מכון טכנולוגי חולון
Holon Institute of Technology

מבחן סדנה מתקדמת בתכנות - 61108

ד"ר מרק קורנבליט, ד"ר לאוניד קוגל, מר טוביה דב רוזנברג

סמסטר ב' , מועד ב', תשפ"ב

תאריך: 25.07.22

הוראות:

- משך המבחן 3 שעות.
- אין להשתמש בחומרי עזר, פרט לדף המצורף לשאלון.

חלק 1. בחלק זה יש לענות על שתי שאלות מתוך שלוש.

שאלה 1 (40 נק')

במחרוזת בעלת אורך זוגי ($2n$) **המחצית הראשונה** מורכבת מהתווים במקומות מס' 0 עד מס' n-1 **והמחצית השנייה** מורכבת מהתווים במקומות מס' n עד מס' $2n$-1.
במחרוזת בעלת אורך אי-זוגי ($2n$+1) **המחצית הראשונה** מורכבת מהתווים במקומות מס' 0 עד מס' n-1 **והמחצית השנייה** מורכבת מהתווים במקומות מס' n+1 עד מס' $2n$. **התו האמצעי** (מס' n) לא שייך לאף מחצית.

כתבו פונקציה אשר מקבלת מחרוזת וסופרת מספר אותיות קטנות במחצית הראשונה ומספר אותיות גדולות במחצית השנייה. על הפונקציה להעביר את התוצאות by reference.
אם אורך המחרוזת הוא זוגי, הפונקציה תחזיר 0.
אם אורך המחרוזת הוא אי זוגי אז הפונקציה תחזיר 1 במקרה שהתו האמצעי הוא ספרה ו- 1- בכל מקרה אחר.

דוגמאות (יש לקרוא את המחרוזות משמאל לימין):

עבור המחרוזת "t&u5Y%as3k$R" (האורך הוא זוגי) הפונקציה תעביר 2 (מספר האותיות הקטנות במחצית הראשונה) ו- 1 (מספר האותיות הגדולות במחצית השנייה) ותחזיר 0.

עבור המחרוזת "t&u5Y%**4**as3k$R" (האורך הוא אי זוגי) הפונקציה תעביר 2 (מספר האותיות הקטנות במחצית הראשונה) ו- 1 (מספר האותיות הגדולות במחצית השנייה) ותחזיר 1.

עבור המחרוזת "t&u5Y%**&**as3k$R" (האורך הוא אי זוגי) הפונקציה תעביר 2 (מספר האותיות הקטנות במחצית הראשונה) ו- 1 (מספר האותיות הגדולות במחצית השנייה) ותחזיר 1-.

הפקולטה למדעים
המחלקה למדעי המחשב
Faculty of Sciences
Department of Computer Science
גולומב 52, ת.ד 305, חולון 5810201
טלפקס: 03-5026528
52 Golomb St., Holon 5810201 Israel
www.hit.ac.il Tel/Fax: 972-3-502-6528

```
int nums_in_halves(char* str, int* num1, int* num2)
{
	int len = strlen(str), i;

	*num1 = *num2 = 0;
	for (i = 0; i < len / 2; i++)
		if ( islower(str[i]) )
			(*num1)++;
	for (i = (len + 1) / 2; str[i] != '\0'; i++)
		if ( isupper(str[i]) )
			(*num2)++;
	if (len % 2 == 0)
		return 0;
	if ( isdigit(str[len / 2]) )
		return 1;
	return -1;
}
```

שאלה 2 (40 נק')

מטריצת חברויות M של N אנשים היא מטריצה **ריבועית סימטרית** בגודל N בה כל האיברים **שווים ל-0 או 1 בלבד**.
במטריצה זו נגדיר אנשים I,J שהם **חברים ישירים** אם M[I][J]=1 (בהתאם ליחס סימטרי בהכרח גם M[J][I]=1).
נגדיר **חברים עקיפים** כזוג אנשים שיש להם חבר משותף. כלומר I,J הם חברים עקיפים אם קיים K כך ש- M[I][K]=1 וגם M[J][K]=1 כאשר I,J אינם חברים ישירים (M[I][J]=0).

דוגמא:

עבור המטריצה הבאה האנשים 1 ו-4 הם חברים עקיפים כי 1 חבר של 2 ו- 4 חבר של 2. אין חברות ישירה של 1 ו- 4.

Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	0	0	0	1	0	0
1	1	0	1	0	0	0	1	0
2	0	1	0	0	1	0	0	1
3	0	0	0	0	0	1	0	1
4	0	0	1	0	0	1	0	0
5	1	0	0	1	1	0	1	0
6	0	1	0	0	0	1	0	1
7	0	0	1	1	0	0	1	0

כתבו פונקציה המקבלת מטריצה חברויות המיושמת כמערך דו-ממדי **דינאמי** וגודלה. על הפונקציה ליצור מערך של כל זוגות החברים העקיפים שבמטריזה.
הפונקציה תחזיר את כתובת תחילת המערך החדש ותעביר (by reference) את גודלו. אם אין במטריצה אף זוג חברים עקיפים, הפונקציה מחזירה NULL ומעבירה 0.

יש להגדיר מעל הפונקציה את **זוג החברים העקיפים** כ**מבנה** שמורכב משני המספרים המהווים **אינדקסים של החברים**.

<u>שימו לב.</u> חברים ישירים אינם נחשבים לחברים עקיפים, אפילו אם אפשר למצוא גם קשר עקיף ביניהם.
<u>שימו לב.</u> על כל זוג חברים עקיפים להופיע פעם אחת בלבד במערך החדש. למשל, אם הזוג {1,4} יופיע במערך אז המערך לא יכיל את הזוג {4,1}.

ניתן להניח שיש בזיכרון מספיק מקום להקצאת המערך החדש.

הפקולטה למדעים
המחלקה למדעי המחשב
Faculty of Sciences
Department of Computer Science
גולומב 52, ת.ד 305, חולון 5810201
טלפקס: 03-5026528
52 Golomb St., Holon 5810201 Israel
www.hit.ac.il Tel/Fax: 972-3-502-6528

```
typedef struct
{
	int i, j;
} pair;


pair* indirect_friends(int** M, int size, int* new_size)
{
	int i, j, k, count = 0;
	pair* arr;

	*new_size = (size * size - size) / 2; //upper bound of possible number
of pairs
	arr = (pair*)calloc(*new_size, sizeof(pair));

	for (i = 0; i < size; i++)
		for (j = 0; j < i; j++)
			if (M[i][j] == 0)
				for (k = 0; k < size; k++)
					if (M[i][k] == 1 && M[k][j] == 1)
					{
						arr[count].i = i;
						arr[count++].j = j;
						break; //search for common friends of i
and j has been terminated
					}

	*new_size = count;
	arr = (pair*)realloc(arr, *new_size * sizeof(pair));
	return arr;
}
```

שאלה 3 (40 נק')

נתונה הגדרת המבנה הבאה:

```
typedef struct {
        char model[20];
        int price;
        int year;
} car;
```

המתאר את הנתונים על מכונית בחברת מכירות, כאשר:

- model – שם הדגם,
- price – מחיר המכונית,
- year – שנת היצור.

כתבו פונקציה בשם ***appropriateCar*** אשר מקבלת מחרוזת ***fileName*** המהווה שם של **קובץ בינארי**. הקובץ מורכב מרשומות הנתונים שהפורמט של כל אחת מהן מתאים למבנה car. כמו כן, הפונקציה מקבלת כפרמטרים את הדגם המבוקש ע"י הלקוח (מחרוזת ***desired_model***), את המחיר המירבי שהקונה מוכן לשלם (מספר שלם ***max_price***) ואת שנת היצור המינימלי (מספר שלם ***min_year***).

על הפונקציה למצוא את כל המכוניות של הדגם desired_model מהקובץ בעלות מחיר שאינו עולה על max_price ושנת יצור לא לפני min_year.
הפונקציה תיצור רשימה מקושרת למחירים ושנות יצור של כל המכוניות המתאימות לתנאים לעיל. בזאת איברי הרשימה יהיו מורכבים משלושת השדות הבאים: מחיר, שנת יצור, ומצביע לאיבר הבא.
הפונקציה תחזיר את כתובת תחילת הרשימה. אם אף מכונית בקובץ לא מתאימה לתנאים לעיל או הקובץ לא נפתח, הפונקציה תחזיר NULL.
יש להגדיר את הטיפוס של איברי הרשימה מעל הפונקציה.

ניתן להניח שיש בזיכרון מספיק מקום להקצאת הרשימה.

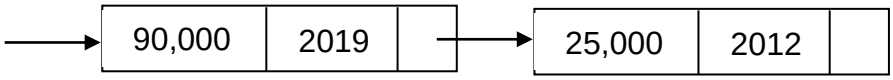
דוגמא:

עבור הקובץ הבא:

model	Subaru	Mazda	Subaru	Toyota	Subaru
price	90,000	95,000	10,000	200,000	25,000
year	2019	2020	2009	2022	2012

