מדעי המחשב – 2 יחידות לימוד
פתרון בחינת הבגרות

פרק א

שאלה 1:

קלט: 4 , 8 , 5 , 3 ⇨

פלט	b זוגי?	b	k ≤ a	k	mul	a
40					1	3
	F	5	T	1	5	
	T	8	T	2	20	
	T	4	T	3	40	
			F	4		

שאלה 2:

פלט	50≤x≤70	x ≥ 0	x	m	
2				0	א.
	F	T	10		
	T	T	64	1	
	T	T	51	2	
	F	T	30		
		F	-4		

ב. -1 (כל מספר ראשון שלילי בקלט)

ג. -1 , 80 , 10 ⇨ (כל המספרים בקלט קטנים מ- 50 או גדולים מ- 70, לפחות מספר אחד לא שלילי בהתחלה.)

שאלה 3:

קטע תכנית: בכל תא במערך יושם מספר אקראי בין 5 ל- 12 (כולל).

```
randomize;
count := 0;
for  i := 1 to 36 do
        begin
                numArr [i] := random (8) + 5;
                if (numArr[i] = 8) then
                        count := count + 1 ;
        end;
writeln ( 'מספרים השווים ל- 8 ', count, ' הוגרלו ');
```

שאלה 4:

function **test** (x, y : integer; z : real) : integer;

i. x := test (3, 7, 8.2);	תקין.
ii. t := test (b, b, 2.5);	תקין.
iii. c := test (7.8);	לא תקין. חסרים שני פרמטרים.
iv. m := test (4, 6.25, 9);	לא תקין. סדר הפרמטרים אינו תואם את כותרת הפעולה.

v. if (test (2, 4, 6.1) = 1) then
begin
k := 7;
end;
	תקין.

שאלה 5:

דרך א: עבור כל תו במחרוזת שהוא רווח, אם התו שלפניו הוא 'y' נוסיף 1 למונה.
דרך ב: עבור כל תו 'y' במחרוזת, אם התו שאחריו הוא תו רווח, נוסיף 1 למונה.

דרך א:

```
readln (str);
count := 0;
for i := 2  to length(str) do
        if (str[i]  = ' ')  and  (str[i-1] = 'Y') then
                count := count + 1;
writeln (count);
```

דרך ב:

```
readln (str);
count := 0;
for i := 1  to length(str)-1 do
        if (str[i]  = 'Y')  and  (str[i+1] = ' ') then
                count := count + 1;
writeln (count);
```

פרק ב

שאלה 6:

```
{ קלט: טמפרטורה יומית בתקופה מסויימת. סיום הקלט בטמפ' גבוהה מ- 100 מעלות.
  פלט: מספר הימים המאקסימלי שעבר בין שתי מדידות טמפ' הזהה לטמפ' שנמדדה ביום הראשון.
  אם לא נמדדה שוב טמפ' זהה לטמפ' של היום הראשון, יוצג 1- }

program T6_bag07;
var     count, max : integer;
        firstTemp, temp : integer;

begin
        count := 0;  max := -1;

        write ('טמפרטורה של היום הראשון ? → ');
        read (firstTemp);

        write ('טמפרטורה של היום הבא ? →');
        read (temp);

        while (temp <= 100) do
                begin
                        if (temp <> firstTemp) then
                                count := count + 1
                        else
                                begin
                                        if (count > max) then
                                                max := count;
                                        count := 0;
                                end;
                        write ('טמפרטורה של היום הבא ? →');
                        read (temp);
                end;

        if (max >= 0) then
                writeln (max, ' מספר הימים המאקסימלי בין שתי מדידות:')
        else
                writeln (max);
end.
```

שאלה 7:

א.

