את"ם – תרגיל בית מס' 4 - סמסטר חורף תשע"ו

תאריך פרסום: 31/12/2015 תאריך הגשה: 21/1/2016 (23:55 בלילה)

- **ההגשה בזוגות בלבד לתא ההגשה של הקורס ובאמצעות הגשה אלקטרונית.**
- **שאלות על התרגיל יש להפנות למאיה לוי mayale.v87@gmail.com.**
- **הגשות באיחור יש לתאם עם מאיה לפני מועד ההגשה הכללי.**
- **אין להגיש לתא הקורס לאחר מועד ההגשה.**

נושא התרגיל: פסיקות

בתרגיל זה שני חלקים:

- חלק א' מכיל שני חלקים. בכל חלק קטע קוד ושאלה לגביו. עליכם לענות על כל סעיפי השאלות בכתב ולהגיש לתא הקורס (יש להדפיס את טופס התרגיל ולענות על גביו).
- חלק ב' דורש כתיבת קוד בשפת האסמבלי של PDP-11, כפי שנלמד בהרצאות ובתרגולים. את הקוד יש לכתוב בקובץ ex4.s11. כדאי לקרוא באתר הקורס ב- FAQ על רמת התיעוד הנדרשת. יש להגיש את הקובץ ex4.s11 אלקטרונית דרך האתר (יש להגיש אלקטרונית רק את הקובץ ex4.s11. אין להגישו מכווץ בתור קובץ Zip כפי שמצוין באתר). **אין צורך להדפיס את הקוד. יש להגיש לתא הקורס רק את החלק היבש ואת התיעוד החיצוני של הקוד.**

חלק יבש (מכונה מדומה)

המעבד PNP-11 זהה בכל למעבד PDP-11, פרט לכך שאוצר הפקודות שלו מכיל בנוסף את הפקודה jsrt, בעלת המבנה הבא בשפת מכונה:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	7			6			*i*			*d*			*d*		

כאשר *i* הינו שדה בן שלוש סיביות המציין מספר אוגר (בין 0 ל 5) ו *dd* הינו שדה בן שש סיביות בו מקודד אופרנד ‹operand› כמקובל ב-PDP-11, כלומר באמצעות שיטת מיעון ומספר אוגר.
הפקודה jsrt מבצעת קריאה לשגרה בכתובת האפקטיבית של האופרנד השני ומגבילה את זמן הריצה שלה ל x שניות לכל היותר, כאשר x הוא הערך של הרגיסטר Ri בזמן ביצוע הפקודה.
במידה ופרק הזמן המוקצב הסתיים לפני שהשגרה הסתיימה לרוץ, ישוחזרו ערכי הרגיסטרים לערכיהם לפני הקריאה לשגרה (כולל sp), דגל ה overflow ידלק והתוכנית תמשיך לרוץ מהפקודה הבאה אחרי jsrt.
אם השגרה תסתיים לפני שפרק הזמן המוקצב יעבור, ההתנהגות תהיה זהה לפקודה jsr במעבד PDP-11.

הקריאה לפקודה jsrt בשפת אסמבלי תראה כך:

jsrt Ri, ‹operand›

כאשר ביצוע הפקודה מתואר ע"י הפסאודו קוד הבא:

$$(1)\ push(pc)$$
$$(2)\ pc \leftarrow EA2$$

<u>הערות:</u>

א. לפקודה jsrt ו jsr יש מבנה דומה בשפת מכונה.
ב. קידוד הפקודה jsrt אינו חוקי ב- PDP-11.

לשאלה זו שני חלקים, א' ו-ב', המופיעים בעמודים הבאים.

