

ארגון המחשב ושפת סף (203.1130)
סמסטר ב' תשנ"ט
בחינה סופית - מועד א'

הוראות לנבחן:

- משך הבחינה שלוש שעות.
- מותר להשתמש בכל חומר עזר, למעט מחשבים מכל סוג.
- יש להשיב על כל השאלות.
- יש לרשום את התשובות בגוף השאלון במקומות המיועדים לכך.
- נא לכתוב בכתב יד ברור ונקי. מומלץ להשתמש בעפרון ומחק.
- בשאלון זה 12 דפים, כולל דף זה. ודא כי כל הדפים נמצאים.

ב ה צ ל ח ה !

ציון	ניקוד	
18	25	שאלה 1
24	25	שאלה 2
24	25	שאלה 3
26	25	שאלה 4
92	100	סה"כ

א. להלן מספר זוגות של מושגים או פקודות בשפת אסמבלי. עבור כל זוג, הסבר בקצרה את ההבדל המהותי בין שני המושגים או הפקודות.

i. שיטת המספרים ללא סימן

שיטת המשלים ל-2

ההבדל המהותי הוא בסימן. בשיטת המספרים ללא סימן, ניתן לייצג מספרים חיוביים (כולל 0) ואלו שליליים. בשיטת המשלים ל-2, ניתן לייצג מספרים חיוביים ושליליים (כולל 0).

ii. public x אובייקט מוכרז הכולל x, וניתן לכתוב אליו מידע מאיזן אחר.

extrn x אובייקט מוכרז הכולל x, אך לא ניתן לכתוב אליו מידע מאיזן אחר. - x אובייקט מוכרז הכולל x, וניתן לכתוב אליו מידע מאיזן אחר, ואלו שליליים. - x אובייקט מוכרז הכולל x, וניתן לכתוב אליו מידע מאיזן אחר, ואלו שליליים.

iii. פסיקת חומרה (hardware interrupt)

פסיקת תוכנה (trap, software interrupt)

פסיקת חומרה נוצרת כתוצאה מירידה בלתי צפונית, ואילו פסיקת תוכנה נוצרת כתוצאה מירידה בלתי צפונית. Int (הפסיקת תוכנה) היא פסיקת תוכנה, ואילו Int (הפסיקת חומרה) היא פסיקת חומרה. iv. lodsb (לעומת mov al, ds:[si])

mov al, ds:[si]

v. קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

קריאה לשיגרה (לעומת mov al, ds:[si])

המשך שאלה מס' 1 בדף הבא

1. XOR ax, ax

2. MOV ax, 0

3. AND ax, 0000h

4. SUB ax, ax

5. SHL ax, 16

1	0	1	1	0	1	0	0
---	---	---	---	---	---	---	---

6	E	3	4
B	0	D	1
1	F	0	5

האם פעולת חיבור זו גורמת לגלישה?
כן, כי המצב
הוא כזה: חילוקי

(ה) סעיף - int (כחלק מה) sse, se ase הנה int - זהו אסמ

כולל שאלות ופאנלים! המעגל!

287, יקס מסמך מ/אניר אנו יקס סה'ל רזיסטר ווא למת יעבדון ולא עזור
✓ נספח (אלטוולן נרצ) סה'ל ייצג כרטיס > צבין)
add by [ax] iii

מחבר: אברהם יצחק הכהן קאניוויטש

push ah .iv
(1730) 386 16 7.7 (1730) 386 16 7.7

$$dx = 8001 \text{ h}$$
$$\frac{2}{G}$$

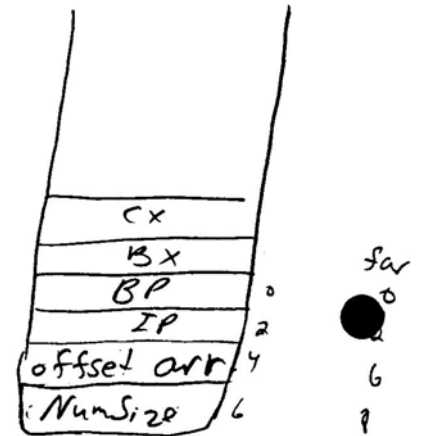
שאלה מס' 2 (25 נקודות)