	1	2	3	4	5	6	7	8	9
a	3	6	-12	-8	-37	-6	2	-9	13

n	ans	i	i < 8 i < n-1	a[i]	a[i+1]	a[i+2]	I a[i+1]>a[i]	II a[i+1]>a[i+2]	I and II
9	0	1	T	3	6	-12	T	T	T
	1	3	T	-12	-8	-37	T	T	T
	2	5	T	-37	-6	2	T	F	F
		7	T	2	-9	13	F	F	F
		9	F						

ב.

	0	1	2	3	4	5	6	7	8
a	3	**6**	-12	**-8**	-37	-6	2	**18**	13

טענת יציאה (לא נדרש בבחינה): קטע הקוד סופר כמה איברים במערך גדולים משני שכניהם (שכן בכל צד).

שאלה 8:

א. פעולה (תת-תכנית):

```
{       טענת כניסה: קוטר של פקק תקני ומספר הפקקים שמייצרת מכונה
        טענת יציאה: מספר הפקקים התקינים שייצרה המכונה
        פקק תקין הוא פקק שההפרש בערך מוחלט בינו לבין
        פקק בקוטר תקני קטן מ- 1 מ"מ.                    }

function check (diameter, num : integer) : integer;
var
        count, plugDiameter, i : integer;
begin
        count := 0;

        for i := 1 to num do
                begin
                        write (' הקש קוטר פקק → ');
                        readln (plugDiameter);
                        if (abs(plugDiameter - diameter) <= 1)  then
                                count := count + 1 ;
                end;

        check := count;
end;
```

ב. קטע התכנית:

```
        for i := 1 to  50 do
                begin
                        write (i, ' קוטר תקני של מכונה → );
                        read (standardDiameter);
                        write ('  מספר הפקקים שיוצרו במכונה  →  ');
                        read (num);

                        count := check (standardDiameter, num);

                        writeln (' פקקים תקינים ', count , ' ייצרה ', i, ' מכונה ');
                end;
```

פרק ג

שאלה 9:

למעשה זוהי שאלה של רשומות/מבנים/עצמים. נציג שני פתרונות: פתרון ללא רשומות ופתרון עם רשומות.

פתרון ללא רשומות:

טבלת משתנים:

שם משתנה	טיפוס משתנה	תפקיד המשתנה
p	מערך בגודל 318 של שלמים	תוכן מקום חניה - שעה ההגעה של מכונית - מספר שלם. תא המכיל 0 מסמל מקום חניה פנוי.
cash	שלם	קופת החניון
arrivalTime	מספר שלם	שעת ההגעה לחניון (בתחום 6 - 22)
departTime	מספר שלם	שעת עזיבה של החניון (בתחום 7 - 23)
code	מספר שלם	קוד המכונית: 1 - כניסה לחניון. 2 - עזיבת החניון.
carPlace	מספר שלם	מציין את מספר התא בו חונה הרכב (בתחום 1-318)

תת-בעיות:

אתחול-חניון (p, cash) procedure **parkingInit** (var p : arr_Type; var cash : integer);

פעולה המקבלת את מערך החניון p ואת הקופה cash ומאתחלת את הקופה ואת כל איברי המערך ל- 0.

כניסת-רכב (p, arrivalTime) procedure **arrivalCar** (var p : arr_Type; arrivalTime : integer);

טענת כניסה: מערך החניון ושעת ההגעה של כלי רכב (מספר שלם בין 6 ל- 22).
טענת יציאה: הפעולה מוצאת מקום חניה פנוי במערך החניון, ומסמנת אותו כתפוס על ידי רישום שעת ההגעה של הרכב.

יציאת-רכב (p, departTime, carPlace, cash)
procedure **departCar** (var p : arr_Type; departTime, carPlace : integer; var cash : integer);

טענת כניסה: מערך החניון, שעת העזיבה של כלי רכב (מספר שלם בין 7 ל- 23), מקום החניה (מספר התא במערך (מספר שלם בין 1 ל- 318) וקופת החניון.
טענת יציאה: הפעולה מחשבת את עלות החניה, מעדכנת את הקופה ומסמנת את המקום כפנוי.

