**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**"Национальный исследовательский университет**

**"Высшая школа экономики"**

Московский институт электроники и математики им. А.Н. Тихонова НИУ ВШЭ

Департамент компьютерной инженерии

**Курс: Алгоритмизация и программирование**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Раздел** | **Max оценка** | **Итог. Оценка 1** | **Итог. Оценка 2** | **Итог. Оценка 3** |
| Постановка | 0,5 |  |  |  |
| Метод | 1 |  |  |  |
| Спецификация | 0,5 |  |  |  |
| Алгоритм | 1,5 |  |  |  |
| Работа программы | 1 |  |  |  |
| Листинг | 0,5 |  |  |  |
| Тесты | 1 |  |  |  |
| Вопросы | 2 |  |  |  |
| Доп. задание | 2 |  |  |  |

**ОТЧЕТ**

**по лабораторной работе № 3**

**Студент: Иванилов Леонид Михайлович**

**Группа: БИВ231**

**Вариант: 143 (4, 9, 4)**

**Руководитель:\_\_\_\_\_\_\_\_**

**Оценка:\_\_\_\_\_\_\_\_**

**Дата сдачи:\_\_\_\_\_\_**

**Задания**

1. (4)Вычислить периметр треугольника с вершинами в точках

, по формуле:

2. Вычислить значение по формуле:

3. Дан целочисленный массив A[1:n]. Написать программу, включающую две функции с параметрами. В первой функции необходимо подсчитать количество повторений каждого элемента массива A. Вторая процедура решает следующую задачу: (4) Удалить из массива *A* все неповторяющиеся элементы путем сдвига (дополнительный массив не использовать).

**Постановка**

Дано:

1. – цел.
2. n – цел., A[1:n,1:n] – вещ.
3. n – цел., A[1:n] – цел.

Результат:

1. S – вещ.
2. F – вещ.
3. k – цел. A[1;k] – цел.

При:

2. n ∈ N, n ≤ lmax
3. n ∈ N, n ≤ lmax

Связь:

1. смотреть формулу

(при p e)

∀ t= C[t]=0, ∀ L=: a[L]=b[t], C[t]=C[t]+1

**Метод решения**

1. k = 0

**Внешняя спецификация**

Задание 1

Введите x1,x2,x3,y1,y2,y3

({<x1>}\*, {<x2>}\*, {<x3>}\*, {<y1>}\*, {<y2>}\*, {<y3>}\*) до <= INT\_MAX и до

При x1 == x2 == x3 и y1 == y2 == y3

Три точки совпадают

При (x3 - x1) / (x2 - x1) == (y3 - y1) / (y2 - y1)

Три точки находятся на одной прямой

иначе

Площадь треугольника равна: <<S>>

Задание 2

Введите n: количество строк и столбцов

{<n>}\* до n > 0 и n ≤ lmax и n

Введите матрицу <<n>> на <<n>>.

Для i = и j =

{<A[i][j]>}\* до A[i][j] ≤ INT\_MAX

Искомое F = <<F>>

Задание 3

Введите n: количество элементов массива

{<n>}\* до n > 0 и n ≤ lmax и n

Введите элементы массива

для i = {<A[i]>}\* до A[i] и A[i] ≤ INT\_MAX

Введённый массив

<<A[1]>>, <<A[2]>>, … <<A[n]>>

для i =

<<B[i]>> встречается <<C[i]>> раз

При n = 0

Полное удаление

Иначе

При flag = 0

Массив не изменился

Иначе

Все повторяющиеся элементы

<<A[1]>>, <<A[1]>>, ….., <<A[k]>>

**Алгоритм на псевдокоде**

Задание 1

Алг “Функция V”

Вход x, y – цел.

Выход res – цел.

