***מערך חד מימדי***

**/\***

**פונקציה רקוסיבית מורידה את המספר האחרון והראשון והמספר ומדפיסה**

**אם כמות הספרות במספר הוא זוגי היא תדפיס בסוף 0**

**דוגמא אי זוגי :**

**15369**

**536**

**3**

**דוגמא זוגי:**

**123456**

**2345**

**34**

**0**

**\*/**

void main()

{

long num = 15369;

func(num);

}

long int func(int long num)

{

int i = 0;

long int high, low, mul, temp = num;

printf("%d\n", temp);

if ((num<10) && (num >= 0))

return num;

mul = 1;

while (temp /= 10)

{

i++;

mul \*= 10;

}

high = num / mul;

low = num % 10;

temp = (num - high\*mul) / 10;

return (high + low\*mul + func(temp) \* 10);

}

int array\_symtri(int v[N]) / בודק אם המערך סימטרי 1, 2, 3, 2, 1 /

{

int i, j, flag = 0;

for (i = 0, j = N - 1; i != j; i++, j--)

{

if (v[i] == v[j])

{

flag = 1;

}

else

{

flag = 0;

}

}

return(flag);

**/\*פונקציה הבודקת אם הערכים שבמערך הינם סכום של המספרים שלפניהם**

**כלומר 2,4,6,10,16 הערכים הלו עונים להגדרה .**

**\*/**

int sum\_of\_two\_numbers\_f(int num1, int num2, int num3)

{

int flag = 1;

if (num1 + num2 != num3)

{

flag = 0;

}

return(flag);

}

void main()

{

int S[N], i, flag = 1, d;

input\_array\_f(S);

for (i = 0;i < N - 2;i++)

{

d = (sum\_of\_two\_numbers\_f(S[i], S[i + 1], S[i + 2]));

if (d != 1)

{

flag = 0;

break;

}

}

if (flag==1)

{

printf("ok\n");

}

else {

printf("no\n");

}

}

void insertionSortRecursive(int arr[], int n)//פונקציה רקרוסיבית לסידור מערך

{

// בדיקה אם המערך הוא תא אחד

if (n <= 1)

return;

insertionSortRecursive(arr, n - 1);

int last = arr[n - 1];

int j = n - 2;

while (j >= 0 && arr[j] > last)

{

arr[j + 1] = arr[j];

j--;

}

arr[j + 1] = last;

}

**/\*פונקציה שמקבלת שני מערכים כפרמטרים**

**הפונקציה תבצע בדיקה על שני המערכים**

**הבדיקה תבדוק אילו ערכים במערך הראשון אינם מופיעים במערך השני וההפך ותדפיס אותם**

**במידה ויש שני מספרים אותו דבר הפונקציה תדפיס פעם אחת את אותו מספר \*/**

#define N 20

int twoarrs(int a[N], int b[N])

{

int c[N] = { 0 };

int i, j, counter = 0, temp1 = 0, temp2 = 0, k = 0;;

for (i = 0; i <= N - 1; i++)

{

temp1 = a[i];

for (j = 0; j <= N - 1; j++)

{

temp2 = b[j];

if (temp1 == temp2)

{

counter = 0;

break;

}

counter++;

}

if (counter == N)

{

c[k] = temp1;

k++;

counter = 0;

}

}

for (i = 0, counter = 0, temp1 = 0, temp2 = 0; i <= N - 1; i++)

{

temp1 = b[i];

for (j = 0; j <=N - 1; j++)

{temp2 = a[j];

if (temp1 == temp2)

{

counter = 0;

break;

}

counter++;

}

if (counter == N)

{

c[k] = temp1;

k++;

counter = 0;

}

}

for (i = 0; c[i] !=0; i++)

{

if (c[i + 1] != 0 && c[i] != c[i + 1])

{

puts("the number is:");

printf("%d\n", c[i]);

}

}

}

void main()

{

int A[N] = { 1, 2, 2, 3, 4, 6, 2, 7, 8, 9, 9, 9, 9, 9, 2, 3, 4, 6, 2, 9 };

int B[N] = { 2, 2, 2, 3, 4, 5, 7, 8, 9, 7, 12, 23, 4, 5, 5, 5, 5, 4, 4, 7 };

