<u>עבודה מסכמת – מערכות הפעלה</u>

שאלה 1:

```
char globBuf[65536]; /* 1. Where is allocated? */
```

1. המשתנה globBuf נמצא ב c, משתנים גלובליים יכולים להימצא או ב uninitialized data segment לפי מבנה הזכרון של תכניות בשפת c, משתנים גלובליים יכולים להימצא או ב data segment נשפת data segment כאשר המשתנה ממוקם בחלק הראשון אם המתכנת לא איתחל אותו ובחלק השני כאשר המתכנת איתחל את המשתנה. globBuf לא אותחל אז הוא ממוקם ב uninitialized data segment מאחר ו צילומי מסך שמאשרים קביעה זו.

```
int primes[] = { 2, 3, 5, 7 };  /* 2. Where is allocated? */
```

.initialized data segment נמצא ב primes .a המשתנה.

deta segment או ב uninitialized, משתנים גלובליים יכולים להימצא או ב c לפי מבנה הזכרון של תכניות בשפת, משתנים גלובליים יכולים להימצא או ב data segment כאשר המשתנה ממוקם בחלק הראשון אם המתכנת לא איתחל אותו ובחלק השני כאשר המתכנת איתחל את המשתנה.

a.initialized data segment אותחל אז הוא ממוקם ב primes אותחל אז

*מופיעים בסוף הקובץ צילומי מסך שמאשרים קביעה זו.

```
static int square(int x) /* 3. Where is allocated? */
```

3. הפונקציה square נמצא ב square. לפי מבנה הזכרון של תכניות בשפת c, הוראות של פונקציות נמצאות ב text segment.

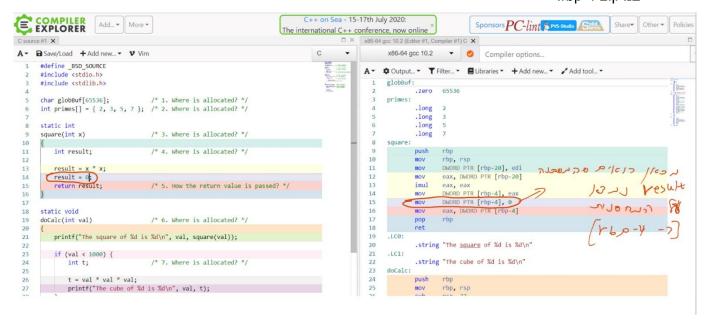
```
int result;  /* 4. Where is allocated? */
```

.stack נמצא ב result 4.

ניתן לראות מהצילום הבא שהשורה result=x*x מתבצעת והתוצאה נשמרת בסופו של דבר על המחסנית בהיסט של 4- מתחילת ה frame של המחסנית של הפונקציה ולכן מוכח מכאן שהמשתנה result נמצא על המחסנית.