חלק א'

להלן תוכנית שנכתבה עבור המעבד PNP-11. קיראו את התוכנית וענו על הסעיפים שבעמודים הבאים.

```
1.  tks        =       177560
2.  tkb        =       177562
3.  tps        =       177564
4.  tpb        =       177566

5.  .          =       torg + 60
6.  .word      input,  0
7.  .word      print,  0

8.  .          =       torg + 1000
9.  main:      mov     pc,         sp
10.            tst     -(sp)
11.            mov     #100,       @#tps
12.            mov     #101,       @#tks
13.            mov     #60.,       r3
14.            jsrt    r3,         unlock
15.            bvc     cont
16.            mov     #stralarm,  strptr
17. cont:      movb    @strptr,    @#tpb
18. waitp:     wait
19.            tstb    @strptr
20.            bne     waitp
21.            halt

22. unlock:    mov     #20.,       r3
23.            jsrt    r3,         attempt
24.            bvs     unlock
25.            rts     pc

26. attempt:   clr     r1
27. loop:      movb    times(r1),  r5
28.            jsrt    r5,         check_cur
29.            bvs     v_set
30.            cmpb    combi(r1),  #'w
31.            bne     advance
32.            br      attempt
33. v_set:     cmpb    combi(r1),  #'w
34.            beq     advance
35.            br      attempt
36. advance:   inc     r1
37.            tstb    combi(r1)
38.            bne     loop
39.            rts     pc

40. check_cur: wait
41.            tstb    lastasc
42.            beq     check_cur
43.            cmpb    combi(r1),  #'w
44.            beq     end_chk
45.            cmpb    lastasc,    combi(r1)
46.            beq     end_chk
47.            jsr     pc,         sleep
48.            clrb    lastasc
49.            br      check_cur
50. end_chk:   clrb    lastasc
51.            rts     pc

52. input:     movb    @#tkb,      lastasc
53.            inc     @#tks
54.            rti

55. print:     inc     strptr
56.            tstb    @strptr
57.            beq     endprint
58.            movb    @strptr,    @#tpb
59. endprint:  rti

60. sleep:     mov     r4,         -(sp)
61.            ____________________ __________
62.            jsrt    r4,         __________
63.            ____________________ __________
64.            ____________________ __________

65. bwait:     br      bwait
66.            rts     pc

67. strptr:    .word   strnoal
68. combi:     .ascii  <4w81ww9>
69.            .byte   0
70. times:     .byte   5, 1, 5, 5, 1, 1, 5
71. lastasc:   .byte   0
72. stralarm:  .ascii  <ALARM!!!>
73.            .byte   0
74. strnoal:   .ascii  <Ok>
75.            .byte   0
76.            .even
```

1. רישמו (בבסיס אוקטאלי) את התרגום לשפת מכונה של הפקודה jsrt r3, attempt שבשורה 23. אם אורכה גדול ממילה אחת רישמו את כל המילים בזו אחר זו.

---

2. הפקודה שבשורה 28 התבצעה. סמנו עבור איזו אופציה לערך של האוגר r1 ולהקלדת המשתמש הפקודה שבשורה 47 תתבצע:

   א. r1 = 0, המשתמש הקליד את התו '7'.
   ב. r1 = 0, המשתמש הקליד את התו '4'.
   ג. r1 = 1, המשתמש לא הקליד דבר במשך שניה.
   ד. r1 = 1, המשתמש הקליד את התו 'w'.
   ה. r1 = 2, המשתמש הקליד את התו '8'.
   ו. r1 = 2, המשתמש לא הקליד דבר במשך 5 שניות.
   ז. אף אחת מהתשובות א'-ו' אינה נכונה.

לצורך הסעיפים הבאים הניחו כי זמן ביצוע כל פקודה הוא 0 וזמן קליטת תו הוא ¾ שניה.

3. משתמש הקליד את רצף התווים '46'. לאחר קליטת התו השני ואחרי החזרה מהשגרה check_cur, איזו מבין ההסתעפויות בשורות 31-35 תגרום לקפיצה?

   א. ההסתעפות שבשורה 31.
   ב. ההסתעפות שבשורה 32.
   ג. ההסתעפות שבשורה 34.
   ד. ההסתעפות שבשורה 35.
   ה. אף אחת מהתשובות א'-ד' אינה נכונה.

