פרויקט מסכם בC

מגיש: אור יאנקו

כיתה: יג1

מרצים:ד"ר ראובן חוטובלי וגב' יונה סעדיה

תאריך הגשה:

8.12.2020

**תוכן עניינים**

נושא עמוד

שער. .................................................................................................1

תוכן עניינים..................... ...................................................................2

שאלה 1-"קרב צוללות"..........................................................................3

שאלה 2 -מלבן ימני תחתון.....................................................................8

שאלה 3- מיון רשימה לפי זוגיות.............................................................11

שאלה 4- קיצורים................................................................................15

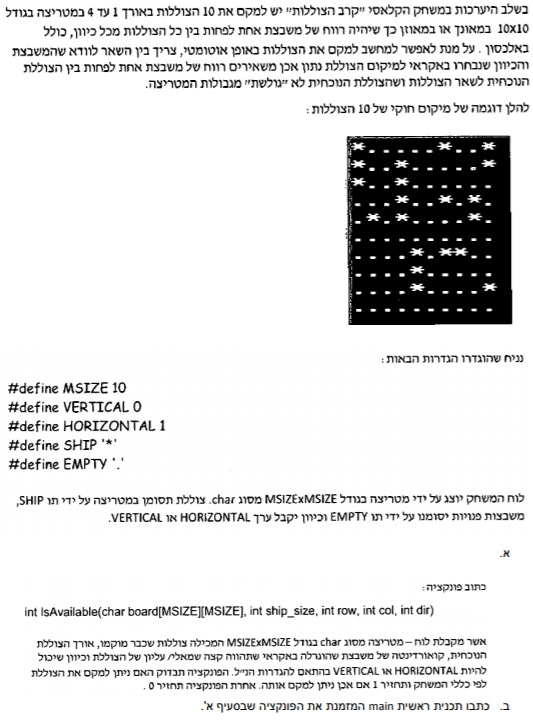
שאלה 5- מטריצצת בית ספר...............................................................18

שאלה 6- מעקב 1...............................................................................23

שאלה 7-מעקב 2...............................................................................25

**שאלה מספר 1- "קרב צוללות"**

**שאלה:**



**פתרון:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#define MSIZE 10

#define VERTICAL 0

#define HORIZONTAL 1

#define SHIP '\*'

#define EMPTY '.'

enum boolean { False, True };

//פונקציה המדפיסה לוח עם חומותיו

void printBourdWithWalls(char board[MSIZE + 2][MSIZE + 2])

{

for (int r = 0; r < MSIZE+2; r++)

{

for (int c = 0; c < MSIZE+2; c++)

{

printf(" %c", board[r][c]);

}

printf("\n");

}

}

//פונקציה המקבלת לוח עם חומות ומדפיסה אותו ללא חומות

void printBourdWithoutWalls(char board[MSIZE + 2][MSIZE + 2])

{

for (int r = 1; r < MSIZE; r++)

{

for (int c = 1; c < MSIZE; c++)

{

printf(" %c", board[r][c]);

}

printf("\n");

}

}

//פונקציה המדפיסה לוח

void printBourd(char board[MSIZE][MSIZE])

{

for (int r = 0; r < MSIZE; r++)

{

for (int c = 0; c < MSIZE; c++)

{

printf(" %c", board[r][c]);

}

printf("\n");

}

}

boolean check(char board[MSIZE+2][MSIZE+2], int row, int col)//פונקציה הבודקת מסביב לנקודה מסוימת שאין ספינה

{

//בודק האם בגבולות הלוח

if (row >= 11)

return False;

if (row <= 0)

return False;

if (col >= 11)

return False;

if (col <= 0)

return False;

//בודק האם לא נמצא מסביבו ספינה

if (board[row+1][col-1] == '\*')

return False;

if (board[row+1][col] == '\*')

return False;

if (board[row+1][col+1] == '\*')

return False;

if (board[row - 1][col-1] == '\*')

return False;

if (board[row - 1][col] == '\*')

return False;

if (board[row - 1][col+1] == '\*')

return False;

if (board[row][col-1] == '\*')

return False;

if (board[row][col] == '\*')

return False;

if (board[row][col+1] == '\*')

return False;

return True;

}

void buildSafeFall(char board[MSIZE][MSIZE],char bw[MSIZE+2][MSIZE+2])//פונקציה מקבלת שני מטריצות והופכת את המטריצה השניה למטריצה הראשונה עם קיר ביטחון

{

for (int c = 0; c < MSIZE + 2; c++)//מכניסה כוכביות בטורים בצדדים של המטריצה

{

bw[MSIZE + 1][c] = '.';

bw[0][c] = '.';

}

for (int r = 1; r < MSIZE; r++)

{

//מכניס שורה למטה ולמעלה בלוח עם הקיר

bw[r][0] = '.';

bw[r][MSIZE - 1] = '.';

//מעתיק את הערכים של הלוח הראשון

for (int c = 0; c < MSIZE; c++)

{

bw[r + 1][c + 1] = board[r][c];

}

}

}

//הפונקציה בסעיף א

int isAvailable(char board[MSIZE][MSIZE], int ship\_size, int row, int col, int dir)

{

if (ship\_size > 4)

return False;

char bw[MSIZE + 2][MSIZE + 2] = {

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','.','.','.','.','.','.','.','.'}};

buildSafeFall(board, bw);

if (dir == HORIZONTAL)//בודק לאורך הספינה כאשר היא בצורה אופקית

{

for (int vec = 0; vec < ship\_size; vec++)

{

if (check(bw, row+1, col + vec+1) == False)

return False;

}

return True;

}

else if (dir == VERTICAL)//בודק לאורך הספינה כאשר היא בצורה אנכית

{

for (int vec = 0; vec < ship\_size; vec++)

{

if (check(bw, row+vec+1, col+1) == False)

return False;

}

return True;

}

return False;

}

//פונקציה ראשית

void main()