השיגרה `_findMax` שמוגדרת להלן מקבלת שני פרמטרים במחסנית: כתובת של מערך של מילים, ומספר המילים במערך. השיגרה מחפשת את המילה הגדולה ביותר במערך, כאשר ייצוג המספרים הוא בשיטת המשלים ל-2, ומחזירה את ערכה של מילה זו באוגר `ax`.

```

1  _findMax proc near
2      push bp
3      mov bp, sp
4      push bx
5      push cx
6      mov bx, [bp+4]  bx = offset array
7      mov ax, [bx]    ax = המספר הגדול
8      mov cx, [bp+6]  cx = מספר המילים
9      dec cx          jmp if cx = 0
10     jcxz endFind
11 select: add bx, 2
12         cmp ax, [bx]
13         jge next
14         mov ax, [bx]
15 next:   loop select
16 endFind: pop cx
17         pop bx
18         pop bp
19         ret
20     endp

```



→ ax לא נשמר במחסנית. הסיבה:

א. להלן הגדרת המודל וסגמנט הנתונים של התכנית.

```

.model use16 small
.data
numbers dw 4, -3, -5, 4, 9, -8, 2, 6
len      dw 8
result   dw ?

```

כתוב קטע קוד בשפת אסמבלי שקורא לשיגרה `_findMax` עם המערך `numbers` והאורך `len`, ומציב את הערך המוחזר מן השיגרה בתוך המשתנה `result`. הקפד על כללי תיכנות נכון של קריאה לשיגרה.

מהו תוכנו של המשתנה `result` בגמר קטע הקוד?

```

.CODE
MOV AX, @DATA
MOV DS, AX
PUSH len
PUSH OFFSET numbers
CALL _findMax
ADD SP, 4
MOV result, AX
END

```

result = 9

המשך שאלה מס' 2 בדף הבא

<u>מס' שורה</u>	<u>קוד חדש</u>
7	mov al, [bx]
11	inc bx
12	cmp al, [bx]
14	mov al, [bx]

מס' שורה	קוד חדש
6	mov bx, [bp+6]
8	mov cx, [bp+8]
1	proc far

קוד חדש מס' שורה

JAC next 13

8707 קורס המעבד של הסדר $\text{FindMax}_{\text{public}}$ - $\text{FindMax}_{\text{private}}$ ✓

בס"ד

המשך שאלה מס' 2 בדף הבא

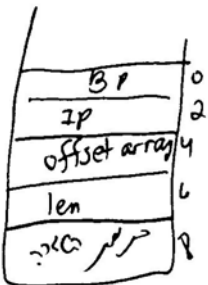
שאלה מס' 2 (המשך)

השיגרה `findBest` (ראה להלן) הינה הכללה של השיגרה `_findMax`. בשיגרה `findBest`, המאפיין של הערך שמחפשים במערך איננו קבוע, אלא מועבר כפרמטר נוסף במחסנית. פרמטר זה הוא מצביע לשיגרה, אשר תפקידה לבחור במילה "יטובה יותר" מבין זוג מילים, לפי מאפיין כלשהו (למשל, המילה הגדולה יותר מבין השתיים, הקטנה יותר, וכו'). "שיגרת הבחירה" מחליפה את הבחירה הקבועה (של המילה הגדולה יותר) שמתבצעת ב- `_findMax` (שורות 12-14).

שיגרת הבחירה מקבלת כפרמטרים זוג מילים: באוגר `ax` ובאוגר `dx`, ומחזירה באוגר `ax` את המילה "יטובה יותר" מבין השתיים. האוגר `dx` נשאר ללא שינוי.