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork_312488596/q_1
File Edit View Search Terminal Help
 ori@ori-VirtualBox:~/Desktop/OS-FinalWork/fwork_312488596/q_1$ objdump -D q1_312488596.o
01 312488596.0:
                        file format elf64-x86-64
Disassembly of section .text:
 000000000000000 <square>:
          48 89 e5
         89 7d ec
8b 45 ec
0f af 45 ec
89 45 fc
                                                %edi,-0x14(%rbp)
-0x14(%rbp),%eax
-0x14(%rbp),%eax
%eax,-0x4(%rbp)
                                                                                Pint Berlie may nextly luckly
                                        mov
imul
          8b 45 fc
5d
                                                 -0x4(%rbp),%eax
                                                                                                                 א הינתסטית.
 000000000000016 <doCalc>:
                                                 %гьр
          48 89 e5
48 83 ec 20
                                        mov
sub
                                                 %rsp,%rbp
                                                 $0x20,%rsp
%edi,-0x14(%rbp)
-0x14(%rbp),%eax
          89 7d ec
8b 45 ec
                                        MOV
                                        mov
callq
                                                %eax,%edi
0 <square>
          e8 d5 ff ff ff
                                        MOV
MOV
                                                  -0x14(%rbp),%eax
                                        mov
lea
                                                 %eax,%esi
0x0(%rip),%rdi
                 3d 00 00 00 00
                                                                            # 39 <doCalc+0x23>
```

בצילום הבא ניתן לראות את קוד האסמבלי של התכנית (מהאתר godbolt) עם הוספה של שורה 14 (result ב tesult), ניתן לראות מקוד האסמבלי של שורה זו שהמשתנה result נמצא על המחסנית במיקום rbp-4.



return result; /* 5. How the return value is passed? */

.5. ערך החזרה מועבר על-ידי רגיסטר

בצילום הבא של קוד האסמבלי של התכנית ניתן לראות שהערך שנמצא במשתנה result (בפונקציה square) מועתק לאוגר eax, כמו כן בקוד האסמבלי של הפונקציה doCalc ניתן לראות שלאחר square) מועתק לאוגר square מועתק הערך שנמצא באוגר eax שכמו שאמרנו בהתחלה הוא מכיל את result שזה ערך החזרה של הפונקציה square.

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork_312488596/q_1
File Edit View Search Terminal Help
ori@ori-VirtualBox:~/Desktop/OS-FinalWork/fwork_312488596/q_1$ objdump -D q1_312488596.o
q1_312488596.o:
                               file format elf64-x86-64
Disassembly of section .text:
 00000000000000000 <square>:
            55
48 89 e5
                                                  push
                                                             %гьр
                                                             %rsp,%rbp
%rsp,%rbp
%edi,-0x14(%rbp)
-0x14(%rbp),%eax
-0x14(%rbp),%eax
%eax,-0x4(%rbp)
-0x4(%rbp),%eax
                                                   MOV
            89 7d ec
8b 45 ec
0f af 45 ec
89 45 fc
                                                   mov
    e:
11:
                                                                                                                                      result
                                                  MOV
                                                                                                                                                        ( Look La
            8b 45 fc
5d
    14:
                                                              %гьь
           000000016 <doCalc>:
                                                             %rbp
%rsp,%rbp
                                                  push
                                                             48 83 ec 20
89 7d ec
8b 45 ec
                                                   sub
                                                  MOV
                                                  MOV
                                                                                       הדוך ל האנור אם [פהטנינו זת שה שה של בה שרך התנרה שם square)
            e8 d5 ff ff ff
89 c2
   26:
2b:
                                                              0 <square>
%eax,%edx }
-0x14(%rbp),%eax
                                                  calla
                                                              0 <square>
                                                  MOV
            89 c2
80 45 ec
89 c6
48 8d 3d 00 00 00 00
88 00 00 00 00
81 7d ec e7 03 00 00
   2d:
30:
                                                  MOV
                                                   mov
lea
                                                              0x0(%rip),%rdi
                                                                                                 # 39 <doCalc+0x23>
            b8 00 00 00 00 81 7d ec e7 7f 29 8b 45 ec 0f af 45 ec 6b 55 ec 0f af c2 89 45 fc 8b 55 fc
                                                             3000, Heax
43 <doCalc+0x2d>
$0x3e7, -0x14(%rbp)
75 <doCalc+0x5f>
-0x14(%rbp), Weax
-0x14(%rbp), Weax
-0x14(%rbp), Wedx
   3e:
                                                  calla
                                                   cmpl
                                                  jg
mov
    4a:
   4c:
4f:
53:
56:
                                                   imul
                                                  mov
imul
                                                              %edx,%eax
                                                              %eax,-0x4(%rbp)
-0x4(%rbp).%edx
```

```
static void doCalc(int val) /* 6. Where is allocated? */
```

.text segment נמצא ב doCalc

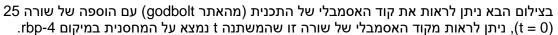
לפי מבנה הזכרון של תכניות בשפת c, הוראות של פונקציות נמצאות ב text segment.

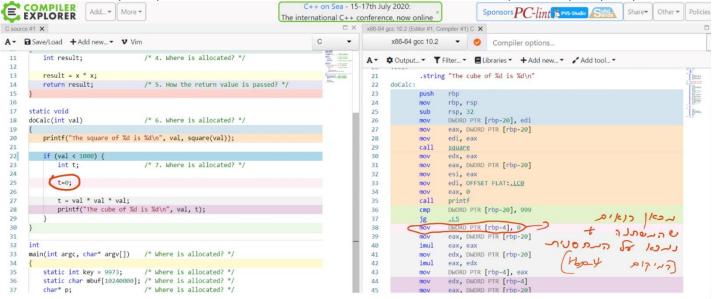
```
int t;  /* 7. Where is allocated? */
```

.stack נמצא ב t המשתנה.