{

//לוח משחק

char board[MSIZE][MSIZE] = {{'\*','.','.','.','.','.','\*','.','.','\*'},

{'\*','.','.','\*','.','.','.','.','.','\*'},

{'\*','.','.','\*','.','.','.','.','.','.'},

{'.','.','.','\*','.','.','\*','.','\*','.'},

{'.','\*','.','\*','.','.','.','.','\*','.'},

{'.','.','.','.','.','.','.','.','.','.'},

{'.','.','.','.','\*','.','\*','\*','.','.'},

{'.','.','.','.','\*','.','.','.','.','.'},

{'.','.','.','.','\*','.','.','.','.','\*'},

{'.','.','.','.','.','.','.','.','.','.'}};

//מימוש הפונקציה

int a = isAvailable(board, 3, 8, 0, HORIZONTAL);//מקרה אופקי אשר המקום פנוי לספינה

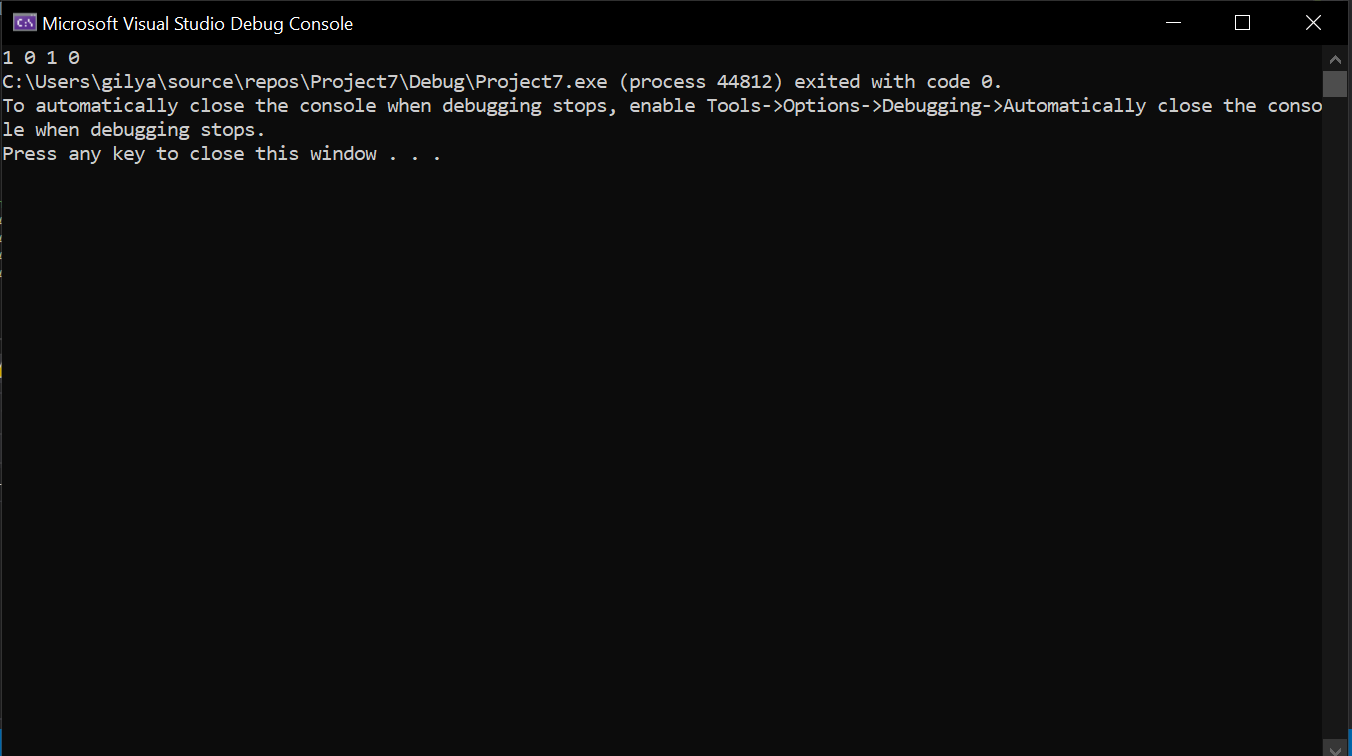
int b = isAvailable(board, 4, 8, 2, HORIZONTAL);//מקרה אופקי אשר המקום לא פנוי לספינה

int c = isAvailable(board, 2, 6, 2, VERTICAL);//מקרה אנכי אשר המקום פנוי לספינה

int d = isAvailable(board, 3, 8, 0, VERTICAL);//מקרה אנכי אשר המקום לא פנוי לספינה

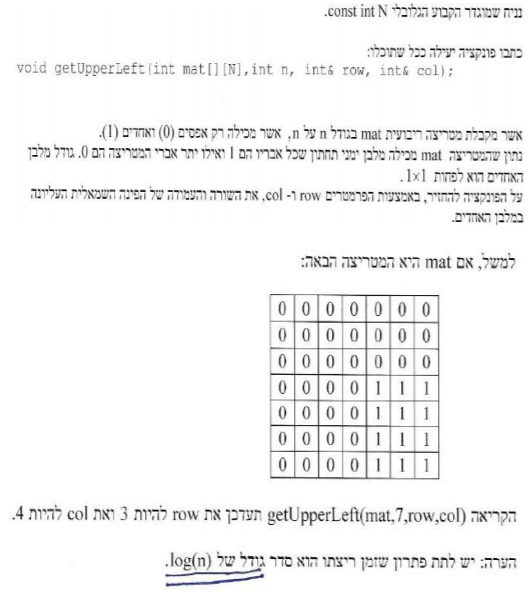
printf("%d %d %d %d", a, b, c, d);//1 0 1 0

}



**שאלה מספר 2- "מלבן ימני תחתון"**

**שאלה:**



**פתרון:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

const int N = 7;

//פונקציה המוצאת את אינדקס השורה שבו מתחיל המלבן ימני תחתון בעזרת רעיון חיפוש בינארי

int bSearchRow(int mat[N][N],int n, int up,int down)