לדוגמא, השיגרה `better1` (ראה להלן) בוחרת במילה הגדולה יותר, כאשר ייצוג המספרים הוא בשיטת המסלים ל-2. השיגרה `findBest` עם הפרמטר `better1` מממשת לפיכך מקרה פרטי הזהה לשיגרה `_findMax`.

להלן הגדרת השיגרה `findBest`. השורות המסומנות בכוכבית הן אלה שנוספו או השתנו ביחס לשיגרה `_findMax`. כמו כן, מוגדרות שתי שגרות בחירה: `better1` ו- `better2`, ודוגמא לקריאה לשגרה `findBest`.



```
findBest proc
    push    bp
    mov     bp, sp
    push    dx      ; *
    push    bx
    push    cx
    mov     bx, [bp+4]
    mov     ax, [bx]
    mov     cx, [bp+6]
    dec     cx
    jcxz    endFind
find:      add     bx, 2
            mov     dx, [bx] ; *
            call    [bp+8]   ; *
next:      loop    find
endFind:   pop     cx
            pop     bx
            pop     dx      ; *
            pop     bp
            ret
endp
```

```
better1 proc
    cmp     ax, dx
    jge     end1
    mov     ax, dx
end1:      ret
endp
```

```
better2 proc
    cmp     ax, dx
    jnb     end2
    mov     ax, dx
end2:      ret
endp
```

```
main:      mov     ax, @data
            mov     ds, ax
            push    offset better1
            push    len
            push    offset numbers
            call    findBest
            add     sp, 6
            mov     result, ax
            end     main
```

1. מהו החיפוש שמממשת השגרה `findBest` עם הפרמטר `better2` ?
דייק בתיאור המאפיין של שיגרת הבחירה.

השגרה `findBest` א הסמל `better2` כואת כמילה הגדולה ביותר
לעין מילך ולורכו len כשר יצא הסמל הן כמספר לא מילניק (סמל, לא)

2. נתון סגמנט הנתונים מסעיף א'. מהו הערך שתחזיר השיגרה `findBest` בקריאה שלהלן?

```
push    offset better2
push    len
push    offset numbers
call    findBest
add     sp, 6
```

המשך שאלה מס' 2 בדף הבא

הסך ומלחזר הא 3- כ סמל 1101
נהל הגדול לקר כמל לא מסמל.

שאלה מס' 2 (המשך)

ח. כתוב שיגרה בשם better3, שמקבלת כפרמטרים זוג מילים: באוגר ax ובאוגר dx, ומחזירה באוגר ax את המילה שערכה המוחלט קטן יותר, כאשר ייצוג המספרים בשיטת המשלים ל-2. הקפד על כללי תיכנות נכונים של שיגרה. רמז: להיפוך הסימן ניתן להשתמש בפקודה neg.

better3 proc

push bx,

push cx

mov bx,ax ; do not change ax

mov cx,dx ; do not change dx

cmp bx,0

jns GoOn1

neg bx

GoOn1:

cmp cx,0

jns GoOn2

neg cx

GoOn2:

cmp bx,cx

jle exitbetter3 ; $bx \leq cx \Rightarrow |ax| \leq |dx|$

mov ax,dx

exitbetter3:

pop cx

pop bx

RET

endp

ט. נתון סגמנט הנתונים מסעיף א'. מהו הערך שתחזיר השיגרה findBest בקריאה שלהלן?

```
push offset better3
push len
push offset numbers
call findBest
add sp,6
```

הערך findBest מחזיר את 2 כי הוא המספר המזערי (המחזיר)
הוא הקטן ביותר

use // →
מחזיר
ערך
קטן
מזערי

שאלה מס' 3 (25 נקודות)

א. כידוע, הפקודה `mov` אינה מאפשרת העתקה בין שני אופרנדים בזיכרון. הגדר מאקרו בשם `move` אשר מרחיב את הפקודה `mov`, ומאפשר העתקה גם בין שני אופרנדים בזיכרון.