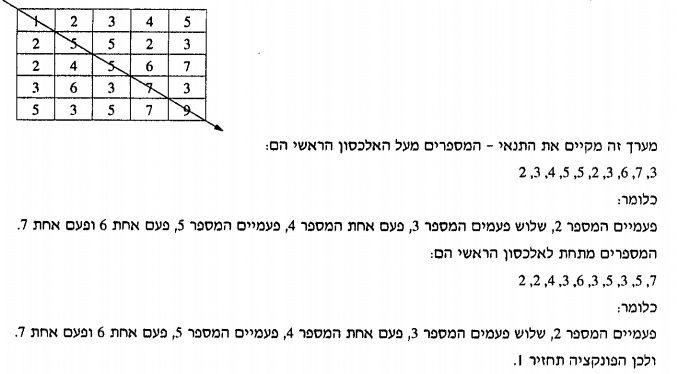
twoarrs(A,B);

}

***מערך דו מימד***

פונקציה המקבלת מערך דו ממדי

הפונקציה בודקת אם ערכי אלכסון עליון נמצאים באלכסון התחתון לא משנה המיקום



\*/

#include <stdio.h>

#include <stdlib.h>

#define N 5

int func(int mat[N][N]);

void main()

{

int valid, mat[N][N] = { { 1,2,3,4,5 },

{ 2,5,5,2,3 },

{ 2,4,5,6,7 },

{ 3,6,3,7,3 },

{ 5,3,5,7,9 } };

valid = func(mat);

if (valid == 1)

puts("ok");

else

puts("not ok.");

}

int func(int mat[N][N])

{

int i, j, flag = 1, max = 0;

int \*over\_main\_diagonal, \*below\_main\_diagonal;

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

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

if (max < mat[i][j])

max = mat[i][j]; /\*שמירה של המספר הגבוהה ביותר במטריצה\*/

over\_main\_diagonal = (int\*)calloc(max, sizeof(int)); /\*הקצאת מערך מונים על פי המספר הגבוהה ביותר שהתקבל קודם לכן\*/

below\_main\_diagonal = (int\*)calloc(max, sizeof(int));

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

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

{

if (i < j) /\*במידה ומעל האלכסון הראשי\*/

over\_main\_diagonal[mat[i][j]]++; /\*העלאת המונה בתא המתאים\*/

if (i > j) /\*במידה ומתחת לאלכסון הראשי\*/

below\_main\_diagonal[mat[i][j]]++; /\*העלאת המונה בתא המתאים\*/

}

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

if (over\_main\_diagonal[i] != below\_main\_diagonal[i]) /\*במידה ומערכי המונים אינם שווים - איפוס הדגל\*/

flag = 0;

if (flag == 1)

return (1);

else

return (0);

}

/\*פונקציה שמחשבת את סכום היקף המערך הדו מימדי הפונקציה מקבלת מערך ואת מספר היקף לחישוב \*/

/\*הפונקציה בודקת אם מספר ההיקף תקין אם לא מחזירה 0 אם כן היא מחשבת את העיקף\*/

/\*דוגמא במערך 6 על 6 יש 3 עיקופים \*/

int scope\_sum\_matrix(int mat[N][N], int \*per)

{

int i, j, max = N, hekef = 0, sum = 0;

if (N % 2 == 0)//בודק אם המטריצה זוגית או אי זוגית

{

hekef = N / 2;//בודק כמה הקפים יש לה

}

else

{

hekef = N / 2 + 1;//בודק כמה הקפים יש לה

}

if (\*per > hekef || \*per < 0)//אם מספר היקפים לא חוקי הפונקציה תחזיר 0

{

return 0;

}

for (i = \*per; i < N - \*per; i++)

{

for (j = \*per; j < N - \*per; j++)

{

if (i == \*per || i == N - \*per - 1 || j == \*per || j == N - \*per - 1)

{

sum += mat[i][j];

}

}

}

return sum;

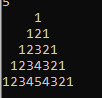
}

void triangle(int num)//פונקציה שמדפיסה משולש פולינדרום

{

int i = 0, j = 0, temp;

for (i = 1; i <= num; i++)

 {

for (j = i; j<num; j++)

printf(" ");

for (j = 1; j <= i; j++)

printf("%d", j);

for (j = i - 1; j >= 1; j--)

printf("%d", j);

printf("\n");

}

}

חפשו את כל המופעים של המילה הנתונה ***word*** במערך הדו-ממדי של תווים ***matr*** (ללא שימוש בפונקציות ספריה). המילה יכולה להיות מאוזנת או מאונכת!