ו- "Subaru"=desired_model, 100,000=max_price, 2010=min_year

הפונקציה תיצור את הרשימה הבאה:



```
typedef struct item {
	int price;
	int year;
	struct item* next;
} Item;


Item* appropriateCar(char* fileName, char* desired_model, int max_price, int
min_year)
{
	FILE* fptr = fopen(fileName, "rb");
	car unit;
	Item* lst, * curr;
	if (!fptr)
		return NULL;

	curr = lst = (Item*)malloc(sizeof(Item)); //dummy item

	fread(&unit, sizeof(car), 1, fptr);
	while (!feof(fptr))
	{
		if (strcmp(unit.model, desired_model) == 0 && unit.price <=
max_price && unit.year >= min_year)
		{
			curr->next = (Item*)malloc(sizeof(Item));
			curr = curr->next;
			curr->price = unit.price;
			curr->year = unit.year;
		}
		fread(&unit, sizeof(car), 1, fptr);
	}
	curr->next = NULL;

	/* Deleting dummy */
	curr = lst;
	lst = lst->next;
	free(curr);

	fclose(fptr);
	return lst;
}
```

מכון טכנולוגי חולון
Holon Institute of Technology

חלק 2.

שאלה 4 (20 נק')

בחרו את התשובה הנכונה **ונמקו** בכל אחד מהסעיפים הבאים (חובה לרשום את הבחירה והנימוק במחברת):

א. איזו מהמילים הבאות אינה מילה שמורה של שפת C?

(1) static
(2) global
(3) extern
(4) auto
(5) כל המילים לעיל הן מילים שמורות של שפת C

אין מילה global בשפת C. כדי לעשות משתנה גלובלי צריך פשוט להצהיר עליו מעל כל פונקציות.

ב. מהו פלט (משמאל לימין) של התוכנית הבאה?

```
#include <stdio.h>
void main() {
     int a[3][3] = { {1,2,3},{4,5,6},{7,8,9} }, i;
     for (i = 0; i < 3; i++)
          printf("%d %d ", *a[i], *(*(a + i) + i));
}
```

(1) 1 2 4 5 7 8
(2) 1 9 4 5 7 1
(3) 1 1 2 5 3 9
(4) 1 1 4 5 7 9
(5) התוכנית אינה תקינה והפעלתה תגרום לשגיאת קומפילציה או לשגיאת ריצה
(6) אף תשובה אינה נכונה

a[i]* שקול ל- a[i][0] ו- *(*(a + i) + i) שקול ל- a[i][i].

ג. מהי הטענה הלא נכונה?

(1) קובץ טקסט שנפתח באופן "r" מיועד לקריאה ולא מיועד לכתיבה
(2) קובץ טקסט שנפתח באופן "w" מיועד לכתיבה ולא מיועד לקריאה
(3) קובץ טקסט שנפתח באופן "r+" מיועד גם לקריאה וגם לכתיבה
(4) קובץ טקסט שנפתח באופן "w+" מיועד גם לקריאה וגם לכתיבה
(5) כל הטענות לעיל הן נכונות

לפי כללי שפת C.

הפקולטה למדעים
המחלקה למדעי המחשב
Faculty of Sciences
Department of Computer Science
גולומב 52, ת.ד 305, חולון 5810201
טלפקס: 03-5026528
52 Golomb St., Holon 5810201 Israel
Tel/Fax: 972-3-502-6528
www.hit.ac.il

ד. בתוכנית נתונה ההצהרה הבאה:

```
char *str = "hello";
str[0] = str[4] = '\0';
```

בחרו מה יקרה אחרי ביצוע פעולת

(1) ישתנה ערך של מצביע str
(2) תוכן המחרוזת str ישתנה ל- "hell"
(3) המחרוזת str תהפוך למחרוזת ריקה
(4) הפעולה תגרום לשגיאת ריצה
(5) אף תשובה אינה נכונה

ניסיון לשנות תוכן של מחרוזת קבועה גורם לשגיאת ריצה.

חלק 3.

שאלה 5 (בונוס – 10 נק')

מהו הפלט של התוכנית הבאה?

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

void main()
{
  printf("%d",strcmp("&","&&")&strcmp("&","&&")|("strcmp(&,&&)"&&"strcmp(&,&&)"));
}
```

יש לנמק את התשובה.

-1

המחרוזת "&" קטנה מילולית מהמחרוזת "&&". לכן strcmp("&","&&") מחזירה -1.
פעולת & (and) על סיביות שוות נותנת אותה סיבית. לכן -1 & -1 שווה ל- -1.
מצד ימין מבצעים פעולה בוליאנית && על שתי המחרוזות, ז"א על כתובות שלהן אשר שונות מ- NULL. הפירוש הלוגי של כל מה ששונה מאפס הוא true ולכן ערך מספרי של תוצאת הפעולה הוא 1.
בסוף מבצעים את הפעולה 1 | -1 . ייצוג בינארי של -1 הוא 111...1 וייצוג בינארי של 1 הוא 000...001. פעולת | (or) על סיביות 1 ו- 0 וגם על סיביות 1 ו- 1 נותנת סיבית 1. לכן ייצוג בינארי של התוצאה הוא 111...1 , ז"א -1.