4. סמנו עבור אלו מבין האפשרויות הבאות לשינוי שורה מס' 70 התוכנית לעולם לא תדפיס את המחרוזת שבתווית strnoal (לכל רצף מקשים שיוכנס בכל קצב). <u>יכולות להיות מספר תשובות נכונות.</u>

   א. times: .byte 100, 1, 5, 5, 1, 1, 5
   ב. times: .byte 5, 21., 5, 5, 1, 1, 5
   ג. times: .byte 3, 3, 3, 3, 3, 3, 3
   ד. times: .byte 2, 7, 4, 4, 7, 7, 5
   ה. times: .byte 60., 1, 60., 60., 1, 1, 60.

5. מטרת השגרה **sleep** היא להשהות את ריצת התוכנית למשך שניה אחת. השלימו את השורות הריקות כך שהשגרה תבצע פעולה זו.

6. מהו פרק הזמן המינימלי (בשניות) שיעבור לפני הדפסת המחרוזת שבתווית strnoal? כתבו מספר בבסיס דצימלי.

חלק ב'

מעוניינים לאפשר להריץ קוד בשפת מכונה, שנוצר בתרגום על-ידי אסמבלר של PNP-11, על המעבד PDP-11. לצורך זה מוצע להוסיף את מנגנון הסימולציה הבא לקוד המכונה של תוכניות המשתמשות בפקודה jsrt:

```
jsrmsk    =        004700
jsrtmsk   =        076000
lcs       =        177546
ticksec   =        50.

.         =        torg + 10
.word     jsrtcmd, 340
.word     thandle, 340
.         =        torg + 30
.word     emthndl, 0
.         =        torg + 100
.word     clock,   340

.         =        torg + 2000
jsrtcmd:  mov      #100,     @#lcs
          sub      #2,       (sp)
          mov      @0(sp),   lastreg
          bic      #177077,  lastreg
          bic      #000700,  stsec
          bis      lastreg,  stsec
stsec:    mov      r0,       lastsec
          mov      (sp),     cmdaddr
          bic      #177700,  @cmdaddr
setcmd:   bis      #jsrmsk,  @cmdaddr
settbit:  bis      #20,      2(sp)
          rtt

clock:    mov      r0,       -(sp)
          mov      #jsrtsp,  r0
cl_loop:  cmp      jsrtsp,   r0
          bge      clk_end
          sub      #22,      r0
          dec      (r0)
          bne      cl_loop
timeout:  mov      r0,       jsrtsp
          add      #22,      jsrtsp
          mov      4(r0),    sp
          clr      (sp)
          bis      #2,       (sp)
          mov      2(r0),    -(sp)
          mov      6(r0),    r5
          mov      10(r0),   r4
          mov      12(r0),   r3
          mov      14(r0),   r2
          mov      16(r0),   r1
          mov      20(r0),   r0
          rti
clk_end:  mov      (sp)+,    r0
          rti

goodret:  emt      0

emthndl:  add      #2,       jsrtsp
          mov      @jsrtsp,  (sp)
          clr      2(sp)
          add      #20,      jsrtsp
          rtt

thandle:  bic      #177700,  @cmdaddr
          bis      lastreg,  @cmdaddr
          bis      #jsrtmsk, @cmdaddr
save:     mov      r5,       -(sp)
          mov      jsrtsp,   r5
          mov      r0,       -(r5)
          mov      r1,       -(r5)
          mov      r2,       -(r5)
          mov      r3,       -(r5)
          mov      r4,       -(r5)
          mov      (sp),     -(r5)
          mov      sp,       -(r5)
          add      #6,       (r5)
          mov      6(sp),    -(r5)
          mov      lastsec,  r1
          mul      #ticksec, r1
          mov      r1,       -(r5)
          mov      16(r5),   r1
          mov      r5,       jsrtsp
          mov      (sp)+,    r5
          mov      #goodret, 4(sp)
thend:    bic      #20,      2(sp)
          rtt

.         =        torg + 3000
jsrtcalls: .blkw   100
jsrtsp:   .word    jsrtsp
lastreg:  .blkw 1
cmdaddr:  .blkw 1
lastsec:  .blkw 1
```

7. מה מבצעת הפקודה בשורה 119?