{

int mid = (up + down) / 2;

if (mat[0][n - 1] == 1)

return 0;

if (down==1+up&&up==mid)

{

return down;

}

if (mat[mid][n - 1] == 1)

{

return bSearchRow(mat,n, up, mid);

}

else

{

return bSearchRow(mat,n, mid, down);

}

}

//פונקציה המוצאת את אינדקס הטור שבו מתחיל המלבן ימני תחתון בעזרת רעיון חיפוש בינארי

int bSearchCol(int arr[N], int left, int right)

{

int mid = (left + right) / 2;

if (arr[0] == 1)

return 0;

if (left + 1 == right && left == mid)

{

return right;

}

if (arr[mid] == 1)

{

return bSearchCol(arr, left, mid);

}

else

{

return bSearchCol(arr, mid, right);

}

}

//הפונקציה

void getUpperLeft(int mat[N][N], int n, int& row, int& col)

{

int\* arr = mat[n - 1];

row = bSearchRow(mat,n, 0, n - 1);

col = bSearchCol(arr, 0, n - 1);

}

//פונקציה ראשית

void main()

{

int mat[N][N] = {

{0,0,0,0,0,0,0},

{0,0,0,0,0,0,0},

{0,0,0,0,0,0,0},

{0,0,0,0,1,1,1},

{0,0,0,0,1,1,1},

{0,0,0,0,1,1,1},

{0,0,0,0,1,1,1}

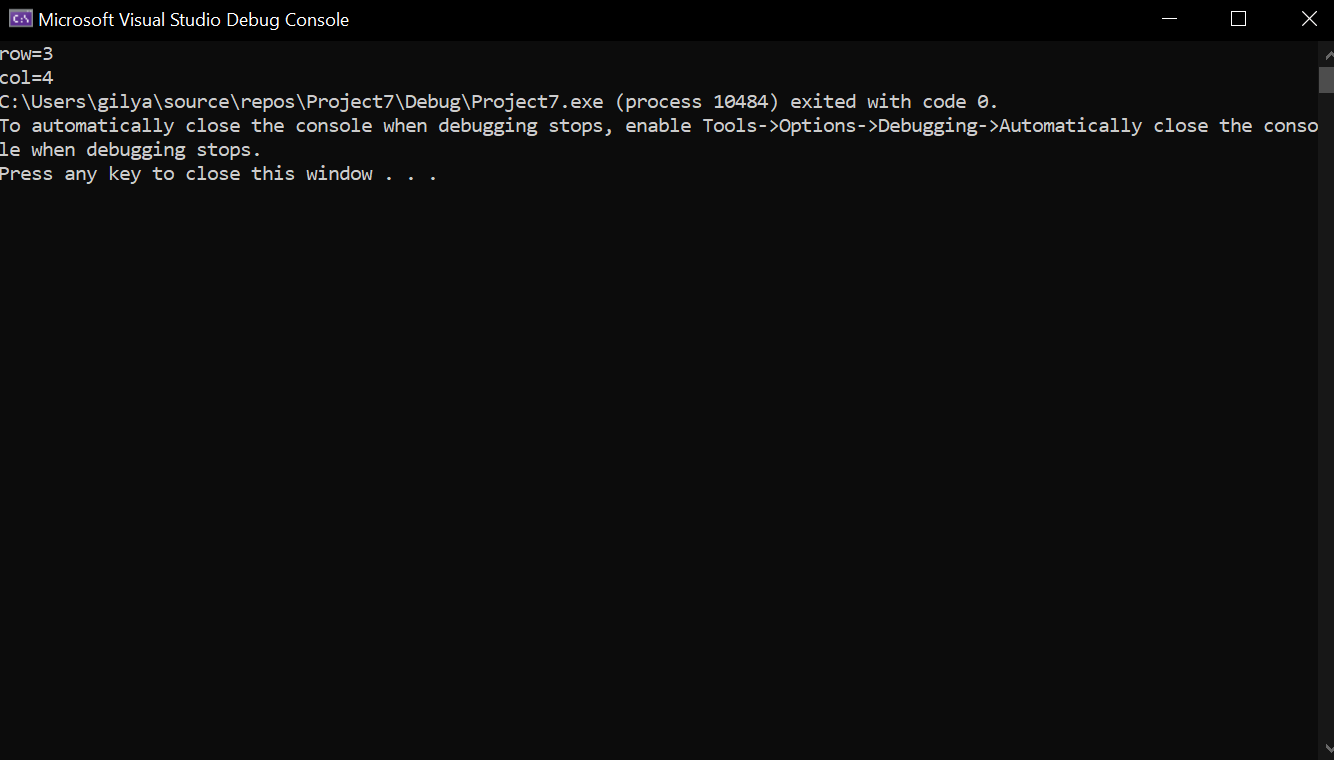
};

int r, c,n=N;

getUpperLeft(mat,n, r, c);

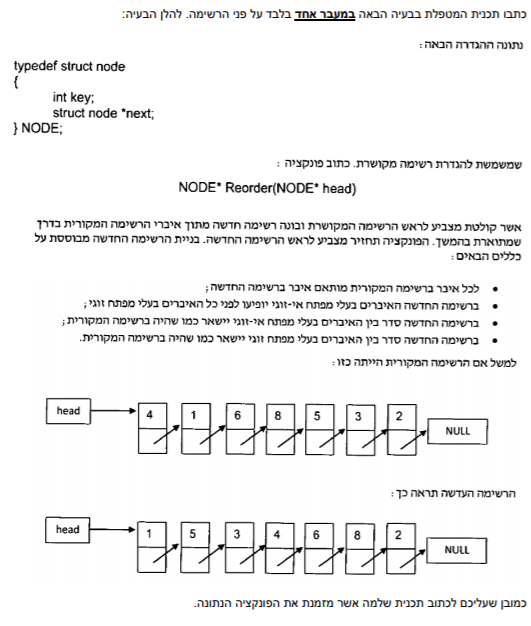
printf("row=%d\ncol=%d",r,c);