**פתרון:**

#include <stdio.h>

#define ROW 5

#define COL 5

#define WORDSIZE 3

void main()

{

char matr[ROW][COL]={{'r','v','o','q','w'},

{'a','h','s','x','l'}, {'n','k','s','d','m'},

{'r','a','n','j','r'},

{'d','k','u','c','a'}};

char word[WORDSIZE+1]="ran";// השאר מקום לתו '0/'

int i,j,k,l;

// חפש מילים אופקיות (לאורך השורות):

for(i=0; i<ROW; i++) // סרוק את השורות.

for(j=0; j<=COL-WORDSIZE; j++) // סרוק את העמודות

{

for(k=j, l=0; l<WORDSIZE && matr[i][k]==word[l]; k++, l++);// סרוק את המילה // אם הוא שם

if(l==WORDSIZE) // בדוק אם כל המילה היתה

printf("The word was found horizontally!!! Its coordinates are: x1=%d, y1=%d, x2=%d, y2=%d\n", i, k-WORDSIZE,i,k-1);

}

// חפש מילים אנכיות (לאורך העמודות):

for(i=0; i<COL; i++) // סרוק את העמודות:

for(j=0; j<=ROW-WORDSIZE; j++) // סרוק את השורות

{

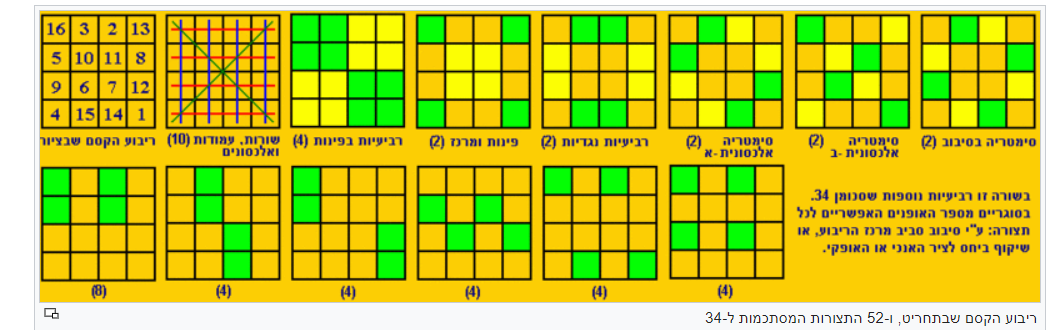
for(k=j, l=0;l<WORDSIZE && matr[k][i]==word[l]; k++, l++);

if(l==WORDSIZE)

printf("The word was found vertically!!! Its coordinates are: x1=%d, y1=%d, x2=%d, y2=%d\n", k-WORDSIZE,i,k-1,i);

}

}

כתבו תוכנית הממלאת ומדפיסה מערך דו-מימדי בגודל 5x5 כריבוע קסם,

**פתרון:**

#include <stdio.h>

#define N 5

void main()

{

int magic[N][N] = { 0 }, i, j, count;

for (count = 1, i = 0, j = N / 2; count <= N\*N; count++)

{

if (i < 0)

{

i += N;

}

if (j == N)

{

j -= N;

}

while (magic[i][j]) // כאשר נתקלים בריבוע שכבר מלא:

{

i++;

j--;

if (i == N)

{

i -= N;

}

if (j < 0)

{

j += N;

}

i++;

if (i == N) i -= N;

}

magic[i--][j++] = count;

}

הפונקציה תמצא את העמודה שבה הסכום המתואר הוא הגדול ביותר

ותחזיר את הערך הזה

האיבר שבשורה הראשונה אינו מסוכם הוא קאונטר של כמה מספרים נסכום

הפונקציה תמשיך ותסכם איברים באופן מעגלי

אם המספר הראשון בעמודה גדול מ( )

#include<stdio.h>

#define N 5

int sum(int arry[N][N])

{

int i, j, count = 0, temp = 0, sum = 0;

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

{

for ( i = 1; count != arry[0][j]; i++)

{

if (i == N && arry[0][j] > N - 1)

{

i = 1;

}

temp = temp + arry[i][j];

count++;

}

if (temp >= sum)

{

sum = temp;

}

temp = 0;

count = 0;

}

return(sum);

}

void main()

{

int i, j, max, arry[N][N] = { {6,7,4,3,8},

{1,1,2,1,1},

{2,1,2,1,2},

{3,1,1,2,1},

{2,1,1,8,2} };

max = sum(arry);

printf("the max is = %d\n", max);

}