לדוגמא, נניח כי `var1` ו-`var2` הינן מילים בסגמנט הנתונים. אזי הקריאה `move var2,var1` מעתיקה את `var1` אל `var2`.

ב. השתמש במחסנית.

```
move macro x,y
    push y
    pop x
endm
```

ב. להלן הגדרת המאקרו `comp`.

```
1 comp macro x,y
2     ifidni <&y>,<ax>
3         cmp x,y
4         exitm
5     endif
6     push ax
7     mov ax,x
8     cmp ax,y
9     pop ax
10    endm
```

לפניך שלוש קריאות למאקרו `comp`. רשום את קוד האסמבלי שנפרש על ידי כל קריאה. הנח כי `var1` ו-`var2` הינן מילים בסגמנט הנתונים.

<p>iii. <code>comp var1,ax</code> <code>cmp var1,ax</code></p>	<p>ii. <code>comp es,bx</code> <code>push ax</code> <code>mov ax,es</code> <code>cmp ax,bx</code> <code>pop ax</code></p>	<p>i. <code>comp var1,var2</code> <code>push ax</code> <code>mov ax,var1</code> <code>cmp ax,var2</code> <code>pop ax</code></p>
---	---	--

ג. i. הקריאה `comp bx,es` גורמת לשגיאת אסמבלי. הסבר מהי השגיאה.

לעומת זאת, הקריאה `comp es,bx` (ראה סעיף ב' ii) אינה גורמת לשגיאה. הסבר מדוע קיים הבדל בין שתי הקריאות.

הקריאה `comp es,bx` אינה (שלא יקרה כי ששגה) הקריאה:

```
push ax
mov ax,bx
cmp ax,es
pop ax
```

שגיאת קואסיסטנציה. תראה כי הקריאה `cmp ax,es` כיוון שהיא

לא יתאפשר לה. יש לה שטח קודן לא יתאפשר לה.

כך העולם ד - `comp es,bx` שטח קודן לא יתאפשר לה.

ii. הסבר מהי השגיאה שגורמת הקריאה `comp bh,ch`.

```
push ax
mov ax,bh
cmp ax,ch
```

`pop ax`

המשך שאלה מס' 3 בדף הבא

כדי לה זכור את כל הקריאות, יש לה שטח קודן לא יתאפשר לה. כיוון שהיא לא יתאפשר לה. יש לה שטח קודן לא יתאפשר לה.

שאלה מס' 3 (המשך)

ד. הסבר בקצרה מה עושה המאקרו `comp`.
מהן מגבלות השימוש במאקרו? (רמז: ראה סעיף ג').

המקורו comp מילאם כצ"ע כסקורה קמס. הסקורה קמס אים יכיל מלכא השולח
בין שני מלכא צבין והמקורו קמס מלכא מלכא השולח גלגל.

ה. המאקרו `move`, שהגדרתו נתונה להלן, מעתיק את האופרנד `y` אל האופרנד `x` אם ורק אם מתקיים התנאי $(x \text{ cond } y)$. האופרנד `cond` הוא סיומת של פקודת הסתעפות מותנית כלשהי (כגון `g, l, ge`, וכד').

לדוגמא, הקריאה `movc ax,bx,ae` מעתיקה את האוגר `bx` אל האוגר `ax` אם תוכנו של `ax` גדול או שווה לתוכנו של `bx`, כאשר הייצוג המספרי הוא ללא סימן (בגלל השימוש בתנאי `ae`).

השלם את הגדרת המאקרו `movc` במקומות המסומנים בקו.

```

movc      macro    x,y,cond
            local   movit,movend
            cmp     x,y
            j&cond  movit
            jmp     movend
movit:    mov      x,y
movend:
            endm

```

1. המאקרו \max , שהגדרתו נתונה להלן, מציב באופרנד x את הגדול מבין שני האופרנדים $x-1$ ו- y , כאשר הייצוג המספרי הוא בשיטת המשלים ל-2. האופרנד y נשאר ללא שינוי. השלם את הגדרת המאקרו \max במקום המסומן בקו.

```
max macro x,y
movc x,y,z
endm
```

1. הגדר את המאקרו $\max3$ (ראה שלד להלן), שמציב באופרנד x את הגדול מבין שלשת האופרנדים $x, y, z-1$.