}



**שאלה מספר 3- "מיון רשימה לפי זוגיות מספרים"**

**שאלה:**



**פתרון:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <malloc.h>

//פונקציון של רשימה מקושרת

typedef struct node

{

int key;

struct node\* next;

}list\_node, \* nodePtr;

int empty(nodePtr L)

{

if (L == NULL)

return 1;

return 0;

}

// הוספה לתחילת הרשימה אותה אנו מחזירים בסיום הפונקציה הנדרשת, הונקציה מוסיפה לרשימה את האיברים כך שהאי זוגיים בהתחלה והזוגיים בסוף

void insertNodeToNewList(nodePtr\* L, int x)

{

nodePtr p;

nodePtr q=\*L;

p = (list\_node\*)malloc(sizeof(list\_node));

p->key = x;

p->next = NULL;

if (q==NULL)

{

\*L = p;

}

else if (q->key%2 == 0)

{

p->next = \*L;

\*L = p;

}

else

{

if (x % 2 == 0)//אם המספר זוגי שם אותו בסוף

{

while (q->next != NULL)

{

q = q->next;

}

q->next = p;

}

else //אם המספר אי זוגי דוחף אותו בין האי זוגיים לזוגיים

{

while (q->next != NULL)

{

if (((q->next->key) % 2 != 0))

q = q->next;

else

break;

}

p->next = q->next;

q->next = p;

}

}

}

//הוספה לתחילת הרשימה

void insertNode(nodePtr\* L, int x)

{

nodePtr p;

p = (list\_node\*)malloc(sizeof(list\_node));

p->key = x;

p->next = NULL;

if (empty(\*L))

{

\*L = p;

}

else

{

p->next = \*L;

\*L = p;

}

}

void printfAll(nodePtr p)

{

nodePtr pos = p;

while (pos != NULL)

{

printf("%d ", pos->key);

pos = pos->next;

}

}

//הפונקציה המבוקשת

nodePtr recorder(nodePtr head)

{

nodePtr p = head;

nodePtr final = NULL;

while (p != NULL)

{

insertNodeToNewList(&final, p->key);

p = p->next;

}

return final;

}

//פונקציה ראשית

void main()

{

nodePtr head=NULL;

insertNode(&head, 2);

insertNode(&head, 3);

insertNode(&head, 5);

insertNode(&head, 8);

insertNode(&head, 6);

insertNode(&head, 1);

insertNode(&head, 4);

printfAll(head);

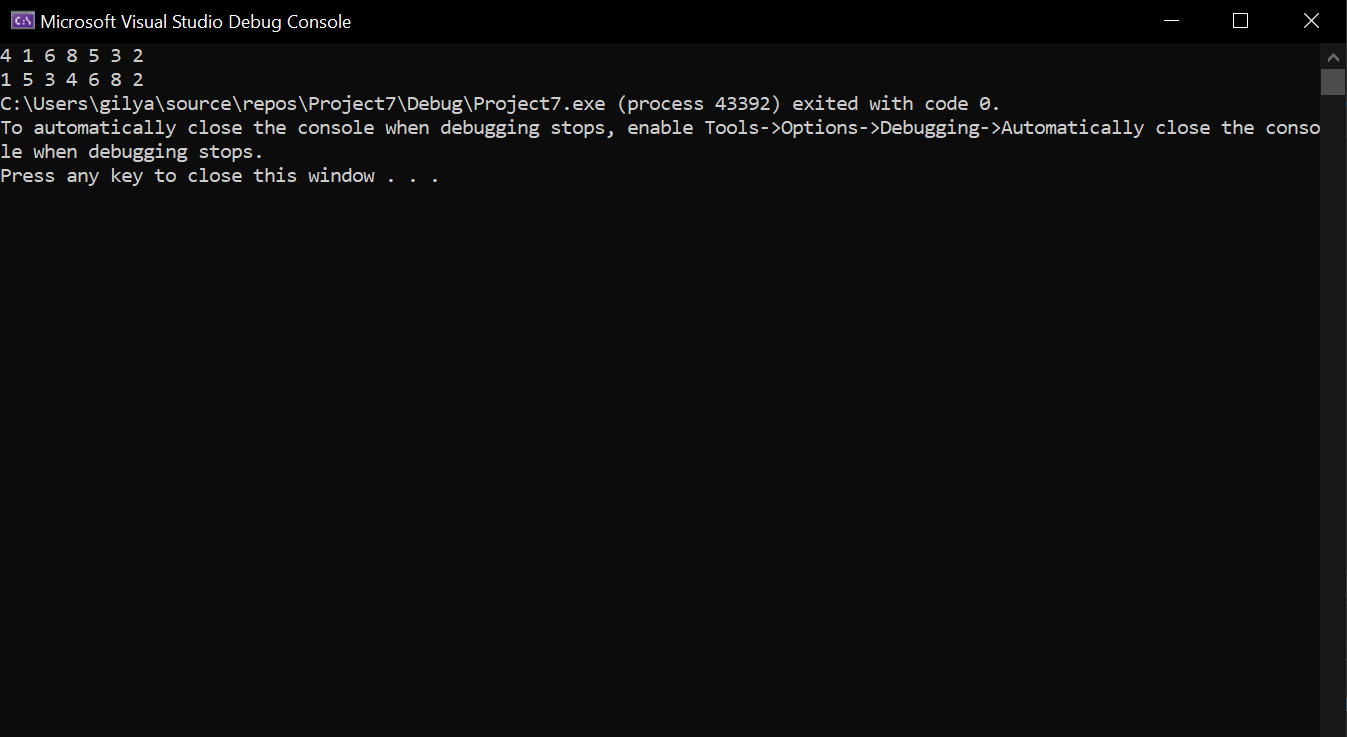
nodePtr final = NULL;

final=recorder(head);

printf("\n");

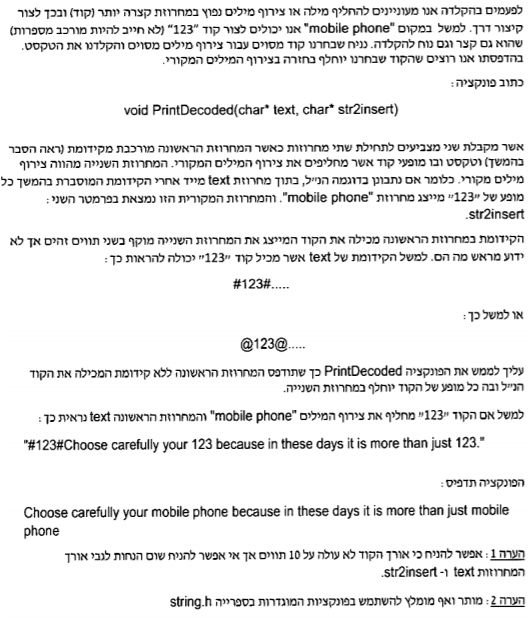
printfAll(final);

}



**שאלה מספר 4- "קיצורים"**

**שאלה:**



**פתרון:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

enum boolean {False,True};

//פונקציה המקבלת מחרוזת וקוד ובודקת אם הקוד נמצא בתחילת המחרוזת

boolean check(char\* text,char\* code)

{

int len = strlen(code);

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

{

if (text[i] != code[i])

return False;

}

return True;

}

//הפונקציה המתבקשת בשאלה

void printDecode(char \*text,char\* str2insert)

{

char codeSigne = text[0];

char\* textPtr = text + 1;

char code[10] = { 0 };

int i = 0;

//מציאת הקוד והעתקתו למשתנה

for (; i < 10 && textPtr[i] != codeSigne;i++)

code[i] = textPtr[i];

code[i] = 0;

textPtr += (i+1);

while (textPtr[0] != 0)//כל עוד לא הגענו לסוף הטקסט

{

if (textPtr[0] == code[0])

{

if (check(textPtr, code) == True)//אם הקוד נמצא

{

printf("%s", str2insert);

textPtr += strlen(code);

}

else

{

printf("%c", textPtr[0]);//כאשר אינו נמצא

textPtr++;

}

}

else//כאשר אינו נמצא

{

printf("%c", textPtr[0]);

textPtr++;

}

}

}

//פונקציה ראשית

void main()

{

char text[] = "$158$hi my name is \_\_\_.\ni am 16 years old and i love 158!\ni live in 158, israel.\nmy brother like 158 too!";

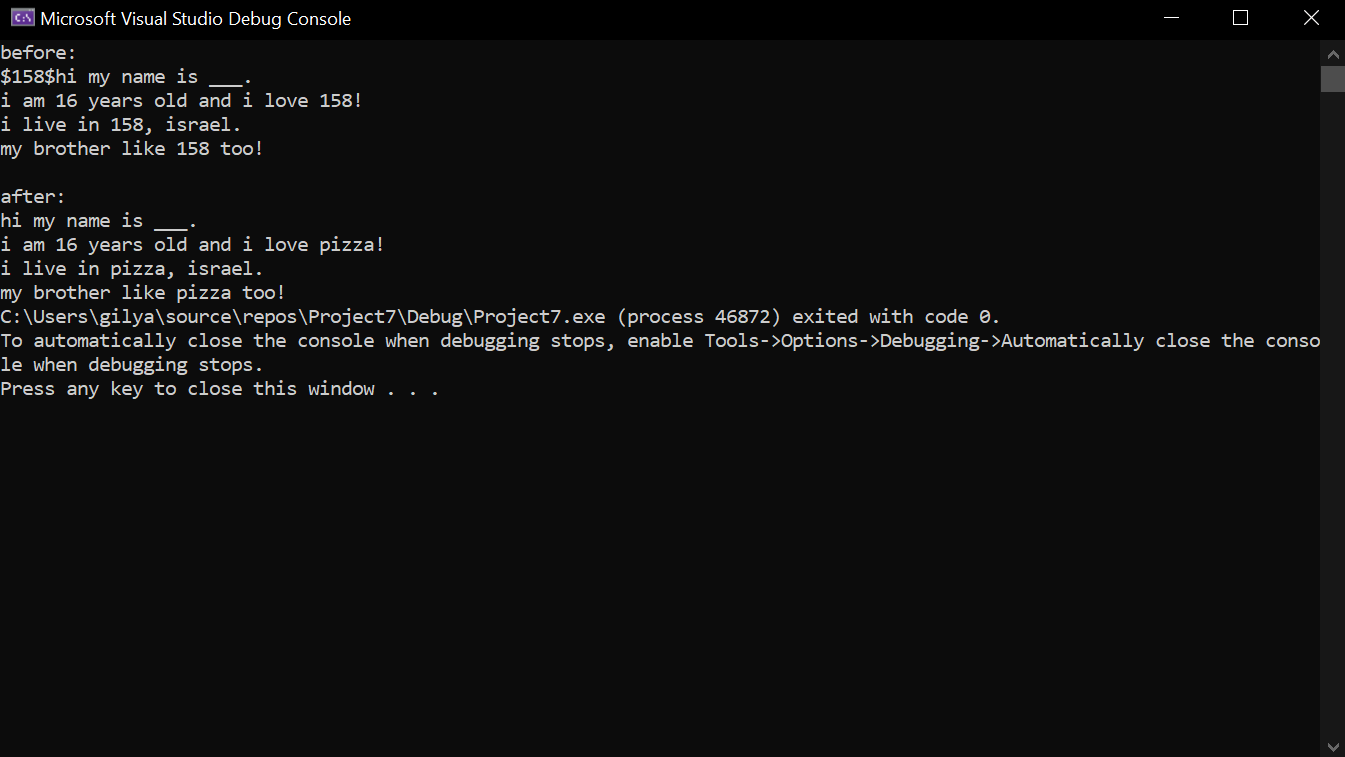
char str[] = "pizza";

printf("before:\n%s", text);

printf("\n\nafter:\n");

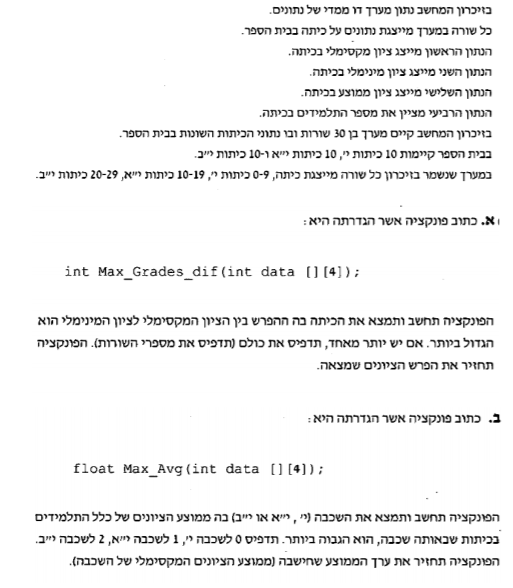
printDecode(text, str);

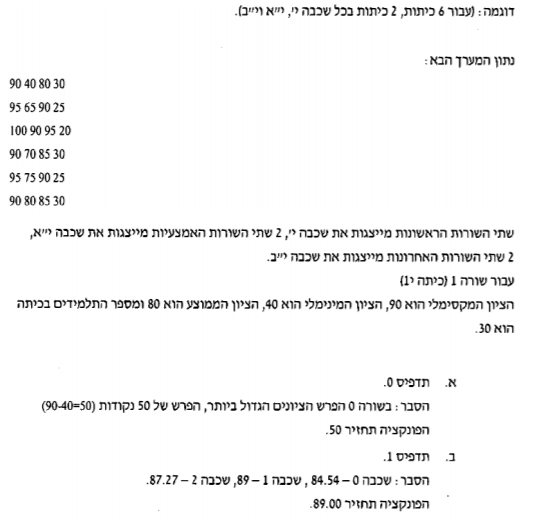
}



**שאלה מספר 5- "מטריצת בית ספר"**

**שאלה:**





**פתרון:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

//הפונקציה המתבקשת בסעיף א

int max\_grade\_dif(int data[][4])

{

int max = 0;

for (int i = 0; i < 30; i++)//מציאת ההפרש הגדול ביותר

{

if (data[i][0] - data[i][1] > max)

max = data[i][0] - data[i][1];

}

printf("lines with best grades dif: ");

for (int i = 0; i < 30; i++)//הדפסת השורות בהן יש את ההפרש הגדול ביותר

{

if (data[i][0] - data[i][1] == max)

printf("%d ", i);

}

printf("\n");

return max;

}

//פונקציה המחזיקה מספר הגדול ביותר בין שני מספרים

float max(float a, float b)

{

return (a > b ? a : b);

}

//הפונקציה המתבקשת בסעיף ב

float max\_avg(int data[][4])

{

float avg1 = 0, avg2 = 0, avg3 = 0;//ממוצע כל שכבה

int sum1 = 0, sum2 = 0, sum3 = 0;//סכום הציונים בכל שכבה

int count1 = 0, count2 = 0, count3 = 0;//מספר התלמידים בכל שכבה

int i = 0;

for (; i < 10; i++)//עובר על השכבה הראשונה

{

sum1 += (data[i][2] \* data[i][3]);

count1 += data[i][3];

}

avg1 = float(sum1) / float(count1);

for (; i < 20; i++)//עובר על השכבה השניה

{

sum2 += (data[i][2] \* data[i][3]);

count2 += data[i][3];

}

avg2 = float(sum2) / float(count2);

for (; i < 30; i++)//עובר על השכבה השלישית

{

sum3 += (data[i][2] \* data[i][3]);

count3 += data[i][3];

}

avg3 = float(sum3) / float(count3);

float maxAvg = max(avg3, max(avg1, avg2));

if (maxAvg == avg1)

printf("\nlayer with the best avg is: 0\n");

else if (maxAvg == avg2)

printf("\nlayer with the best avg is: 1\n");

else

printf("\nlayer with the best avg is: 2\n");

return maxAvg;

}

//פונקציה ראשית

void main()

