שאלה 4 (16 נק')

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

א.

```
int (*p) (int *);
```

מה זה p?

(1) מצביע לפונקציה שמקבלת מצביע ל-int ומחזירה int
(2) פונקציה שמקבלת מצביע ל-int ומחזירה מצביע ל-int
(3) מצביע לפונקציה שמקבלת מצביע ל-int ומחזירה מצביע ל-int
(4) אף תשובה אינה נכונה

(1) – לפי התחביר של שפת C

ב.

```
z = x^y;
```

מה יהיה ערך של z?

(1) x בחזקת y
(2) המספר המתקבל מהזזת y סיביות שמאלה ב-x
(3) המספר המתקבל מהזזת y סיביות ימינה ב-x
(4) אף תשובה אינה נכונה

(4) – כי ^ זה XOR

ג.

```
static int c;
```

מה הוא ערך התחלתי של c?

(1) c לא מאותחל
(2) 0
(3) 1
(4) אף תשובה אינה נכונה

(2) – כי ברירת המחדל ערך התחלתי של משתנה static הוא 0

ד.

```
const int* q;
```

מה הוא q?

(1) מצביע קבוע למשתנה
(2) מצביע שיכול להשתנות לקבוע
(3) מצביע קבוע לקבוע
(4) אף תשובה אינה נכונה

(2) – לפי התחביר של שפת C

שאלה 4 (16 נק')

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

א. איזה סוג של מצביע **לא מוגדר** בשפת C?

(1) מצביע לפונקציה
(2) מצביע לקובץ
(3) מצביע למצביע
(4) מצביע לסיבית
(5) מצביע להכל
(6) כל הסוגים לעיל מוגדרים

(4) – לפי כללי שפת C

ב. מהי כמות הזיכרון שצריך מבנה כלשהו?

(1) היא שווה לסך הזיכרון שדורשים כל השדות שלו
(2) היא גדולה או שווה לסך הזיכרון שדורשים כל השדות שלו
(3) היא שווה לגודל מקסימאלי בין גדלים של כל השדות שלו
(4) אף תשובה אינה נכונה

(2) – סוגים בסיסיים של הנתונים מאוחסנים בכתובות שהן כפולות של הגדלים שלהם. לכן בין שדות של מבנה יכולים להיות פערים

ג. נתונה התכנית הבאה:

```
#include <stdio.h>

int x;
void main()
{
     x++;
     printf("%d",x);
}
```

מהי הטענה הנכונה?

(1) התכנית שגויה כי x לא מאותחל
(2) התכנית שגויה כי אסור להצהיר על משתנה מעל פונקציות
(3) התכנית שגויה משתי הסיבות לעיל
(4) התכנית תקינה
(5) התכנית תקינה אבל לא מומלץ להשתמש בסגנון ללא סיבה מוצדקת
(6) אף טענה אינה נכונה

(5) – x הוא משתנה גלובלי (לא מומלץ להשתמש) וברירת המחדל ערך התחלתי שלו הוא 0

ד. נתונה התכנית הבאה:

```
#include <stdio.h>

int size (double arr[])
{
     return sizeof(arr);
}

void main()
{
     double arr[10];
     printf ("%d\n", size(arr));
}
```

מהו פלט התכנית?

(1) 4
(2) 8
(3) 10
(4) 80
(5) אף תשובה אינה נכונה

(1) – כי arr הוא מצביע וגודל של מצביע שווה ל-4

ה. בתוכנית הוצהרו משתנה לוקאלי בשם x בתוך פונקציה main וגם משתנה גלובלי בשם x שמוכר בכל התוכנית. לאיזה משתנה ניגשים כאשר פונים ל-x בתוך main?

(1) למשתנה לוקאלי
(2) למשתנה גלובלי
(3) אין משמעות בשאלה כי משתנה לוקאלי ומשתנה גלובלי בעלי אותו שם תופסים אותו מקום בזיכרון
(4) תלוי במערכת ובמהדר
(5) אף תשובה אינה נכונה כי המהדר ייתן הודעה על שגיאה

כללי הטווח: לקטעים מקוננים עם משתנים בעלי אותו שם עדיפות בגישה – למשתנה הפנימי ביותר.