א. שומרת בכתובת lastsec את מספר האוגר מהקריאה של jsrt.
ב. שומרת בכתובת lastsec את הערך של האוגר r0.
ג. שומרת בכתובת lastsec את פרק הזמן שהשגרה הנקראת מורשית לרוץ בשניות.
ד. שומרת בכתובת lastsec את פרק הזמן שהשגרה הנקראת מורשית לרוץ במס' פסיקות שעון.
ה. שומרת בכתובת lastsec את מס' מחזורי השעון שחלפו מתחילת התוכנית.
ו. שומרת בכתובת lastsec את מס' השניות שחלפו מתחילת התוכנית.
ז. אף אחת מהתשובות א'-ו' אינה נכונה.

8. מריצים את הקוד מתוכנית א' עם מנגנון הסימולציה מחלק ב'. מה תהיה הפקודה שתבוצע מיד אחרי ביצוע הפקודה שבשורה 124 ב**פעם הראשונה**?

א. bvc cont
ב. jsrt r3, unlock
ג. bic #177700, @cmdaddr
ד. jsr pc, unlock
ה. תבוצע פקודה עם קידוד לא חוקי (שאינה jsrt).
ו. אף אחת מהתשובות א'-ה' אינה נכונה.

9. מריצים את התוכנית מחלק א'. תארו את תוכן המחסנית מיד לפני ביצוע הפקודה rtt שבשורה 175 בפעם הראשונה. ניתן לכתוב ביטוי מהצורה "הכתובת של שורה 15" וניתן להשתמש בתוויות. הניחו שבתחילת התוכנית תוכן של אוגר r_i הוא i (עבור $i<6$). במידה שקיים במחסנית ערך לא ידוע, כיתבו "לא ידוע" במקום המתאים.

כתובת	תוכן (מספר אוקטאלי)
770	
772	
774	
776	
1000	**010706**

10. בתווית jsrtcalls מוגדרת מחסנית השומרת נתונים בכל ביצוע של הפקודה jsrt. המילה בתווית jsrtsp מציינת את הכתובת של ראש המחסנית הזו. תארו את תוכן המילה שבתווית jsrtsp ואת תוכן מחסנית זו מיד לפני ביצוע הפקודה rtt שבשורה 175 בפעם הראשונה.

כתובת	תוכן (מספר אוקטאלי)
3156	
3160	
3162	
3164	
3166	
3170	
3172	
3174	
3176	
3200 (jsrtsp)	

11. מתי תתבצע הפקודה שבשורה 147?

א. בחזרה משגרה שנקראה באמצעות jsrt והשלימה את עבודתה (חזרה באמצעות rts).

ב. בחזרה משגרה שנקראה באמצעות jsrt ולא השלימה את עבודתה (הסתיים הזמן שהוקצה לה לרוץ).

ג. בחזרה מהשגרה thandle.

ד. הפקודה לא תתבצע.

ה. אף אחת מהתשובות א'-ד' אינה נכונה.

12. מריצים את התוכנית מחלק א'. בהנחה שלא מוקלד אף תו תארו את תוכן המחסנית מיד לפני ביצוע הפקודה בשורה 144 בפעם הראשונה.

כתובת	תוכן (מספר אוקטאלי)
764	
766	
770	
772	
774	
776	
1000	**010706**

חלק רטוב (Sliding Puzzle)

תזכורת מתרגיל בית 2

ה Sliding Puzzle היא חידה המורכבת מלוח בגודל $m \times n$ המכיל את המספרים 1 עד $m \cdot n - 1$ ומשבצת אחת ריקה. פתרון החידה מתקבל ע״י סידור המספרים לפי האינדקסים של המערך (כך שהמספר i יהיה באינדקס i) ע״י הזזה בכל שלב של מספר הנמצא בסמוך למשבצת הריקה לתוכה.

תיאור המשימה

בתרגיל זה תממשו משחק sliding puzzle מלא. שימו לב כי מותר, ואף רצוי, להיעזר בפתרונות לתרגילי הבית הקודמים.

במשחק אותו תממשו ישנם שני שחקנים, שחקן א׳ אשר מגדיר את ממדי וערכי לוח המשחק, ושחקן ב׳ אשר משחק ומטרתו היא להביא את הלוח ההתחלתי למצב הסופי כפי שהוגדר בש״ב 2.