עליך להשתמש בקריאות למאקרו `.max` אין להשתמש ישירות בפקודות `mov` או `cmp`.

```
max3 macro x,y,z
    max    x,y
    max    x,z
endm
```

ח. על ידי הכנסת שני שינויים פשוטים בהגדרת המאקרו `move` נאפשר למאקרו לקבל אופרנדים מסוגים נוספים. הסבר בקצרה למה הכוונה. נא לא לכתוב קוד.

המקור mov נקרא המקור cmp ע"י המבין את המקור את המקור
האלמנטים שהיו קודם ח"ל.ל.ל. המספר, ה"י. המספר את המקור (בין המספר את המקור)
נ"ל להיות י. המקור מספר בין שני המקור מספר ונ"ל בין
את המקור אל המקור ונ"ל המקור.

שאלה מס' 4 (25 נקודות)

כידוע, חוצץ המקלדת יכול להכיל 15 תווים בלבד. כל עוד התכנית שמתבצעת אינה משתמשת בקלט מחוצץ המקלדת, לא תתאפשר הקלדה של יותר מאשר 15 תווים. זוהי מגבלה, שכן לעיתים נוח להקליד קלט לתכנית דוקא כאשר זו עסוקה בדברים אחרים. התכנית שלהלן מדגימה כיצד אפשר להתגבר על מגבלת המקום בחוצץ המקלדת.

נקצה שטח, גדול כרצוננו, באחד הסגמנטים של תכנית המשתמש, ונכתוב תוספת לשיגרת הטפול בפסיקת המקלדת (int 9), אשר תעביר אל השטח שהוקצה כל תו שמגיע מן המקלדת. רק כאשר שטח גדול זה מתמלא לגמרי, לא תתאפשר הקלדה נוספת. למנגנון זה יתרון נוסף, והוא שתכנית המשתמש יכולה לגשת ישירות אל התווים שהוקלדו, ללא צורך בשירותי מערכת ההפעלה.

להדגמת המנגנון, התכנית הראשית שלהלן ממתינה עד שהשטח שהוקצה מתמלא לגמרי, ואז מדפיסה בבת אחת את כל הקלט על המסך ומרוקנת את השטח. שיגרת המקלדת והתכנית הראשית מסמנות זו לזו כאשר השטח מתמלא או מתרוקן לגמרי.

<pre> 1 .model use16 small 2 .stack 100h 3 .data 24 db 10,13 45 keybrdBuf db 1000 dup (?) 6 trailer db 10,13,'\$' 7 repeat db 3 8 .code 9 oldInt9 dd ? 10 bufStart dd far ptr keybrdBuf 11 bufLen dw trailer-keybrdBuf 12 bufIndex dw 0 13 isBufFull db 0 14 newInt9: pushf 15 call oldInt9 16 cmp isBufFull,1 17 je ret9 18 push es 19 push ax 20 push bx 21 push di 22 mov ah,1 23 int 16h 24 jz exit9 25 mov ah,1 26 int 21h 27 les bx,bufStart 28 mov di,bufIndex 29 mov es:[bx+di],al 30 inc di 31 mov bufIndex,di 32 cmp di,bufLen 33 jl exit9 34 mov bufIndex,0 35 mov isBufFull,1 36 exit9: pop di 37 pop bx 38 pop ax 39 pop es 40 ret9: iret </pre>	<pre> 41 main: mov di,@data 42 mov ds,di 43 mov ax,3509h 44 int 21h 45 mov word ptr oldInt9,bx 46 mov word ptr oldInt9+2,es 47 push ds 48 mov dx,cs 49 mov ds,dx 50 mov dx,offset newInt9 51 mov ax,2509h 52 int 21h 53 pop ds 54 idle: cmp isBufFull,0 55 je idle 56 mov dx,offset keybrdBuf-2 57 mov ah,9 58 int 21h 59 dec repeat 60 jz exit 61 mov isBufFull,0 62 jmp idle 63 exit: lds dx,oldInt9 64 mov ax,2509h 65 int 21h 66 mov ah,4ch 67 int 21h 68 end main </pre>
--	--