סגירת-חניון (p) procedure **parkingClose** (cash : integer);

טענת כניסה: מערך החניון.
טענת יציאה: הדפסת סך כל הכסף שנגבה במשך היום בעבור חניית המכוניות בחניון.
הנחה: החניון ריק ממכוניות.

ניהול חניון מכוניות:
קלט - קוד ושעת הגעה או עזיבה.
פלט - סך ההכנסות ליום זה

```
program Parking;
const
        N = 318;
type
        arrType = array [1..N] of integer;

var
        p : arr_Type;
        cash, code : integer;
        arrivalTime, departTime, carPlace : integer;


{       פעולה המאתחלת את כל איברי המערך ל- 0.   }
procedure parkingInit (var p : arr_Type; var cash : integer);
var     i : integer;
begin
        for i := 1 to N do
                p[i] := 0;
        cash := 0;
end;

{                       מערך החניון ושעת ההגעה של כלי רכב (מספר שלם בין 6 ל- 22).   :טענת כניסה
    הפעולה מוצאת מקום חניה פנוי במערך החניון, ומסמנת אותו כתפוס על ידי רישום   :טענת יציאה
                                                שעת ההגעה של הרכב.                    }

procedure arrivalCar (var p : arr_Type; arrivalTime : integer);
var     i : integer;
begin
        i := 1;
        while (i <= N)  and  (p[i] <> 0)  do
                i := i + 1;
        if (i <= N) then
                begin
                        p[i] := arrivalTime;
                        writeln (' Car is parking in place ', i);
                end
        else
                writeln ('Parking is full');
end;

{          מערך החניון, שעת העזיבה של כלי רכב (מספר שלם בין 7 ל- 23) ומקום החניה   :טענת כניסה
                                    (מספר התא במערך (מספר שלם בין 1 ל- 318).
        הפעולה מחשבת את עלות החניה, מעדכנת את הקופה ומסמנת את המקום כפנוי.   :טענת יציאה   }

procedure departCar (var p : arr_Type; departTime, carPlace : integer; var cash : integer);
var     toPay : integer;
begin
```

```
        toPay := (departTime - p[carPlace]) * 14;
        writeln (' payment for this car : ', toPay , ' nis');
        cash := cash + toPay;
        p[carPlace] := 0;
end;
{                                                    מערך החניון.    :טענת כניסה
        הדפסת סך כל הכסף שנגבה במשך היום בעבור חניית המכוניות בחניון.    :טענת יציאה
                                                    החניון ריק ממכוניות.    :הנחה    }

procedure parkingClose (cash : integer)
begin
        writeln (' total parking money is: ', cash);
end;
{--- התכנית הראשית ---}
begin
        parkingInit (p);
        writeln (' type 1 for car arrival, 2 for car departure ');
        write (' type -1 to end day → );
        read (code);
        while (code <> -1)  do
                begin
                        if (code = 1)  then
                                begin
                                        write (' type time of arriving (6 to 22) → );
                                        read (arrivalTime);
                                        arrivalCar (p, arrivalTime);
                                end
                        else    { code = 2 }
                                begin
                                        write (' type departure time (7 to 23) → );
                                        read (departTime);
                                        write (' type number of car location → ');
                                        read (carPlace);
                                        departCar (p, departTime, carPlace, cash);
                                end;
                        writeln (' type 1 for car arrival, 2 for car departure ');
                        write (' type -1 to end day → );
                        read (code);
                end;
        parkingClose (cash);
end.
```

פתרון עם רשומות:

הפתרון מורכב משתי רשומות: ParkingPlace המגדירה מקום חניה בודד
ו- Parking המגדירה את מגרש החניה.