Нач

res := x – y

кон

Алг “основная функция”

Нач

Вывод (<<введите x1, x2, x3, y1, y2, y3>>)

Ввод(x1,x2,x3,y1,y2,y3)

Если x1 = x2 = x3 и y1=y2=y3 то

Вывод(<<Три точки совпадают>>)

Иначе если ((x3 - x1) / (x2 - x1) = (y3 - y1) / (y2 - y1) то

Вывод(<<точки находятся на одной прямой>>)

Иначе

s := ½ \* |V(x1, x3) \* V(y2, y3) – V(x2, x3) \* V(y1, y3)|

вывод(<<Искомое значение S = >>, s)

все

кон

Задание 2

Алг “vv\_matr”

Вход -

Выход a[lmax][lmax], n

нач

вывод(<<Введите число строк и столбцов матрицы n: >>)

ввод(n)

вывод(<<Введите элементы матрицы из >>, n , <<строк и >>, n ,<<столбцов:>>)

цикл от i:=0 до n-1

цикл от j:=0 до n-1

ввод(a[i][j])

кц

кц

кон

Алг “f ” // вычисление F

Вход a[lmax][lmax], n

Выход min

Нач

Min := INT\_MAX

Цикл от j := 0 до n-1

Max := a[0][j] – a[1][j]

Цикл от i := 1 до n – 2

если |a[i][j] – a[i+1][j]| > |max| то

max := a[i][j] – a[i+1][j]

все

кц

если min > max то

min := max

кц

кц

кон

Алг “Основная функция”

Нач

Vv\_matr()

F := f(a[lmax][lmax], n)

Вывод(<<Искомое F = >>, F)

Кон

Задание 3

Алг “Ввод\_массива”

Вход -

Выход n, A[lmax][lmax]

Нач

вывод(<<Введите количество элементов n: >>)

ввод(n)

вывод(<<Введите элементы массива из >> , n, << элементов>>)

цикл от i:=0 до n-1

ввод(A[i])

кц

кон

Алг “del”

Вход A[], n

Выход A[], n

Нач

flag := 0

Цикл от i := 0 до n-1

cnt := 0

j := 0

цикл-пока j < n и cnt 2

если A[i] = A[j] то

cnt := cnt + 1

все

j := j + 1

кц

если cnt = 1 то

цикл от k := i до n – 1

A[k] := A[k+1]

кц

n := n – 1

flag := 1

i := i – 1

все

кц

кон

Алг “default\_func”

Вход A[], n

Выход -

Нач

K := 0

цикл от i := 0 до n – 1

p := -1

цикл от j := 0 до k – 1

если B[j] = A[i] то

p := j

все

кц

если p = -1 то

B[k] := A[i]

C[k] := 1

K := k + 1

иначе

С[p] := C[p] + 1

все

кц

цикл от j := 0 до k-1

Вывод(B[j], <<встречается>>, C[j], <<раз>>)

Кц

кон

Алгоритм “Вывод\_массива”

Вход a[n], n

выход

Нач

Вывод(<<Введённый массив>>)

Цикл от i:= 0 до n – 1

Вывод(a[i])

Кц

Кон

Алгоритм “Конечный\_вывод”

Вход a[n], n, flag

Выход -

Нач

Если n = 0 то

Вывод(<<Полное удаление>>)

Если flag = 0 то

Вывод(<<Массив без изменений>>)

Иначе

Вывод(<<Все повторяющиеся элементы>>)

Цикл от i:= 0 до n – 1

Вывод(a[i])

Кц

все

Кон

Алгоритм “Основная\_функция”

Нач

Ввод массива()

Вывод массива(a[n], n)

Default\_func(A[], n)

Del(A[], n)

Конечный\_вывод(A[n], n, flag)

кон

Листинг программы

Задание 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <limits.h>

#include <locale.h>

#include <string.h>

**int** v(**int** x, **int** y){

**int** s;

s = x - y;

**return** s;

}

**int** input(){

**int** num;

**float** k1;

num = scanf("%f", &k1);

**while**(k1 != (**int**)k1 || k1 > INT\_MAX || num != 1 ){

printf("Введите заново\n");

num = scanf("%f", &k1);

}

**return** (**int**)k1;

}

**int** main(**void**){

printf("Введите x1\n");

**int** x1 = input();

printf("Введите x2\n");

**int** x2 = input();

printf("Введите x3\n");

**int** x3 = input();

printf("Введите y1\n");

**int** y1 = input();

printf("Введите y2\n");

**int** y2 = input();

printf("Введите y3\n");

**int** y3 = input();

**if** (x1 == x2 == x3 && y1 == y2 == y3){

printf("Три точки совпадают\n");

}

**else** **if** ((x3 - x1) / (x2 - x1) == (y3 - y1) / (y2 - y1)){

printf("Три точки находятся на одной прямой.\n");