Handwritten notes:

- Line 28: $es:bx = ds:04h$
- Line 29: $di = 0$
- Line 30: $es:[bx+di], al \rightarrow$
- Line 31: di
- Line 32: $di < 1000$
- Line 33: $di < 1000$
- Line 34: $di = 0$
- Line 35: $isBufFull = 1$
- Line 36: $exit9$
- Line 37: $pop di$
- Line 38: $pop bx$
- Line 39: $pop ax$
- Line 40: $pop es$
- Line 41: $main$
- Line 42: $mov di, @data$
- Line 43: $mov ds, di$
- Line 44: $mov ax, 3509h$
- Line 45: $int 21h$
- Line 46: $mov word ptr oldInt9, bx$
- Line 47: $mov word ptr oldInt9+2, es$
- Line 48: $push ds$
- Line 49: $mov dx, cs$
- Line 50: $mov ds, dx$
- Line 51: $mov dx, offset newInt9$
- Line 52: $mov ax, 2509h$
- Line 53: $int 21h$
- Line 54: $pop ds$
- Line 55: $cmp isBufFull, 0$
- Line 56: $je idle$
- Line 57: $mov dx, offset keybrdBuf-2$
- Line 58: $mov ah, 9$
- Line 59: $int 21h$
- Line 60: $dec repeat$
- Line 61: $jz exit$
- Line 62: $mov isBufFull, 0$
- Line 63: $jmp idle$
- Line 64: $lds dx, oldInt9$
- Line 65: $mov ax, 2509h$
- Line 66: $int 21h$
- Line 67: $mov ah, 4ch$
- Line 68: $int 21h$
- Line 69: $end main$

שאלה מס' 4 (המשך)

עליך להסביר את אופן פעולתה של התכנית שבדף הקודם, לפי השאלות שלהלן.
רמז: השאלות נועדו להנחות אותך בהבנת התמונה הכללית. מומלץ לעבור על כל השאלות מראש.

א. מהו תפקידן של שורות 43-46? מהו תפקידן של שורות 47-53?

43-46 מעביר את כתובת של `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 47-53 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 54-55 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 56-57 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 58-59 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 60-61 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 62-63 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 64-65 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 66-67 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 68-69 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 70-71 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 72-73 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 74-75 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 76-77 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 78-79 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 80-81 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 82-83 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 84-85 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 86-87 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 88-89 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 90-91 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 92-93 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 94-95 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 96-97 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 98-99 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`
 100 מעביר את כתובת `int9` המקורי למערכת קטלוגי הספר `oldInt9`

ב. שיגרת השירות החדשה של פסיקת המקלדת (newInt9) מפעילה קודם כל את שיגרת השירות הישנה (המוצבעת על ידי oldInt9), ורק לאחר מכן מבצעת משימות נוספות. הסבר מדוע. (רמז: מה עושה שיגרת הפסיקה הישנה?)

פס'ק'ה ט"ז ד' כ"א ה"ת"ר ש"כ"ה לכתוב לכתוב המעלה, ולא מעלה לכתוב
 וכל שיהיה א"ת קודם לכתוב א"ת המעלה. לכתוב א"ת ש"ת ש"ת
 ית"ה א"ת לכתוב המעלה הי"ש לכתוב כתוב לכתוב המעלה.
 ח"כ כ"א נ"ת א"ת לכתוב קודם לכתוב המעלה א"ת ש"ת ש"ת
 המעלה וכתוב א"ת (ש"ת) לכתוב המעלה מה מבטיח שתהיה חזרה אל שיגרת
 כיצד מתבצע המעבר אל שיגרת הפסיקה הישנה? מה מבטיח שתהיה חזרה אל שיגרת
 הפסיקה החדשה? (רמז: מהו האפקט המשולב של שורות 14-15?)