ו. עבור איזה משפט המהדר ייתן הודעה קשורה עם אי-התאמת טיפוסים?

(1) int n='a';
(2) char c=123;
(3) double x=38;
(4) int m=2000000000;
(5) float y=2.6;
(6) כל המשפטים לעיל הם תקינים

מספר ממשי הוא בברירת מחדל double. לכן המהדר נותן את האזהרה:

```
truncation from 'double' to 'float'
```

שאלה 4 (16 נק')

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

א. איזה פעולה לא מוגדרת למבנים?

(1) השמה
(2) לקיחת כתובת
(3) גישה לאיבר
(4) השוואה
(5) sizeof
(6) כל הפעולות לעיל מוגדרות

(4) – לפי כללי שפת C, כדי להימנע מהשוואת תכני המרווחים בין איברי המבנה

ב. מהי הטענה הנכונה?

(1) אחרי ביצוע פעולת הזזה שמאלה במספר, מצד ימין המספר מתמלא ב-0-ים
(2) אחרי ביצוע פעולת הזזה שמאלה במספר, מצד ימין המספר מתמלא ב-1-ים
(3) אחרי ביצוע פעולת הזזה שמאלה במספר, מצד ימין המספר מתמלא בסיביות שהלכו לאיבוד מצד שמאל
(4) אחרי ביצוע פעולת הזזה שמאלה במספר, שיטת מילוי המספר מצד ימין תלויה במערכת
(5) אף טענה אינה נכונה

(1) – לפי כללי שפת C

ג.

```
int x, *ptr1;
const int y = 5, *ptr2;
```

איזה פעולה תהיה בלתי חוקית?

(1) ptr1 = &x;
(2) ptr2 = &y;
(3) ptr1 = &y;
(4) ptr2 = &x;
(5) כל הפעולות לעיל חוקיות

(3) – כי גישה לקבוע דרך מצביע רגיל תאפשר לשנות את הקבוע

ד. איזו מילה היא לא חלק של הנחית קדם-מעבד?

(1) define
(2) ifdef
(3) undef
(4) ifndef
(5) typedef
(6) כל המילים לעיל הן חלקים של הנחיות קדם-מעבד

(5) – משתמשים ב-typedef להגדרת טיפוס חדש

שאלה 4 (16 נק')

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

א. מה ההבדל בין מבנה לאיגוד?

(1) לשדה של מבנה ניתן לפנות דרך מצביע למשתנה ולשדה של איגוד ניתן לפנות רק דרך שם המשתנה

(2) לשדה של איגוד ניתן לפנות דרך מצביע למשתנה ולשדה של מבנה ניתן לפנות רק דרך שם המשתנה

(3) לעומת מבנה, שדות איגוד משתפים אותו מקום בזיכרון

(4) לעומת איגוד, שדות מבנה משתפים אותו מקום בזיכרון

(5) אף תשובה אינה נכונה

(3) – לפי כללי שפת C

ב. מהי הטענה הנכונה?

(1) גודל מצביע ל-double עולה על גודל מצביע ל-char

(2) גודל מצביע ל-double שווה לגודל מצביע ל-char

(3) גודל מצביע ל-double קטן מגודל מצביע ל-char

(4) לא ניתן להשוות גדלי מצביעים ל-double ול-char

(5) אף טענה אינה נכונה

(2) – כי גדלי של כל מצביעים שווים

ג. מהו פלט של `printf("%d",~0)`?

(1) 1

(2) -1

(3) 0

(4) אף תשובה אינה נכונה

(2) – ייצוג בינארי של 0 הוא 000...0. לכן ייצוג בינארי של ~0 הוא 111...1 אשר מהווה ייצוג בינארי של -1

ד.

```
int *const *q;
```

מה הוא q?

(1) פוינטר לפוינטר קבוע שמצביע למשתנה

(2) פוינטר לפוינטר לקבוע

(3) פוינטר קבוע לפוינטר שמצביע למשתנה

(4) אף תשובה אינה נכונה

(1) – לפי כללי שפת C

חלק 2.

