namespace std;

#define \_CRT\_SECURE\_NO\_WARNINGS

#define MAX\_STRING\_LENGTH 100

char\* removeChar(char\* inputString, char charToRemove);

int main() {

setlocale(LC\_ALL, "Rus");

char\*\* strings = NULL; // Указатель на указатель на строки

char input[MAX\_STRING\_LENGTH];

int numStrings = 0; // Количество введенных строк

int i;

// Ввод строк

while (strcmp(input, "ok") != 0) {

printf("Введите строку (для завершения введите 'ok'): ");

fgets(input, MAX\_STRING\_LENGTH, stdin);

input[strcspn(input, "\n")] = '\0'; // Удаление символа новой строки

if (strcmp(input, "ok") != 0) {

// Выделение памяти под новую строку

strings = (char\*\*)realloc(strings, (numStrings + 1) \* sizeof(char\*));

strings[numStrings] = (char\*)malloc((strlen(input) + 1) \* sizeof(char));

strcpy\_s(strings[numStrings],strlen( input)+1, input);

numStrings++;

}

}

// Печать введенных строк

printf("\nВведенные строки:\n");

for (i = 0; i < numStrings; i++) {

printf("%s\n", strings[i]);

}

for (int i = 0; i <(numStrings); ++i) {

for (int j = 0; j < strlen(strings[i]); j++) {

if (isdigit(strings[i][j])) { /\*проверка на число \*/

if ((int)strings[i][j]%2 == 0) {

printf(" %c", strings[i][j]\*2);

}

else if ((int)strings[i][j] % 2 == 1) {

char\* modifiedFirstString = removeChar(strings[i], strings[i][j]);

printf(" %s\n ", modifiedFirstString);

}

}

}

}

// Очистка памяти

for (i = 0; i < numStrings; i++) {

free(strings[i]);

}

free(strings);

return 0;

}

char\* removeChar(char\* inputString, char charToRemove)

{

int inputLength = strlen(inputString);

int charCountToRemove = 0;

// Подсчет количества символов для удаления

for (int i = 0; i < inputLength; ++i) {

if (inputString[i] == charToRemove) {

charCountToRemove++;

}

}

// Выделение памяти для новой строки без символов для удаления

char\* resultString = (char\*)malloc(inputLength - charCountToRemove + 1);

int resultIndex = 0;

// Копирование символов в новую строку, исключая символ для удаления

for (int i = 0; i < inputLength; ++i) {

if (inputString[i] != charToRemove) {

resultString[resultIndex++] = inputString[i];

}

}

resultString[resultIndex] = '\0';

return resultString;

}

2 номер 2 уровень

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <malloc.h>

using namespace std;

void printArray(float\*\* a, int N, int M)

{

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

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

printf("%5.1f", a[i][j]);

}

printf("\n");

}

printf("\n");

printf("\n");

}

float \*Verx(float\*\* a, int N, int M)

{

const int n2 = N / 2;

int i, j;

float\* min = &a[0][0];

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

{

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

{

if (i < n2 && j < n2 && i <= j || i < n2 && j >= n2 && j < (N - i)) {

if (\*min > a[i][j]) {

\*min = a[i][j];

}

}

}

}

return min;

}

float \*Nizz(float\*\* a, int N, int M)

{

const int n2 = N / 2;

int i, j;

float\* min = &a[N-1][M-1];

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

{

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

{

if (i >= n2 && j < n2 && i >= (N - j - 1) || i >= n2 && j >= n2 && i >= j) {

if (\*min > a[i][j]) {

\*min = a[i][j];

}

}

}

}

return min;

}

int main()

{

setlocale(LC\_ALL, "Rus");

time\_t k;

srand(time(&k));

int M;

int N;

float\*\* a;

printf("Введите количество строк - ");

scanf("%d", &N);

printf("Введите количество столбцов - ");

scanf("%d", &M);

a = (float\*\*)calloc(N, sizeof(float\*));

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

a[i] = (float\*)calloc(M, sizeof(float));

}

int i, j;

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

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

/\*a[i][j] = 10.0;\*/

a[i][j] = 10 - (rand() % 200 / 10.0);

}

}

printArray(a, N, M);

float \* min1 = Nizz(a, N, M);

float \* min2 = Verx(a, N, M);

printf("Минимальное значение: ");

if (\*min1 < \*min2) {

printf("%2.1f", \*min1);

}

else {

printf("%2.1f", \*min2);

}

return 0;

}