[illegible]

שנים 24-22 דיוקן לראש ים אשר חתום המעצף. ים אשר חתום (2F=0)
 גלי סדקיה המסומנים, ודמייה לראש - חתום המעצף. המסומן.
 שנים 25-26 סדקיה דמייה אוש אשר חתום המעצף. - לאשי הן מציגים
 האשר ודיוקן לראש אשר חתום המעצף. חתום המעצף.
 ה. מהו תפקידן של שורות 27-31?

על 31-27 מרצם 2023
 → bufstart ישרת ויחזיק את תחילת הסט המוקד (ds:04h) →
 ישרת הבקר 1000 עכיל אלף הסט המוקד

שלישית אלו כל המידע שהיה בתיק הלאומי, תוך הולקו, סי, סטון, המיון, והוא ש 1000 =
המשך שאלה מס' 4 בדף הבא

שלישית, הסט המוקדש לא, אלו מקבוצת אינה למכירת המסחר ב הסט המוקדש הלא, הלא, ה- יום האחרון ב פדק.

אם מניסיון פדק אינה נמצאה מקום והוא אספה המיון, סי, סטון, ה- יום האחרון ב פדק.

שלישית, הסט המוקדש (מקום) ה- יום האחרון ב פדק, ה- יום האחרון ב פדק, ה- יום האחרון ב פדק.

שאלה מס' 4 (המשך)

1. אילו ערכים מקבל המשתנה isBuffFull במשך מהלך התכנית? איזה חלק של התכנית (שיגרת המקלדת, או התכנית הראשית) מציב כל אחד מערכים אלה במשתנה?

isBuffFull מקבל 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים. זה נקבע על ידי המערכת הראשית.

2. שיגרת המקלדת "מסמנת" לתכנית הראשית כאשר השטח החדש שהוקצה מתמלא לגמרי. בדומה, התכנית הראשית "מסמנת" לשיגרת המקלדת כאשר שטח זה מתרוקן לגמרי. הסבר כיצד הדבר נעשה. (רמז: ראה סעיף ג.)

שיגרת המקלדת משתמשת במשתנה isBuffFull כדי לדעת מתי היא צריכה להפסיק לקבל נתונים.

המשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.

3. מהו תפקידן של שורות 16-17? מהו תפקידן של שורות 54-55? מהו תפקידן של שורות 55-54?

שורות 16-17: קובצות את הנתונים שהמערכת קיבלה, ומוציאות אותם למסך. שורות 54-55: קובצות את הנתונים שהמערכת מציגה, ומוציאות אותם למסך.

4. מהי הסיבה לכך שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים?

הסיבה לכך היא שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.

5. מה יקרה אם הקלט מן המקלדת מכיל את התו "q"? מה יקרה אם הקלט מן המקלדת מכיל את התו "q"?

אם הקלט מן המקלדת מכיל את התו "q", המערכת תציג אותו למסך. אם הקלט מן המקלדת מכיל את התו "q", המערכת תציג אותו למסך.

6. מהי הסיבה לכך שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים?

הסיבה לכך היא שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.

7. מהי הסיבה לכך שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים?

הסיבה לכך היא שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.

8. מהי הסיבה לכך שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים?

הסיבה לכך היא שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.

9. מהי הסיבה לכך שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים?

הסיבה לכך היא שהמשתנה isBuffFull מציב 0 כאשר המערכת לא קיבלה שום נתונים, ו-1 כאשר היא קיבלה נתונים.