}

**else**{

**float** ss;

ss = 0.5 \* abs((v(x1, x3) \* v(y2, y3)) - (v(x2, x3) \* v(y1, y3)));

**if** ((**int**)ss != 0)

printf("Площадь равна: 10.5%f\n", ss);

}

**return** 0;

}

Задание 2

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <limits.h>

#include <locale.h>

#define lmax 20

**void** vv\_matrix(**float** a[][lmax], **int** \*n)

{

setlocale(LC\_ALL, "Rus");

**float** k1;

**int** n1, i, j, num, p;

**do**

{

printf("Введите n: количество строк и столбцов\n");

p=scanf("%f",&k1);

**while**(getchar()!='\n');

}**while**(p != 1||k1 != (**int**)k1 || k1 > lmax);

n1 = (**int**)k1;

\*n=n1;

printf("Введите матрицу %d на %d \n",\*n, \*n);

**for**(i=0; i<\*n; i++)

{

**for**(j=0; j<\*n; j++)

{

num = scanf("%f", &k1);

**while**(num != 1 ){

printf("Введите заново\n");

num = scanf("%f", &k1);

**while**(getchar()!='\n');

}

a[i][j] = k1;

}

}

}

**float** S(**int** n, **float** a[][lmax])

{

**float** \*i, \*j;

**float** min, max;

min=INT\_MAX;

**for**(j=\*a;j<\*a+n;j++)

{

max = \*j - \*(j+lmax);

**for**(i=j+lmax;i<a[n-1];i+=lmax)

**if**(fabs(\*i - \*(i+lmax)) > fabs(max))

max= \*i - \*(i+lmax);

**if**(max<min)

min=max;

}

**return** min;

}

**int** main(**void**)

{

**float** a[lmax][lmax], F;

**int** n;

vv\_matrix(a,&n);

F=S(n,a);

printf("Искомое F = %10.5f\n",F);

**return** 0;

}

Задание 3

#include <stdio.h>

#include <stdlib.h>

#define lmax 10

**void** input(**int** \*A, **int** \*n)

{

printf("Введите количество элементов массива:\n");

**int** num, n1;

**float** k1;

num = scanf("%f", &k1);

**while**(k1 != (**int**)k1 || k1 < 0 || num != 1 || k1 > lmax ){

printf("Введите заново\n");

num = scanf("%f", &k1);

}

n1 = (**int**)k1;

\*n=n1;

printf("Введите %d элементов массива:\n",\*n);

**for**( **int** i=0; i<\*n; i++){

num = scanf("%f", &k1);

**while**(k1 != (**int**)k1 || num != 1 ){

printf("Введите заново\n");

num = scanf("%f", &k1);

**while**(getchar()!='\n');

}

A[i] = (**int**)k1;

}

}

**void** output(**int** \*A, **int** n)

{

printf("Введённый массив\n");

**for**(**int** i=0;i<n;i++)

printf("%d ", \*(A+i));

printf("\n");

}

**void** output1(**int** \*A, **int** n, **int** flag){

**if** (n==0) {

printf("Все элементы удалены\n");

}

**else** **if** (flag == 0){

printf("Удаления нет");

}

**else** {

printf("Все повторяющиеся элементы:\n");

**for**(**int** i = 0; i < n; i++)

printf("%d ",A[i]);

}

}

**void** default\_func(**int** \*A, **int** n)

{

**int** B[lmax], C[lmax];

**int** k=0;

**for** (**int** i=0; i<n; i++){

**int** p=-1;

**for** (**int** j=0; j<k; j++){

**if** (B[j]==A[i]){

p=j;

}

}

**if** (p==-1){

B[k] = A[i];

C[k] = 1;

k++;

}

**else**{

C[p]++;

}

}

**for** (**int** j=0; j<k; j++)

printf("%d встречается %d раз\n", B[j], C[j]);

}

**void** del(**int** \*A, **int** \*n, **int** \*flag) {

**for** (**int** i = 0; i < \*n; ++i) {

**int** cnt = 0;

**for** (**int** j = 0; j < \*n && cnt != 2; ++j) {

**if** (A[i] == A[j]) {

cnt++;

}

}

**if** (cnt == 1) {

**for** (**int** k = i; k < \*n; ++k) {

A[k] = A[k+1];

}

(\*n)--;

\*flag = 1;

i--;