תחילת המשחק

המשחק כולל את הצעדים הבאים:

1. שחקן א׳ יישאל מהי כמות השורות בלוח במשחק:

```
Please enter number of rows:
```

בשורה הבאה תקלט כמות השורות. ניתן להניח שיתקבל מספר בין 1 ל-4.

2. שחקן א׳ יישאל מהי כמות העמודות בלוח במשחק:

```
Please enter number of columns:
```

בשורה הבאה תקלט כמות העמודות. ניתן להניח שיתקבל מספר בין 1 ל-4.

3. שחקן א׳ יישאל כמה שניות מוקצות למשחק:

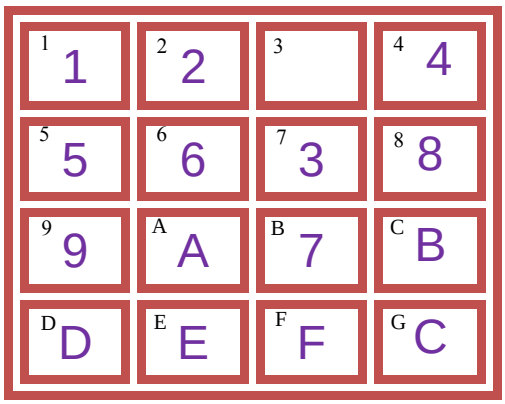
```
Please enter game duration in seconds:
```

בשורה הבאה תקלט כמות השניות שיותרו למשחק עבור שחקן ב׳. יש לפרש את המספר כמספר אוקטלי וניתן להניח שהוא בין 1 ל-1000.

4. שחקן א׳ יתבקש להכניס את לוח המשחק:

```
Please enter the sliding puzzle:
```

בשלב זה שחקן א׳ יקיש את לוח המשחק על פי שיטת קידוד הלוח שהוצגה בש״ב 2. המספרים שיוכנסו יהיו ספרות בבסיס 17, הספרות ייוצגו ע״י התווים 1-9, A-G.
לדוגמא, הלוח:



יוכנס ע"י שחקן א' באופן הבא:

```
127456B89ACGDEF
```

ניתן להניח שמתקבל לוח חוקי שתואם את הממדים שהוכנסו בסעיפים הקודמים.

5. כעת השליטה עוברת לשחקן ב'. כמות השניות שהוכנסה בסעיף 3 תודפס ושחקן ב' יתבקש להקיש Enter כאשר הוא מוכן להתחיל במשחק:

```
We are all set. Total time: 450 sec
Press Enter key when you are ready…
```

כאשר שחקן ב' יקיש Enter הזמן יתחיל להימדד ונעבור לסעיף 6. ניתן להניח שהתו שיוקש הוא Enter. שימו לב, עד שלב זה אין להפעיל את פסיקות השעון.

מהלך המשחק

6. עליכם להציג את הלוח הנוכחי ואחריו הודעה המנחה את שחקן ב' להכניס מהלך. ערכי הלוח יוצגו בבסיס 16, בין כל שני מקומות באותה שורה יודפס התו רווח ואחרי כל שורה יודפס התו Enter. במשבצת הפנויה תודפס כוכבית. לדוגמא, הלוח מסעיף 4 יוצג כך:

```
Current Board:
1 2 * 4
5 6 3 8
9 A 7 B
D E F C
Please enter your move:
```

7. בשלב זה על שחקן ב' להכניס את המהלך. מהלך מיוצג על ידי אחת מהאותיות 'U' , 'D' , 'R' או 'L'. כל אות מייצגת מהלך מתאים כפי שהוגדר בתרגילי הבית הקודמים. ניתן להניח כי המהלך שיתקבל יהיה אחת מארבע האותיות הנ"ל.

אם המהלך לא חוקי (על פי הגדרת מהלך חוקי מש"ב קודמים), עליכם להדפיס את ההודעה הבאה:

```
Illegal move. Try again
```

ולחזור על סעיף זה.