שאלות אמריקאיות (20 נק') **סטודנט יקבל ארבע שאלות**

בחר את התשובה הנכונה **ונמק** בכל אחד מהסעיפים.
התשובה הנכונה ללא נימוק או עם נימוק לא נכון תזכה רק בנקודה אחת.

א. נגדיר כי פעולה Y היא **הכללה** של פעולה X אם X מוגדרת אך ורק לחלק מהמקרים שבהם מוגדרת Y ולכל המקרים שבהם מוגדרת X התנהגותה של Y זהה **לחלוטין** להתנהגותה של X.
מהי הטענה הלא נכונה?

(1) fprintf היא הכללה של printf
(2) fscanf היא הכללה של scanf
(3) fgets היא הכללה של gets
(4) fgetc היא הכללה של getchar
(5) כל הטענות לעיל הן נכונות

הצבת stdout או stdin (בהתאם) ב-fprintf, fscanf, fgetc הופכת אותן מבחינת ההתנהגות ל-printf, scanf, getchar בהתאמה. fgets עם stdin לא זהה לחלוטין ל-gets כי ל-fgets יש עוד פרמטר למספר תווים מקסימלי ובנוסף היא קולטת גם '\n' לעומת gets.

ב. מה קורה בצד שמאל של מספר int שלילי לאחר ביצוע פעולת הזזת סיביות ימינה בתוכו?

(1) מצד שמאל המספר מתמלא ב-0-ים
(2) מצד שמאל המספר מתמלא ב-1-ים
(3) מצד שמאל המספר מתמלא בסיביות שהלכו לאיבוד מצד ימין
(4) שיטת מילוי המספר מצד שמאל תלויה במערכת
(5) אף תשובה אינה נכונה

לפי כללי שפת C.

ג.

```
const int **p;
```

מה הוא p?

(1) פוינטר לפוינטר קבוע שמצביע למשתנה
(2) פוינטר קבוע לפוינטר קבוע שמצביע למשתנה
(3) פוינטר קבוע לפוינטר שמצביע למשתנה
(4) פוינטר קבוע לפוינטר שמצביע לקבוע
(5) אף תשובה אינה נכונה

התשובה הנכונה: פוינטר לפוינטר שמצביע לקבוע.

ד. מהי הטענה הנכונה?

(1) משתנה מסוג static מוכר גם אחרי יציאה מהפונקציה בה הוא הוצהר
(2) משתנה מסוג static יכול להשתנות גם אחרי יציאה מהפונקציה בה הוא הוצהר
(3) משתנה מסוג static חייב להיות מאותחל באופן מפורש
(4) אחרי יציאה מהפונקציה בה הוצהר משתנה מסוג static ערך המשתנה נשמר בזיכרון ולא ניתן לשנות אותו
(5) שינוי משתנה מסוג static בשם x שהוצהר באחת מהפונקציות גורם לשינוי משתנה מסוג static בשם x שהוצהר בפונקציה אחרת
(6) אף טענה אינה נכונה

משתנה מסוג static לא מוכר אחרי יציאה מהפונקציה אבל נשמר בזיכרון. הפונקציה יכולה להחזיר כתובתו ודרכה אפשר לשנות את המשתנה גם אחרי יציאה מהפונקציה.

חלק 2.

שאלות אמריקאיות (20 נק')

בחר את התשובה הנכונה **ונמק** בכל אחד מהסעיפים הבאים.
התשובה הנכונה ללא נימוק או עם נימוק לא נכון תזכה רק בנקודה אחת.
"לפי כללי שפת C" הינו גם יכול להיות נימוק מקובל.

א. מה לא מאותחל כברירת המחדל?
(1) משתנה מסוג static
(2) משתנה גלובלי
(3) האיבר האחרון של מערך סטטי בו האיברים הראשונים מאותחלים באופן מפורש בהצהרה
(4) איברי מערך דינאמי שהוקצה באמצעות פונקציית calloc
(5) התו האחרון במערך תווים סטטי
(6) סמן המיקום הנוכחי בקובץ ברגע פתיחתו
(7) כל הנתונים שניתנו כאן מאותחלים כברירת המחדל

כברירת המחדל: משתנים static וגלובלי מאותחלים ב-0, איברים אחרונים במערך סטטי מאותחלים ב-0 אם איברים ראשונים מאותחלים במפורש, איברי מערך דינאמי שהוקצה באמצעות calloc מאותחלים ב-0, המיקום הנוכחי בקובץ הוא תחילת הקובץ בפתיחה לקריאה או לכתיבה וסוף הקובץ בפתיחה להוספה.
במקרה הכללי אף איבר של מערך סטטי לא מאותחל כברירת המחדל.

ב. מה קורה בצד שמאל של מספר unsigned לאחר ביצוע פעולת הזזת סיביות ימינה בתוכו?
(1) מצד שמאל המספר מתמלא ב-0-ים
(2) מצד שמאל המספר מתמלא ב-1-ים
(3) מצד שמאל המספר מתמלא בסיביות שהלכו לאיבוד מצד ימין
(4) שיטת מילוי המספר מצד שמאל תלויה במערכת
(5) אף תשובה אינה נכונה

לפי כללי שפת C.

ג. מהו "abcd" - "xyz" ?
(1) מחרוזת
(2) כתובת
(3) מספר שלם
(4) הפעולה היא בלתי חוקית מבחינת המהדר
(5) אף תשובה אינה נכונה

מחרוזת כבועה במשפט היא כתובת של התו הראשון שלה. לפי כללי אריתמטיקת מצביעים הפרש הכתובות זה מספר שלם.

ד. נניח שבן-אדם פותח קובץ בינארי באמצעות עורך טקסט. מהי הטענה הנכונה לגבי נתונים הנשמרים בקובץ?

*** "<u>לא</u> ניתן לקריאה על ידי בני אדם" פירושו: אדם יראה ג'יבריש ("קשקוש").**

*** "ניתן לקריאה על ידי בני אדם" פירושו: אדם יוכל לראות את הערך שבו מדובר ולא ג'יבריש.**

(1) מספרים שלמים ניתנים לקריאה ע"י בן-אדם כאשר תווים לא ניתנים לקריאה ע"י בן-אדם

(2) מספרים שלמים לא ניתנים לקריאה ע"י בן-אדם כאשר תווים ניתנים לקריאה ע"י בן-אדם

(3) גם מספרים שלמים וגם תווים לא ניתנים לקריאה ע"י בן-אדם

(4) גם מספרים שלמים וגם תווים ניתנים לקריאה ע"י בן-אדם אם רק מספרים שלמים או רק תווים נשמרים בקובץ אולם הם לא ניתנים לקריאה אם מופיעים בקובץ ביחד

(5) גם מספרים שלמים וגם תווים תמיד ניתנים לקריאה ע"י בן-אדם

קובץ בינארי הוא רצף בתים (כמו בזיכרון פנימי). מצד שני תו זה בית אחד ולכן כל בית מופיע כתו. אזי בן-אדם יראה את התווים כמו שהם אבל מספרים אראה כקשקוש.

שאלה 4 (20 נק')

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

א. מה מאותחל כברירת המחדל?

(1) משתנה מסוג int שמוגדר בתוך פונקציה, למשל `int x;`

(2) משתנה מסוג פוינטר ל-int שמוגדר בתוך פונקציה, למשל `int *p;`

(3) קבועים של טיפוס enum

(4) משתנה בקרה בלולאת for אשר לא מאותחל במפורש בחלק הראשון של הלולאה, למשל `for( ; i<n; i++)`

(5) איברי מערך דינאמי שהוקצה באמצעות פונקציית malloc

(6) אף תשובה אינה נכונה

ערכים של הקבועים של enum נקבעים אוטומטית, מתחילים מ-0 וגדלים ב-1.

ב. מה קורה בצד שמאל של מספר int חיובי לאחר ביצוע פעולת הזזת סיביות ימינה בתוכו?

(1) מצד שמאל המספר מתמלא ב-0-ים

(2) מצד שמאל המספר מתמלא ב-1-ים

(3) מצד שמאל המספר מתמלא בסיביות שהלכו לאיבוד מצד ימין

(4) שיטת מילוי המספר מצד שמאל תלויה במערכת

(5) אף תשובה אינה נכונה

לפי כללי שפת C.