}

}

}

**int** main(**void**) {

**int** A[lmax];

**int** n, flag;

input(A, &n);

output(A, n);

default\_func(A, n);

del(A, &n, &flag);

output1(A, n, flag);

**return** 0;

}

Тестовые данные

Задание 1

|  |  |  |
| --- | --- | --- |
| № | Тестовые данные | Вывод программы |
| 1 | X1, x2,x3,y1,y2,y3 = 1 | Три точки совпадают |
| 2 | x1 = 1, x2 = 2, x3 = 3, y1 = 1, y2 = 2, y3 = 3 | Три точки находятся на одной прямой |
| 3 | x1 = 1, x2 = 5, x3 = 7, y1 = 3, y2 = 2, y3 = 9 | Площадь равна: 15 |

Задание 2

|  |  |  |
| --- | --- | --- |
| № | Тестовые данные | Вывод программы |
| 1 | A = {1, 2, 3  1, 2, 3  1, 2, 3} | F = 0.00000 |
| 2 | A = {1, 1, 1  1, 1, 1  1, 1, 9} | F = -8.00000 |
| 3 | A = {3, 4, 5.6  3.4, 5, 6.8  3, 5, 7} | F = -1.20000 |

Задание 3

|  |  |  |
| --- | --- | --- |
| № | Тестовые данные | Вывод программы |
| 1 | A = {1, 1, 1, 1} | Введённый массив  1 1 1 1  1 встречается 4 раз  Удаления нет |
| 2 | A = {1, 2, 1, 1} | Введённый массив  1 2 1 1  1 встречается 3 раз  2 встречается 1 раз  Все повторяющиеся элементы:  1 1 1 |
| 3 | A = {1, 2, 3, 4} | Введённый массив  1 2 3 4  1 встречается 1 раз  2 встречается 1 раз  3 встречается 1 раз  4 встречается 1 раз  Все элементы удалены |

Дополнительное Задание

#include <stdio.h>

#include <stdlib.h>

#define lmax 20

**void** input(**int** \*A, **int** \*n)

{

printf("Введите количество элементов массива:\n");

**int** num, n1;

**float** k1;

num = scanf("%f", &k1);

**while**(k1 != (**int**)k1 || k1 < 0 || num != 1 || k1 > lmax ){

printf("Введите заново\n");

**while**(getchar()!='\n');

num = scanf("%f", &k1);

}

n1 = (**int**)k1;

\*n=n1;

printf("Введите %d элементов массива:\n",\*n);

**for**( **int** i=0; i<\*n; i++){

num = scanf("%f", &k1);

**while**(k1 != (**int**)k1 || num != 1 ){

printf("Введите заново\n");

num = scanf("%f", &k1);

**while**(getchar()!='\n');

}

A[i] = (**int**)k1;

}

}

**void** puz(**int** \*mass, **int** n){

**int** tmp;

**int** noSwap;

**for** (**int** i = n - 1; i >= 0; i--)

{

noSwap = 1;

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

{

**if** (mass[j] > mass[j + 1])

{

tmp = mass[j];

mass[j] = mass[j + 1];

mass[j + 1] = tmp;

noSwap = 0;

}

}

**if** (noSwap == 1)

**break**;

}

}

**void** Z(**int** A[], **int** n){

**int** maxx = 0;

**for** (**int** i = 0; i < n; i++){

**int** temp = 0;

**for** (**int** j = 0; j < n; j++){

**if** (A[i] == A[j]){

temp++;

}

}

**if** (temp > maxx){

maxx = temp;

}

}

// Нашли наиб кол-во чисел

**for** (**int** i = 0; i < n; i++){ // проходим по каждому элементу

**int** temp1 = 0;

**for** (**int** j = 0; j < n; j++){

**if**(A[i] == A[j]){

temp1++;

}// считываем кол-во A[i] чисел

}

**if** (temp1 == maxx)

{

printf("%d ", A[i]);

i = i + temp1-1;

}

**else**{

**if** (n < (i + temp1 - 1))

{

printf(" конец ");

**break**;

}

**else**{

i = i + temp1 - 1;

}

}

}

}

**int** main(){

**int** flag = 0;

**while** (flag != 3){

**int** A[lmax], n;

input(A, &n);

puz(A, n);

Z(A, n);

printf("\n");

}

**return** 0;

}

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание