מכון טכנולוגי חולון
Holon Institute of Technology

המחלקה למדעי המחשב COMPUTER SCIENCE DEPARTMENT

סדנה מתקדמת בתכנות 61108
סמסטר ק' תשע"א
מועד ב'
7.11.2011

מרצה: ד"ר מרק קורנבליט

משך המבחן: שלוש שעות

אין להשתמש בחומרי עזר, פרט, לדף מצורף לשאלון

שאלה 1 (28 נק')

כתוב פונקציה המקבלת מערך A של מספרים שלמים לא שליליים.
המערך מורכב מסדרות של מספרים חיוביים המובדלות באפסים (הסדרה יכולה להיות ריקה, ז"א אפסים ב-A יכולים להופיע גם ברציפות). ידוע כי האיבר הראשון והאיבר האחרון של A הם לא אפסים.

על הפונקציה לבנות מערך דו-ממדי **דינאמי** בעל שורות עם גדלים שונים כדלקמן:

- מספר שורות במערך דו-ממדי יהיה שווה למספר סדרות של מספרים חיוביים ב- A;
- איבר מס' **0** בשורה מס' **i** במערך דו-ממדי יהיה שווה למספר איברים בסדרה מס' **i** (משמאל לימין) במערך A;
- תוכן של שאר האיברים בשורה מס' **i** במערך דו-ממדי יהיה שווה לתוכן של סדרה מס' **i** (משמאל לימין) במערך A (איבר מס'**j** בשורה מס' **i** יהיה שווה לאיבר מס' **j-1** בסדרה מס' **i**).

הפונקציה תחזיר כתובת של המערך הדו-ממדי הדינאמי הבנוי ותעביר את מספר השורות בתוכו.

דוגמא:

A = 5, 7, 9, 0, 12, 11, 0, 1, 3, 10, 87, 0, 6, 5, 0, 0, 31

מערך חדש

EMBED Equation.3

הפונקציה תעביר 6 בדוגמא.

ניתן להניח שיש בזיכרון מספיק מקום להקצאה.

```
int **one_to_two_dimensional (int *A, int A_size, int *rows_num)
{
        // temp. array of row sizes (initially - zeros)
        int *row_size = calloc (A_size-1, sizeof(int)),
            **arr, offset=0, i=0, j;
        row_size[0] = 1;
        for (j=1; j<A_size; j++)
                if (A[j] != 0)
                        row_size[i]++;
                else
                        i++; // new sequence
        *rows_num = i + 1;
        arr = calloc (*rows_num, sizeof(int*));
        for (i=0; i < *rows_num; i++)
        {
                arr[i] = calloc (row_size[i]+1, sizeof(int)); // allocation of row i
                arr[i][0] = row_size[i];
                /* copying sequence i into row i */
                for (j=0; j < row_size[i]; j++)
                        arr[i][j+1] = A[offset+j];
                offset += row_size[i] + 1;
        }
        free (row_size);
        return arr;
}
```

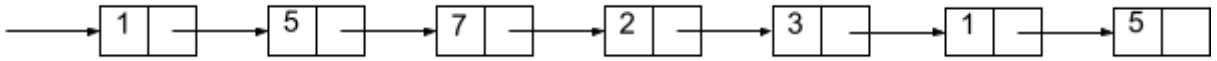
שאלה 2 (28 נק')

כתוב פונקציה המקבלת רשימה מקושרת של מספרים שלמים ומספר שלם k.

על הפונקציה לבנות שתי רשימות מקושרות חדשות כך שהרשימה הראשונה תהיה מורכבת מאיברי הרשימה המקורית הגדולים מ- k והרשימה השנייה תהיה מורכבת מאיברי הרשימה המקורית הקטנים מ- k.

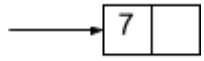
הפונקציה תעביר את המצביעים לאיברים הראשובים של שתי הרשימות החדשות ותחזיר את מספר האיברים של הרשימה המקורית השווים ל- k.

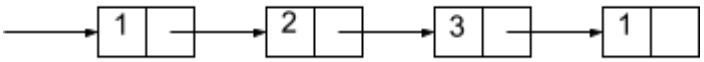
לדוגמא, עבור הרשימה הבאה:



ומספר k השווה ל-5

הפונקציה תיצור את שתי הרשימות החדשות הבאות:





ותחזיר 2.

ניתן להניח שיש בזיכרון מספיק מקום להקצאה.

```
typedef struct element
{
        int data;
        struct element *ptr_next;
}       element;

int lists_of_large_small_num (element *lst, int k, element **lst_large,
                                        element **lst_small)
{
        element *ptr_large, *ptr_small;
        int equal_num=0;
        /* allocation of dummy elements */
        *lst_large = malloc(sizeof(element));
        *lst_small = malloc(sizeof(element));
        /* ---------------------------- */
        ptr_large = *lst_large;
        ptr_small = *lst_small;
        while (lst)
        {
                if (lst->data > k)
                {
                        ptr_large->ptr_next = malloc(sizeof(element));
                        ptr_large = ptr_large->ptr_next;
                        ptr_large->data = lst->data;
                }
                else
                        if (lst->data < k)
                        {
                                ptr_small->ptr_next = malloc(sizeof(element));
                                ptr_small = ptr_small->ptr_next;
                                ptr_small->data = lst->data;
                        }
                        else
                                equal_num++;
                lst = lst->ptr_next;
        }
        ptr_large->ptr_next = ptr_small->ptr_next = NULL;
        /* deletion of dummy elements */
        ptr_large = *lst_large;
        *lst_large = (*lst_large)->ptr_next;
        free (ptr_large);
        ptr_small = *lst_small;
        *lst_small = (*lst_small)->ptr_next;
        free (ptr_small);
        /* --------------------------- */
        return equal_num;
}
```

שאלה 3 (28 נק')

כתוב פונקציה המקבלת מספר שלם (חיובי או שלילי או אפס) ומחזירה מצביע למחרוזת מספרית **דינאמית** שהיא ההצגה של המספר הנתון.
כל תוויי המחרוזת פרט לתו הראשון יהיו ספרות. עבור המספר הנתון השלילי התו הראשון יהיה '-'. אחרת הוא יהיה ספרה.

לדוגמא, עבור המספרים 1896, -569, 0 הפונקציה תחזיר את המחרוזות "1896", "-569", "0" בהתאמה.

יש להקצות את המחרוזת בתוך הפונקציה.
ניתן להניח שיש בזיכרון מספיק מקום להקצאה.

לא להשתמש בפונקציה סטנדרטית **itoa** המממשת את האלגוריתם הנ"ל!

```
char *num_to_string (num)
{
        int temp = num, len=0, i;
        char *str;
        /* computation of the number of digits */
        do
        {
                len++;
                temp /= 10;
        }
        while (temp != 0);
        /* -------------------------- */
        if (num < 0)
                len++; // place for '-'
        str = malloc (len+1); // including '/0'
        str[len] = '\0';
        temp = abs(num);
        i = len-1;
        /* conversion of digits to characters of string */
        do
        {
                str[i] = '0' + temp%10;
                i--;
                temp /= 10;
        }
        while (temp != 0);
        /* -------------------------- */
        if (num < 0)
                str[0] = '-';
        return str;
 }
```

שאלה 4 (16 נק')

נתונה התוכנית הבאה:

```
#include <stdio.h>
#include <string.h>

void fun (char *, const char *);

void main ()
{
     char *str="abc", msg[]="xyz";
     str = "hello";
     puts (str);
     //running error:
     strcpy (str, msg); // changing constant string
     puts (str);
     //running error:
     strcpy (msg, "hello"); // not enough place
                            in array msg (4 chars only)
     puts (msg);
     msg = "good bye"; // '=' : left operand must
                               be l-value
     puts (msg);
     fun (msg, str);
     puts (msg);
     str = "good bye";
     //running error:
     fun (str, msg); // changing constant string
     puts (str);
}

void fun (char *s1, const char *s2)
{
     s2++;
     *s1 = *s2;
}
```

התוכנית שגויה!
'ש לתאר את כל השגיאות (קומפילציה וריצה).

שאלה 5 (בונוס – 10 נק')

מהו פלט התוכנית הבאה?

```
#include <stdio.h>
#include <ctype.h>

void main()
{
     char c = getchar();
     printf ("%d", isdigit(c)&&isalpha(c) ||
                   islower(c)&&isupper(c) ||
                   isalnum(c)&&isspace(c));
}
```

0