ג. איזה מערך מועבר לפונקציה by value (ז"א הפונקציה מקבלת גישה להעתק של תוכנו אבל לא לתוכן המקורי שלו)?

(1) מערך סטטי חד-ממדי

(2) מערך דינאמי חד-ממדי

(3) מערך סטטי דו-ממדי

(4) מערך דינאמי דו-ממדי

(5) מערך סטטי אשר מהווה שדה של המבנה כאשר המבנה מועבר לפונקציה by value

(6) אף תשובה אינה נכונה

מערך עצמו תמיד מועבר by reference כי הפונקציה מקבלת לא מערך עצמו אלא כתובתו. אבל מבנה מועבר by value כך שכל השדות שלו מועתקים כולל שדות שהם מערכים.

ד. נתונה הפונקציה הבאה:

```
int *fun() {
     static int x;
     x++;
     return &x;
}
```

מהי הטענה הנכונה?

(1) הפונקציה לא תקינה רק מכיוון שהיא מחזירה כתובת של משתנה אשר לא קיים מעבר לפונקציה

(2) הפונקציה לא תקינה רק מכיוון שניגשים בתוכה למשתנה אשר לא מאותחל

(3) הפונקציה לא תקינה גם מכיוון שהיא מחזירה כתובת של משתנה אשר לא קיים מעבר לפונקציה וגם מכוון שניגשים בתוכה למשתנה אשר לא מאותחל

(4) הפונקציה תקינה

(5) אף טענה אינה נכונה

משתנה static נשמר בזיכרון גם אחרי יציאה מהפונקציה. משתנה static ללא אתחול מפורש מאותחל ב-0.

שאלה 4 (20 נק')

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

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

```
#include <stdio.h>

int x;

void f() {
     static int x;
     printf("%d", x);
     printf("%d", ++x);
}

void main() {
     f();
     printf("%d", x);
     printf("%d", ++x);
     f();
}
```

(1) 010101

(2) 000000

(3) 000011

(4) 010112

(5) 011223

(6) התוכנית אינה תקינה

(7) אף תשובה אינה נכונה

כברירת מחדל ערך התחלתי של static x הוא 0. לכן בקריאה ראשונה של f() מדפיסים 0, מקדמים x ומדפיסים 1. אחרי מדפיסים ערך התחלתי של x גלובלי ששווה ל-0, מקדמים x גלובלי ומדפיסים 1. בקריאה שנייה של f() מדפיסים static x שערכו נישאר 1, מקדמים x ומדפיסים 2.

ב. כמה פעמים יודפס Hi עקב ביצוע התוכנית הבאה?

```
#include <stdio.h>
void main() {
     int i = 64;
     for (; i; i >>= 1)
          printf("Hi \n");
}
```

(1) פעם אחת
(2) 7 פעמים
(3) 64 פעמים
(4) אף פעם
(5) תהיה לולאה אינסופית
(6) אף תשובה אינה נכונה

ייצוג בינארי של 64 הוא 1000000 (6 אפסים, 7 סיביות). כל פעולה i >>= 1 מביאה להזזה של 1 בסיבית אחת ימינה. לכן תוך 7 בדיקות (6 הזזות) סיבית 1 תהיה עוד במספר i, ערך של i יהיה שונה מ-0 ותנאי המשך הלולאה יתקיים. אחרי הזזה שביעית 1 יצאה מייצוג בינארי של i אשר יהיה מורכב רק מאפסים, ולכן ערך עשרוני שלו יהיה 0 כך שתנאי המשך הלולאה בבדיקה שמינית לא יתקיים.

ג. נתון פוינטר ל-int p שמצביע למערך דינאמי. למה יצביע p אחרי ביצוע פעולת

```
p = (int*)realloc(p, 0);?
```

(1) לאותו מערך
(2) לאיזשהו מקום חדש
(3) p יהיה שווה ל- NULL
(4) הפעולה תגרום לשגיאת ריצה
(5) אף תשובה אינה נכונה

לפי כללי שפת C כאשר גודל חדש של בלוק זיכרון ב-realloc שווה ל-0, הפונקציה מחזירה NULL.

ד. בתוכנית בה a מוגדר כמשתנה מסוג int ו-fptr1 ו-fptr2 הם פוינטרים לקבצים בינאריים קיימות שתי השורות הבאות:

```
fread(&a, sizeof(a), 1, fptr1);
fwrite(&a, sizeof(a), 1, fptr2);
```

מהי הטענה הנכונה?

(1) ברשימת פרמטרי fread בשורה הראשונה יש סימן & מיותר והשורה השנייה תקינה
(2) ברשימת פרמטרי fwrite בשורה השנייה יש סימן & מיותר והשורה הראשונה תקינה
(3) ברשימות פרמטרי fread ו-fwrite יש סימן & מיותר
(4) שתי השורות תקינות
(5) באחת מהשורות או בשתי השורות סימן & נמצא במקום לא נכון
(6) אף טענה אינה נכונה

גם fread_ וגם fwrite מקבלות כפרמטר מצביע לבלוק הזיכרון. לכן צריך להציב כתובת של המשתנה.

חלק 2.

שאלה 4 (20 נק')

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

א. נתון קובץ טקסט בעל התוכן הבא: **5 4 3 2 1** (משמאל לימין). אחרי כל אחד מחמישה המספרים (גם אחרי 5) יש רווח. מיד אחרי הפתיחה המוצלחת של הקובץ באמצעות מצביע fptr מופיע הקטע הבא בתוכנית (x הוא משתנה מסוג int):

```
while (!feof(fptr)) {
     fscanf(fptr, "%d", &x);
     printf("%d ", x);
}
```

כמה פעמים מבוצע גוף הלולאה?

(1) 5 פעמים
(2) 4 פעמים
(3) מספר אינסופי פעמים
(4) פעם אחת
(5) אף תשובה אינה נכונה

הקוד מאורגן בצורה לא תקינה – ללא קלט לפני בדיקת סוף הקובץ. כתוצאה אחרי טיפול בחמישה המספרים השמורים ניכנס לתוך הלולאה גם בפעם השישי כאשר נהיה כבר בסוף הקובץ.

ב.

```
double (**q) ();
```

על מה מצביע פוינטר q?

(1) על פוינטר ל- double
(2) על פונקציה
(3) על פוינטר לפונקציה
(4) על פוינטר לפוינטר ל- double
(5) q זה לא פוינטר אלא פונקציה
(6) אף תשובה אינה נכונה

לפי תחביר שפת C, q זה פוינטר לפוינטר לפונקציה ללא פרמטרים אשר מחזירה double. ז"א, q מצביע על פוינטר לפונקליה.

ג. נתון פוינטר ל-int בשם p. בחרו מה יקרה אחרי ביצוע פעולת

```
p = (int*)malloc(0);
```

(1) malloc תקצה ביית אחד בעל אינדקס 0
(2) p יצביע על איזשהו מקום
(3) p יהיה שווה ל- NULL
(4) הפעולה תגרום לשגיאת ריצה
(5) אף תשובה אינה נכונה

במידה והפרמטר של malloc שווה ל- 0 , הפונקציה לא מקצה זיכרון אבל לא מחזירה NULL.

ד. בתוכנית נתונה הצהרה הבאה:

```
char *str = "hello";
str = "good-bye";
```

בחרו מה יקרה אחרי ביצוע פעולת

(1) ישתנה ערך של מצביע str
(2) תוכן המערך str ישתנה ל- "good-bye"
(3) תוכן המערך str ישתנה ל- "good-"
(4) תוכן המערך str ישתנה ל- "good-byehello"
(5) הפעולה תגרום לשגיאת ריצה
(6) אף תשובה אינה נכונה

לא מוגדר פה מערך ו- str הוא מצביע. עבור הגדרה זו של str המחשב מקצה שטח זיכרון ל- str ולמחרוזת "hello" ומציב ל- str את כתובת התו הראשון של המחרוזת.

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.

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.