{

int data[30][4] = {

{100,72,76,248},

{98,42,75,24},

{94,72,84,20},

{92,62,80,28},

{95,52,82,27},

{97,42,78,24},

{78,2,67,40},

{89,32,77,25},

{75,62,70,26},

{91,52,82,27},

{80,75,77,29},

{90,72,80,25},

{71,45,60,20},

{91,64,77,23},

{99,35,79,22},

{86,62,80,27},

{77,55,68,25},

{96,43,80,24},

{77,36,56,21},

{60,28,49,29},

{60,29,50,27},

{93,28,81,25},

{90,48,82,23},

{90,57,78,21},

{90,72,85,24},

{88,34,79,22},

{94,35,85,24},

{97,37,90,27},

{84,26,80,25},

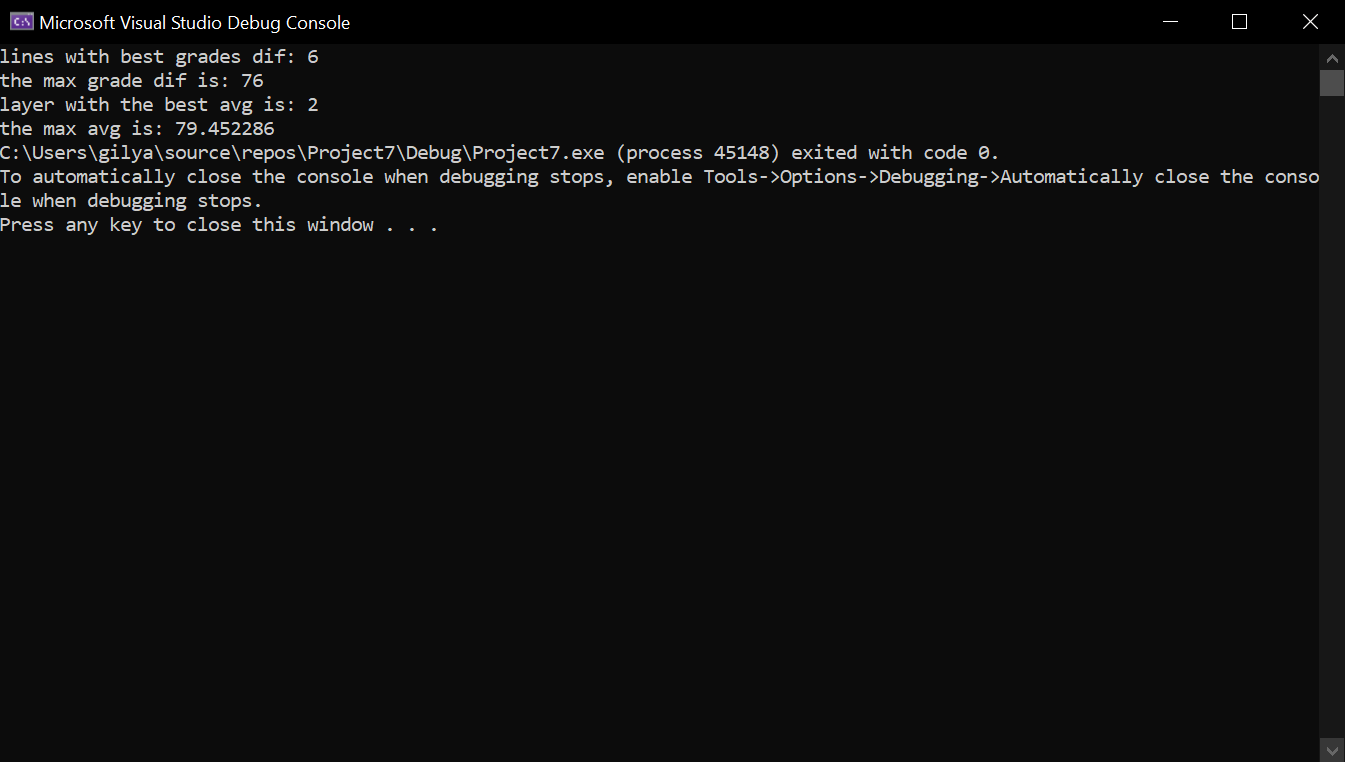
{100,69,87,23}

};

printf("the max grade dif is: %d",max\_grade\_dif(data));

printf("the max avg is: %f", max\_avg(data));

}



**שאלה מספר 6- מעקב 1**

**שאלה:**



**פתרון:**

התוכנית:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<string.h>

void fix\_star(char\* s)

{

int i;

for (i = 0; i < strlen(s); i++)

{

if (s[i] == ' ')

{

s[i] = '\0';

fix\_star(s + i + 1);

strcat(s, s + i + 1);

}

}

}

int main()

{

char s1[30] = "question number 5 in c exam";

printf("%s\n", s1);

fix\_star(s1);

printf("%s\n", s1);

return 0;

}

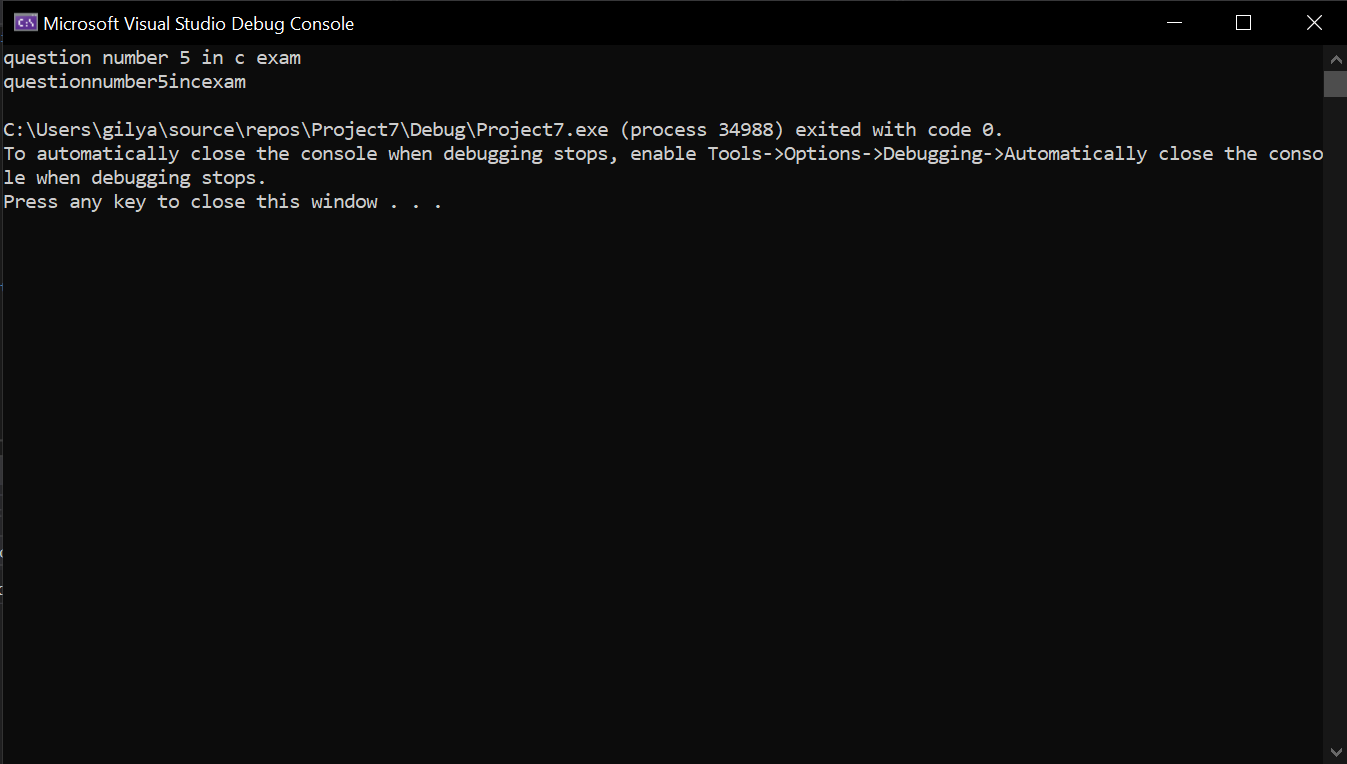
1. פלט התוכנית הוא :

Question number 5 in c exam

questionnumber5incexam

1. הפונקציה מקבלת מחרוזת ומורידה ממנה את כל הרווחים. לדוגמה:

Hello world!->Helloworld!