טבלת משתנים:

שם משתנה	טיפוס משתנה	תפקיד המשתנה
מקום חניה: arrivalTime avail	עצם שתכונותיו: שעת-הגעה - שלם פנוי? - בוליאני	
park: cash p	עצם שתכונותיו: קופה - שלם מערך - של עצמים	מגרש החניה - מערך שכל אחד מאיבריו הוא מטיפוס **מקום-חניה**.
arrivalTime	מספר שלם	שעת ההגעה לחניון (בתחום 6 - 22)
departTime	מספר שלם	שעת עזיבה של החניון (בתחום 7 - 23)
code	מספר שלם	קוד המכונית: 1 - כניסה לחניון. 0 - עזיבת החניון.
carPlace	מספר שלם	מציין את מספר התא בו חונה הרכב (בתחום 1-318)

חלוקה לתת-בעיות:

אתחול-חניון (park) procedure **ParkingInit**(var park : Parking);

פעולה בונה המחזירה מערך חניות מאותחל וקופה מאופסת.

כניסת-רכב (park, arrivalTime) procedure **arrivalCar** (var park : Parking; arrivalTime : integer);

טענת כניסה: שעת ההגעה של כלי רכב (מספר שלם בין 6 ל- 22).
טענת יציאה: הפעולה מוצאת מקום חניה פנוי במערך החניון, מסמנת אותו כתפוס ורושמת את שעת ההגעה של הרכב.

יציאת-רכב (park, departTime, carPlace)
procedure **departCar** (var park : Parking; departTime, carPlace : integer);

טענת כניסה: שעת העזיבה של כלי רכב (מספר שלם בין 7 ל- 23) ומקום החניה (מספר התא במערך (מספר שלם בין 0 ל- 317 כולל).
טענת יציאה: הפעולה מחשבת את עלות החניה, מעדכנת את הקופה ומסמנת את המקום כפנוי.

סגירת-חניון (p) procedure **parkingClose** (park : Parking);

טענת יציאה: הדפסת סך כל הכסף שנגבה במשך היום בעבור חניית המכוניות בחניון.
הנחה: החניון ריק ממכוניות.

ניהול חניון מכוניות:
קלט - קוד ושעת הגעה או עזיבה.
פלט - סך ההכנסות ליום זה

```
program Yesodot_2007_Parking;
const   N = 5;
type
        parkingPlace = record
                arrivalTime : integer;
                avail : boolean;
        end;

        Parking = record
                cash : integer;
                p : array [1..N] of parkingPlace;
        end;

var
        park : Parking;
        code, arrivalTime, departTime, carPlace : integer;

{       פעולה בונה המחזירה מערך חניות מאותחל וקופה מאופסת.       }
procedure ParkingInit(var park : Parking);
var i : integer;
begin
        park.cash := 0;
        for i := 1 to N do
                begin
                        park.p[i].arrivalTime := 0;
                        park.p[i].avail := true;
                end;
end;
```