שאלה 5 (10 נק')

בחרו את התשובה הנכונה **ונמקו** בכל אחד מהסעיפים הבאים:

א. מהי הטענה הנכונה?

(1) הקריאות puts("abc") ו- fputs("abc",stdout) הן שקולות

(2) הקריאות puts("abc") ו- fputs("abc\n",stdout) הן שקולות

(3) הקריאות puts("abc\n") ו- fputs("abc",stdout) הן שקולות

(4) הקריאות puts("abc") ו- fputs("\nabc",stdout) הן שקולות

(5) אף טענה אינה נכונה

נימוק:

לעומת puts, fputs לא מוסיפה '\n' בסוף המחרוזת.
לכן fputs("abc",stdout) מציגה במסך abc ללא הורדת שורה.
בזאת fputs("abc\n",stdout) מציגה במסך abc ומורידה סמן לשורה הבאה, בדיוק כמו puts("abc").

ב. איזה פעולה היא בלתי חוקית?

(1) של מספר שלם למצביע

(2) חיסור של מספר שלם ממצביע

(3) חיבור של מצביע למצביע

(4) חיסור של מצביע ממצביע

(5) השוואת מצביעים

(6) כל הפעולות לעיל הן חוקיות

נימוק:

בשפת C מוגדרות הפעלות הבאות על מצביעים:
חיבור (חיסור) מספר שלם למצביע (התוצאה היא כתובת),
חיסור מצביעים (התוצאה היא מספר שלם),
השוואת מצביעים.
פעולת חיבור מצביעים לא מוגדרת.

שאלה 5 (10 נק')

בחרו את התשובה הנכונה **ונמקו** בכל אחד מהסעיפים הבאים:

א. איזה פעולה לא תיתן פלט **0**?

(1) `printf("%d\n", 0);`

(2) `printf("%.0lf\n", 0.0);`

(3) `printf("%d\n", NULL);`

(4) `printf("%d\n", '0');`

(5) `printf("%d\n", '\0');`

(6) `printf("%d\n", EOF + 1);`

(7) כל הפעולות לעיל יתנו פלט **0**

<u>נימוק:</u>

---

(1) מדפיסים מספר שלם 0.

(2) מדפיסים מספר ממשי 0.0 עם אפס ספרות אחרי הנקודה, ז"א מדפיסים רק 0.

(3) NULL הוא שם של קבוע 0. לכן מדפיסים 0.

(4) קוד ASCII של תו '0' הוא המספר ששונה מ-0. לכן מדפיסים לא 0.

(5) קוד ASCII של תו '\0' הוא 0. לכן מדפיסים 0.

(6) קוד ASCII של תו EOF הוא 1-. לכן מדפיסים ערך של 1+1-, ז"א 0.

ב. נתונה התוכנית הבאה:

```
#include <stdio.h>

int f(int n) {
     int m = 2 * n;
     return m;
}

void main() {
     int k = 3;
     printf("%d\n", f(k));      // 1
     printf("%d\n", (*f)(k));   // 2
}
```

מה ההבדל בין הפעולה המסומנת ב- 1 לפעולה המסומנת ב- 2?

(1) פעולה 1 תציג ערך של משתנה לוקלי m ואילו פעולה 2 תציג את הכתובת של m

(2) פעולה 1 תציג ערך המוחזר ע"י הקריאה f(k) ואילו פעולה 2 תציג את הערך מוכפל בעצמו

(3) פעולה 1 תציג ערך המוחזר ע"י הקריאה f(k) ואילו פעולה 2 תציג את הערך מוכפל ב- k

(4) פעולה 2 בנוסף למה שמבצעת פעולה 1 גם משנה את הערך של k

(5) פעולה 2 פשוט לא תקינה

(6) פעולות 1 ו- 2 שקולות

<u>נימוק</u>

שם של פונקציה הוא למעשה שם הפוינטר שמצביע לתחילת הפונקציה. לכן f הוא פוינטר שמצביע לתחילת הפונקציה שלנו. ובכך, בפעולה 2 פונים לאותה פונקציה כמו בפעולה 1 אך במפורש דרך הפוינטר.