**שאלה מספר 7- מעקב 2**

**שאלה:**



**פתרון:**

התוכנית:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

int cor\_func(unsigned short num1, char num2)

{

int i, j;

unsigned short num2s, mask = 0xf;

int res = 0;

num2s = num2 & 0xf;

for (i = 0; i < sizeof(num1) \* 8; i++)

{

res += (((num1 & mask) >> i) == num2s);

mask = mask << 1;

}

return res;

}

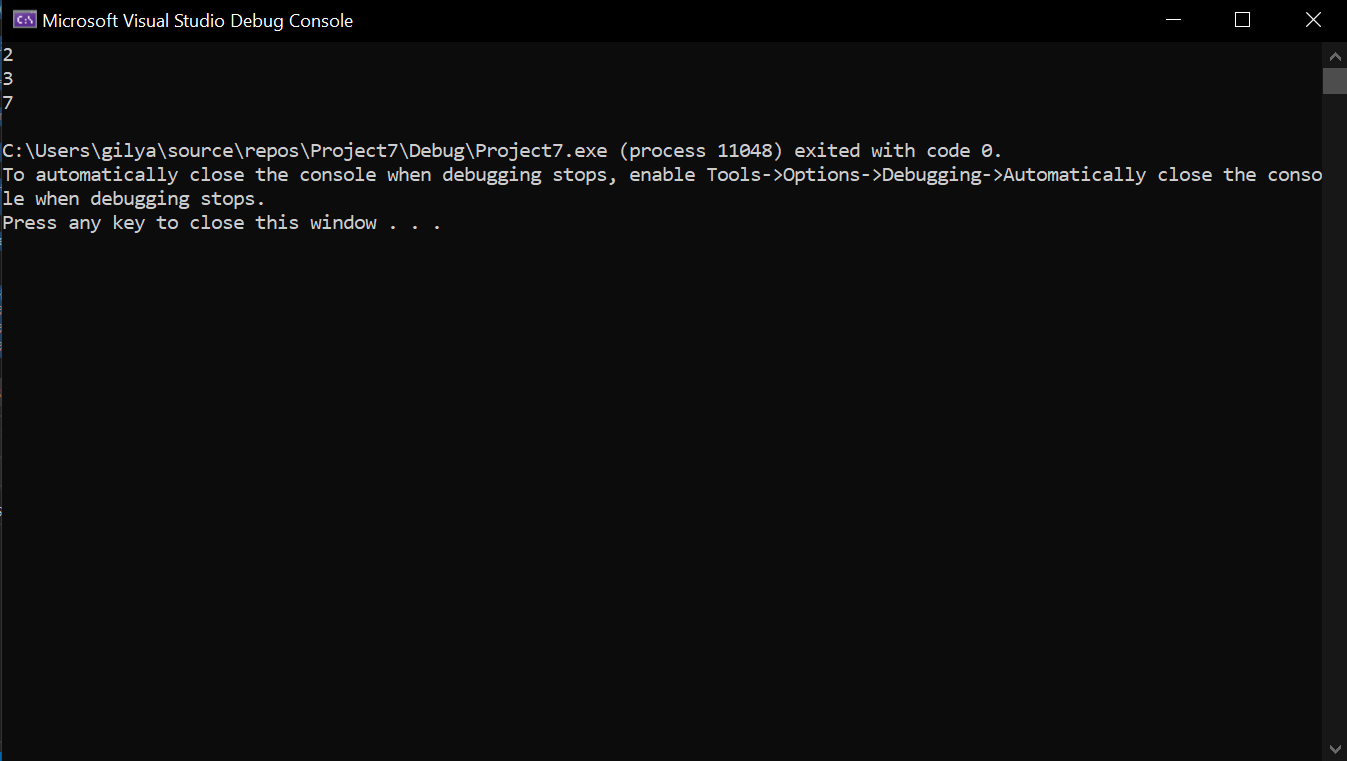
void main() {

printf("%d\n", cor\_func(0x1234, 0x1));

printf("%d\n", cor\_func(0xe387, 0x7));

printf("%d\n", cor\_func(0xaaaa, 0x5));

}



1. הפונקתיה מחזירה כמה פעמים נמצא הרצף הבינארי של num2 בתוך הרצף הבינארי של num1 לדוגמה: num1=0101,1101,0111,0001 ו- num2=1110 הפונקציה תחזיר 2 כי הרצף 1110 נמצא פעמיים: 0101,1101,0111,0001.
2. התוכנית תדפיס: 2

3

7

כיוון ש:

כאשר num1=0x1234 ו- num2=0x1 בבינארי הם שווים num1=0001,0010,0011,0100 ו- num2=0001 , הרצף של num2 נמצא 2 פעמים בnum1 כך: 0001,0010,0011,0100.

כאשר num1=0xe387 ו- num2=0x7 בבינארי הם שווים num1=1110,0011,1000,0111 ו- num2=0111 , הרצף של num2 נמצא 3 פעמים בnum1 כך: 1110,0011,1000,0111

כאשר num1=0xaaaa ו- num2=0x5 בבינארי הם שווים num1=1010,1010,1010,1010ו - num2=0101 , הרצף של num2 נמצא 7 פעמים בnum1 כך:

1010,1010,1010,1010  
 1010,1010,1010,1010  
 1010,1010,1010,1010