park - משתנה מטיפוס חניון.
park.cash - קופת החניון.
park.p - מערך מקומות החניה.
park.p[i] - מקום חניה בודד.
park.p[i].arrivalTime - שעת ההגעה של רכב לחניה ה- i
park.p[i].avail - האם מקום חניה i פנוי או תפוס

```
{       פעולה המציגה את מצב מגרש החניה לאחר כל כניסה או יציאת רכב
                                הערה: פעולה זו לא נדרשה בבחינה.       }
procedure parkingShow(park : Parking);
var i : integer;
begin
        for i := 1 to N do
                begin
                        if (park.p[i].avail) then
                                writeln('place #', i, '  Avail: ', park.p[i].avail)
                        else
                                begin
                                        write('place #', i, '  Avail: ', park.p[i].avail);
                                        writeln ('    Arriving Time:  ', park.p[i].arrivalTime);
                                end;
                end;
        writeln;
end;
```

```
{                                   פעולה המטפלת בכניסת רכב לחניון.
        הפעולה מוצאת מקום חניה פנוי במערך החניון, מסמנת אותו כתפוס ורושמת את
                                    שעת ההגעה של הרכב.                    }
procedure arrivalCar (var park : Parking; arrivalTime : integer);
var i : integer;
begin
        i := 1;
        while (i < N)  and  (not park.p[i].avail)  do
                i := i + 1;
        if (i < N) then
                begin
                        park.p[i].arrivalTime := arrivalTime;
                        park.p[i].avail := false;
                        writeln ('Car is parked in place ', i);
                end
        else
                writeln ('Parking is full');
end;

{                                   פעולה המטפלת ביציאת רכב מהחניון.
        הפעולה מחשבת את עלות החניה, מעדכנת את הקופה ומסמנת את המקום כפנוי    }
procedure departCar (var park : Parking; departTime, carPlace : integer);
var toPay : integer;
begin
        toPay := (departTime - park.p[carPlace].arrivalTime) * 14;
        writeln ('payment for this car : ', toPay, ' nis ');
        park.cash := park.cash + toPay;
        park.p[carPlace].arrivalTime := 0;
        park.p[carPlace].avail := true;
end;

{       פעולה המציגה את סך הכסף שנאסף החניון בסיום יום העבודה       }
procedure parkingClose (park : Parking);
begin
        writeln ('total parking money is: ', park.cash);
end;
```

```
{       main program  }
begin
        parkingInit (park);

        writeln (' type 1 for car arrival, 2 for car departure ');
        write (' type -1 to end day  → ');
        read (code);
        while (code <> -1)  do
                begin
                        if (code = 1)  then
                                begin
                                        write (' type time of arriving (6 to 22) → ');
                                        read (arrivalTime);
                                        arrivalCar (park, arrivalTime);
                                        parkingShow(park);
                                end
                        else    { code = 2 }
                                begin
                                        write (' type departure time (7 to 23) → ');
                                        read (departTime);
                                        write (' type number of car location → ');
                                        read (carPlace);
                                        departCar (park, departTime, carPlace);
                                        parkingShow(park);
                                end;

                        writeln (' type 1 for car arrival, 2 for car departure ');
                        write (' type -1 to end day → ');
                        read (code);
                end;

        parkingClose (park);
end.
```

שאלה 10:

```
{                תכנית המטפלת במערך. אם אין במערך לפחות 7 תאים רציפים המכילים 0
        תבצע התכנית הזזות במערך עד אשר יהיו בסופו לפחות 7 תאים רציפים המכילים 0.      }
program Targil10;

const N = 105;

type    arrType = array [1..N] of integer;

var     a : arrType;
        k : integer;

{---                                                    ט.כניסה: מערך a  ---
 ט.יציאה: מוחזר 1 אם קיימים לפחות 7 תאים רציפים המכילים 0, ו- 0 אחרת ---}
function seven (a : arrType) : integer;
var     i, zero;
begin
        zero := 0;

        for i := 1 to N do
                begin
                        if (a[i] = 0) then
                                zero := zero + 1
                        else
                                if (zero < 7)  then
                                        zero := 0;
                end;

                if (zero >= 7) then
                        seven := 1
                else
                        seven := 0;
end;


{---                                    ט.כניסה: מערך ומספר שלם (בין 1 ו- 4)  ---
 ---                        ט.יציאה: כל האיברים במערך זזים k מקומות שמאלה.  ---
 --- k האיברים הראשונים "גולשים". k האיברים האחרונים מתאפסים  ---}
procedure shift (var a : arrType ; k : integer);
var     i : integer;
begin
        for i := 1 to N - k  do
                a[i] := a[i+k];

        for i := (N - k +1)  to N do
                a[i] := 0;
end;
```

```
{--- הדפסת המערך ---}
procedure arrPelet (a : arrType):
var     i : integer;
begin
        for i := 1 to N do
                write (a[i], '   ');
        writeln ();
end;

{--- התכנית הראשית ---}
begin
        {   כאן יבוא קטע הקלט למערך    }
        while (seven(a) = 0)  do
                begin
                        write(' type a number between 1 to 4 → ');
                        read (k);
                        shift (a, k);
                end;
        arrPelet (a);
end.
```