גיליון תשובות- תרגיל 2 – מבני נתונים (10117)

|  |  |
| --- | --- |
| שם משפחה: משה  שם פרטי : בר  מס' ת.ז. : 206921777  המרצה : איציק גנות | שם משפחה: למפרט  שם פרטי : גבריאל  מס' ת.ז. : 206215311  המרצה : איציק גנות |

**שאלה מספר 1** סמן ב- X במשבצת המתאימה את התשובה הנכונה.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | ת ש ו ב ה | | | |
| שאלה |  | 1 | 2 | 3 | 4 |
| 1. | א |  |  |  | x |
|  | ב | x |  |  |  |
|  | ג |  | x |  |  |
|  | ד | x |  |  |  |
|  | ה |  | x |  |  |
|  | ו |  |  | x |  |
|  | ז | x |  |  |  |
|  | ח |  | x |  |  |
|  | ט |  |  | x |  |
|  | י |  |  | x |  |
|  | י"א |  |  |  | x |
|  | י"ב |  | x |  |  |
|  | י"ג |  |  | x |  |
|  | י"ד | x |  |  |  |
|  | ט"ו |  | x |  |  |
|  | ט"ז |  |  | x |  |

**שאלה מספר 2** סמן ב- X במשבצת המתאימה את התשובה הנכונה.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | ת ש ו ב ה | | | |
| שאלה |  | 1 | 2 | 3 | 4 |
| 2. | א | X |  |  |  |
|  | ב |  | X |  |  |
|  | ג |  |  | X |  |
|  | ד |  | X |  |  |
|  | ה |  |  |  | X |
|  | ו |  |  |  | X |
|  | ז | X |  |  |  |
|  | ח |  | X |  |  |
|  | ט |  |  | X |  |
|  | י |  |  |  | X |
|  | י"א |  |  | X |  |
|  | י"ב |  |  | X |  |

## שאלה מס' 3

בעבור כל סעיף ציין אם הטענה נכונה או לא. כמו כן נמק או הפרך את קביעתך.

א. **לא נכון.**

ב. **לא נכון.**

לא קיים שום קבוע כך ש

ג. **לא נכון**

ד. **לא נכון.**

ה. **לא נכון.**

ו. **לא נכון.**

## 

## 

## שאלה מס' 4

**המיון לפי סדר יורד הוא:**

נימוק לסידור המקיים את הדרישה לכל זוג עוקב:

## שאלה מס' 5:

בעבור כל סעיף – ראשית יש לקבוע את סדר הגודל של T(n) ולאחר מכן הוכחה בעבור קביעת סדרי הגודל:

## שאלה מס' 6

## שאלה מס' 7

SimpleCheck(int A[], int start, int end)

{

if (A[end] - A[start] > end - start)

return 1;

else

return 0;

}

SolveB(int A[], int start,int end)

{

if (start+1 == end)

return A[start] + 1;

int middle = (start + end) /2;

if (SimpleCheck(A, start, middle) == 1)

return SolveB(A, start, middle);

else

return SolveB(A, middle, end);

}

SolveC(int A[],int start,int end)

{

if (end < start)

return 0;

int m = (end + start) / 2;

if (A[m] == m)

return 1;

if (A[m] > m)

return SolveC(A, start, m - 1);

else

return SolveC(A, m + 1, end);

}

C is in log(n) like binary search.

## שאלה מס' 8

##### Matrix struct

#include <stdio.h>

#define SIZE 3

void main()

{

int a1[SIZE][SIZE] = { {0,1,1},{1,1,1},{0,1,0} };

int a2[SIZE][SIZE] = { {0,1,1},{1,1,1},{0,1,0} };

Init(SIZE, a1);

Flip(a1, 1, 1);

int result = DNF(a2);

printf("%d", result);

}

void Init(int size, int A[SIZE][SIZE])

{

for (int row = 0; row < size; row++)

for (int col = 0; col < SIZE; col++)

A[row][col] = 1;

}

void Flip(int a[SIZE][SIZE], int row, int col)

{

if (a[row][col] == 0)

a[row][col] = 1;

else

a[row][col] = 0;

}

int DNF(int a[SIZE][SIZE])

{

for (int row = 0; row < SIZE; row++)

{

for (int col = 0; col < SIZE; col++)

{

if (a[row][col] == 0)

break;

else if (col == SIZE - 1)

return 1;

}

}

return 0;

}

## שאלה מס' 9