אם המהלך חוקי, עליכם לבצע אותו ולחזור לסעיף 6.

סיום המשחק

שחקן ב' יוכל לשחק (באמצעות הפקודות שהופיעו למעלה) עד שאחד משני התנאים הבאים יתקיים:

1. הזמן המוקצב למשחק הסתיים.
2. שחקן ב' פתר את המשחק.

פקיעת זמן המשחק

8. אם תם הזמן (שהוקצב על-ידי שחקן א') נבחין בין המקרים הבאים:

- אם תם הזמן במהלך ההמתנה לקלט בסעיף 7, תופיע מיד ההודעה הבאה:

```
Time's up. Game Over!
```

ולאחר מכן יסתיים המשחק.

- אם תם הזמן בשלב עיבוד צעד, והצעד הוביל לניצחון, נעבור לסעיף 9.
- אם תם הזמן בכל שלב אחר במשחק, ההודעה:

```
Time's up. Game Over!
```

תופיע מיד לפני שהביצוע הבא של סעיף 7 היה אמור להתרחש, ולאחר מכן יסתיים המשחק.

פתרון המשחק

9. אם שחקן ב׳ פתר את המשחק, יודפס הלוח הפתור והודעה על ניצחון, לדוגמא:

```
Current Board:
1 2 3 4
5 6 7 8
9 A B C
C E F *
Well done, you won!
```

ומיד לאחר מכן יסתיים המשחק.

דוגמת הרצה

```
Please enter number of rows:
4
Please enter number of columns:
3
Please enter game duration in seconds:
500
Please enter the sliding puzzle:
12345679CAB
We are all set. Total time: 500 sec
Press Enter key when you are ready…
Current Board:
1 2 3
4 5 6
7 * 8
A B 9
Please enter your move:
L
Current Board:
1 2 3
4 5 6
7 8 *
A B 9
Please enter your move:
L
Illegal move. Try again
U
Current Board:
1 2 3
4 5 6
7 8 9
A B *
Well done, you won!
```

קבלת קלט מהמשתמש

בחלק זה נסביר כיצד על התוכנית להתנהג כאשר היא מחכה לקלט מהמשתמש.
את קבלת הקלט תממשו ע"י השיגרה scanf. על השיגרה להציג את התווים שהמשתמש מקליד (echo), ולאפשר למשתמש למחוק תווים שהקליד באמצעות שימוש במקש Backspace. אם הוכנס Backspace ואין תווים למחוק יש להתעלם ממנו. כאשר המשתמש לוחץ על Enter קבלת הקלט מסתיימת, והתווים שמוצגים באותה עת על המסך מפורשים כמחרוזת הסופית שהוכנסה כקלט ומוחזרים לפונקציה הקוראת ל scanf.
אם הזמן שהוקצב ע"י שחקן א' תם בזמן ההמתנה לקלט, scanf תדליק את הדגל times_up ותסיים.
ניתן להניח שאורך הקלט המקסימאלי המתקבל הינו 50 תווים. נדגיש כי אם במהלך ההקלדה הוקש Backspace אין להחשיב את התווים שנמחקו במסגרת 50 התווים המותרים וניתן להכניס תווים חדשים במקומם.
להלן פירוט תפקיד השיגרה והמנשק שלה. הקפידו לממש את המנשק במדויק.

שם השיגרה	תפקיד השיגרה	אוגר קישור	פרמטרים ושטח העברתם
scanf	מקבלת קלט מהמקלדת עד הקשת Enter ומכניסה את המחרוזת שהתקבלה למערך היעד.	**pc**	**קלט:** כתובת מערך היעד מועברת במחסנית. **פלט:** המחרוזת שהוקלדה ע"י המשתמש פרט ל Enter. סוף המחרוזת יסומן ע"י הערך 0. הדגל times_up בשטח משותף יקבל את הערך 1 אם תם הזמן תוך כדי ההמתנה לקלט, אחרת ערכו יהיה 0.

הערות חשובות:
- השיגרה scanf תניח שמערך היעד ארוך מספיק כדי להכיל את המחרוזת **הסופית** שהוקלדה ע"י המשתמש. שימו לב שגם אם ידוע שאורך המחרוזת הסופית קטן מ-50 תווים, עדיין ייתכן שבזמן קבלת הקלט המשתמש הכניס 50 תווים ומחק חלק לפני הקשת ה- Enter. scanf תתמוך במצב זה.
- כל פעולות <u>הקלט</u> המתבצעות במהלך התוכנית צריכות להתבצע באמצעות פסיקות. המשמעות היא שאין לבנות לולאה הדוקה הבודקת את מצבו של הדגל Done באוגר המנשק TKS.
- הדפסת התווים המוקלדים תתבצע בשיטת polling, כלומר באמצעות לולאה הדוקה שדוגמת את מצב הדגל Ready באוגר המנשק TPS.

הדפסת מחרוזות

בחלק זה נסביר כיצד התוכנית תדפיס מחרוזות שמקורן **אינו** תהליך ה- echo בקבלת הקלט.
את הדפסת המחרוזות תממשו ע"י השיגרה printf.
פרט לתווים שמודפסים בתהליך ה- echo בקבלת הקלט, כל מחרוזת שתודפס למסך תודפס ע"י קריאה ל- printf.
להלן פירוט תפקיד השיגרה printf והמנשק שלה. הקפידו למשש את המנשק במדויק.

שם השיגרה	תפקיד השיגרה	אוגר קישור	פרמטרים ושטח העברתם
printf	מקבלת כתובת מחרוזת ומדפיסה את המחרוזת ע"י מנגנון הפסיקות. **השיגרה מסתיימת רק כאשר המחרוזת הודפסה במלואה.**	**pc**	**קלט:** כתובת המחרוזת תועבר במחסנית. סוף המחרוזת יסומן ע"י הערך 0. **פלט:** המחרוזת תודפס למסך

הערות חשובות:
- כל פעולות <u>הפלט</u> המתבצעות במהלך printf צריכות להתבצע באמצעות פסיקות. המשמעות היא שאין לבנות לולאה הדוקה הבודקת את מצבו של הדגל Ready באוגר המנשק TPS.
- printf תסיים את ריצתה רק בתום הדפסת המחרוזת.

השעון

עליכם להיעזר בפסיקת השעון בשביל למדוד את משך הזמן שעובר.
יש להשתמש בתווית בשם rate המצביעה למילה המציינת כמה פסיקות שעון יוזם השעון בשנייה.
עליכם להשתמש בנתון זה ולא להניח שהוא 50 כפי שנלמד בשיעור. התווית עצמה תתוסף לתוכנית באופן אוטומטי במהלך תהליך הבדיקה באמצעות שורה כדוגמת:

```
rate:    .word 1000.
```

בגלל אופן בניית הסימולטור, מספר פסיקות השעון בשנייה משתנה ממחשב למחשב, ועשוי אף להגיע לכמה אלפים.

רמזים\טיפים

בחלק זה יוצגו פריטי מידע ומספר המלצות מימוש שיעזרו לכם לבנות את התוכנית.

- מסיבות היסטוריות המקש Enter מזוהה עם זוג תווי ה-ASCII:
  - Line Feed (LF) שמיוצג על-ידי המספר 10 (בסיס דצימלי).
  - Carriage Return (CR) שמיוצג על-ידי המספר 13 (בסיס דצימלי).

  בסימולטור שבידיכם כאשר המשתמש לוחץ על המקש Enter, התו שיתקבל יהיה CR בלבד. לעומת זאת, כאשר ברצונכם לעבור לשורה הבאה, עליכם להדפיס את התו LF ולאחריו את התו CR.
- המקש Backspace (BS) מיוצג בטבלת ה-ASCII על-ידי המספר 8. הדפסת התו BS תחזיר את "סמן המערכת" מקום אחד שמאלה: כלומר, אם התו הבא שיוקלד אמור להיות מודפס במקום החמישי בשורה ונקליד לפניו 3 פעמים את התו BS, אז התו יודפס במקום השני בשורה ולא במקום החמישי. נדגיש כי התווים שהודפסו כבר לא ימחקו, אבל ניתן כמובן להדפיס תווים אחרים שיחליפו אותם. חישבו כיצד ניתן להשתמש בתכונה זו בשביל למחוק תווים שהודפסו.
- חשוב לשים לב לקדימויות של הפסיקות השונות. שימו לב למה שיקרה במידה שתתרחש פסיקה תוך כדי הטיפול בפסיקה קודמת.

תהליך בדיקת נכונות התוכנית

לסימולטור מספר פקודות שלא נלמדו בשיעור. אחת הפקודות הללו היא P. שימוש אפשרי בפקודה זו היא השורה:

Pclk_cycle=200

שורה זו קובעת ל- 200 את היחס בין קצב פסיקות שעון המעבד (של המחשב עליו הסימולטור רץ), לקצב פסיקות השעון בסימולטור. כחלק מבדיקת התרגיל, השורה הנ"ל תיכתב בכל הרצות הסימולטור שישמשו לבדיקת התוכנית, לפני הרצת התוכנית. אנו ממליצים לכם לבדוק את התוכנית בתנאים זהים, כלומר לכתוב את השורה הנ"ל בכל פעם שאתם מפעילים את הסימולטור (די לכתוב שורה זו פעם אחת עבור כל הפעלה של הסימולטור).

לפני הרצת התוכנית, אנו נוסיף את התווית rate, לסוף הקובץ אותו אתם מגישים, בכתובת מעל 12000_8. לכן, אין להשתמש בכתובות מעל 12000_8 בכתיבת התוכנית. כמו כן, אין להגיש קובץ המכיל את הגדרת התווית הנ"ל (שכן הגדרה זו מוספת במהלך הבדיקה). אתם, כמובן, רשאים להוסיף תווית זו במהלך כתיבת התוכנית וניפוי השגיאות (debugging).
לצורך הבהרת עניין זה, יסופקו שני קבצים: ex4_test1.txt ו- ex4_test.bat. הקובץ ex4_test1.txt מכיל את הגדרת התווית, והקובץ ex4_test.bat הוא קובץ הרצה המשמש להוספת התווית. עליכם לבצע את הפעולות הבאות לפני הגשת התרגיל:

1. יש לוודא כי שם הקובץ של התוכנית הוא ex4.s11,
2. להוריד את שני הקבצים (ex4_test1.txt ו-ex4_test.bat) מהאתר לאותו המיקום בו נמצא קובץ התוכנית.
3. להריץ את הקובץ ex4_test.bat.
4. ייוצר קובץ חדש בשם ex4_temp.s11 המכיל את קוד התוכנית המקורי (מהקובץ ex4.s11) וכן את הגדרת התוויות (מהקובץ ex4_test1.txt). יש לוודא כי עבור הקובץ החדש אין שגיאה בזמן תרגום וכי התוכנית מביאה לפלט הצפוי.
5. בכל אופן, יש להגיש את הקובץ ex4.s11.

שימו לב: לא יתקבלו ערעורים הקשורים בעניין הטכני הנ"ל.

הערות נוספות

1. התוכנית צריכה לפעול נכון עבור **כל** קלט שעומד בהנחות המתאימות.
2. התוכנית צריכה לרוץ על הסימולטור המסופק באתר הקורס.
3. שימו לב כי באתר יהיה FAQ עבור תרגיל זה אשר יעודכן באופן סדיר. **אנא בדקו תחילה אם התשובה לשאלתכם מופיעה ב- FAQ**.
4. **יש להקפיד על תיעוד פנימי וחיצוני של התוכנית.** יורדו נקודות בגין תיעוד לא מלא. ניתן לקרוא באתר הקורס ב-FAQ על רמת התיעוד הנדרשת.
5. שאלות על התרגיל יש להפנות **למאיה לוי** בלבד.
6. **הגשות באיחור יש לתאם לפני מועד ההגשה.**
7. הגשה לתא הקורס: חלק יבש + תיעוד חיצוני (אין צורך להגיש את התוכנית מודפסת).
הגשה אלקטרונית: קובץ הקוד ex4.s11 בלבד.

עבודה נעימה!