בצילום הבא ניתן לראות שהשורה t=val*val*val מתבצעת והתוצאה נשמרת בסופו של דבר על המחסנית בהיסט של 4- מתחילת ה frame של המחסנית של הפונקציה ולכן מוכח מכאן שהמשתנה t נמצא על המחסנית.

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork_312488596/q_1
             0f af 45 ec
89 45 fc
8b 45 fc
                                                                             -0x14(%rbp),%eax
%eax,-0x4(%rbp)
-0x4(%rbp),%eax
                                                               imul
 e:
11:
14:
15:
                                                               MOV
MOV
                                                               pop
retq
                                                                             %гьр
0000000000000016 <doCalc>:
                                                                            %rbp
%rsp,%rbp
$0x20,%rsp
%edi,-0x14(%rbp)
-0x14(%rbp),%eax
              55
48 89 e5
                                                               push
                                                               mov
sub
             48 89 e5
48 83 ec 20
89 7d ec
8b 45 ec
89 c7
e8 d5 ff ff ff
                                                               MOV
MOV
  1e:
  21:
24:
26:
2b:
2d:
30:
32:
39:
44:
46:
53:
56:
59:
55:
                                                               mov
callq
                                                                            %eax,%edi
0 <square>
              89 c2
8b 45 ec
                                                               mov
mov
                                                                             %eax,%edx
-0x14(%rbp),%eax
             8b 45 ec
89 c6
48 8d 3d 00 00 00 00
b8 00 00 00 00
88 00 00 00 00
81 7d ec e7 03 00 00
7f 29
8b 45 ec
0f af 45 ec
8b 55 ec
0f af c2
89 45 fc
8b 55 fc
8b 45 ec
8b 45 ec
                                                               mov
lea
                                                                             %eax,%esi
0x0(%rip),%rdi
                                                                                                                          # 39 <doCalc+0x23>
                                                                            $0x0,%eax
43 <doCalc+0x2d>
$0x3e7,-0x14(%rbp)
75 <doCalc+0x5f>
                                                               mov
callq
                                                               cmpl
jg
mov
imul
                                                                            -0x14(%rbp),%eax
-0x14(%rbp),%eax
-0x14(%rbp),%edx
                                                               mov
imul
                                                                                                                                באונה ב נמפיל א הירחסות
                                                                         %edx,%eax
%eax,-0x4(%rbp)
-0x4(%rbp),%edx
-0x14(%rbp),%eax
                                                               MOV
                                                                                                                                                                               MOV
MOV
             89 C6
48 8d 3d 00 00 00 00
b8 00 00 00 00
e8 00 00 00 00
90
 62:
64:
                                                               mov
lea
                                                                             %eax,%esi
0x0(%rip),%rdi
                                                                                                                         # 6b <doCalc+0x55>
  6b:
70:
75:
76:
                                                               mov $0x0,%eax callq 75 <doCalc+0x5f>
                                                               nop
leaveq
              c9
000000000000078 <main>:
                                                              push %rbp
mov %rsp.%rbi
             55
48 89 e5
```





```
int main(int argc, char* argv[]) /* 8. Where is allocated? */
```

8. הפונקציה main נמצאת ב text segment.

לפי מבנה הזכרון של תכניות בשפת c, הוראות של פונקציות נמצאות ב text segment.

```
static int key = 9973;  /* 9. Where is allocated? */
```

.initialized data segment נמצא ב key המשתנה.9

לפי מבנה הזכרון של תכניות בשפת c, משתנים סטטיים יכולים להימצא או ב cuninitialized data לפי מבנה הזכרון של תכניות בשפת c initialized data segment כאשר המשתנה ממוקם בחלק הראשון אם המתכנת לא איתחל אותו ובחלק השני כאשר המתכנת איתחל את המשתנה.

a.initialized data segment אותחל אז הוא ממוקם ב key אותחל אז

*מופיעים בסוף הקובץ צילומי מסך שמאשרים קביעה זו.

```
{f static} char mbuf[10240000]; /* 10. Where is allocated? */
```

10. המשתנה mbuf נמצא ב mbuf ומצא ב toss) uninitialized data segment.

לפי מבנה הזכרון של תכניות בשפת c, משתנים סטטיים יכולים להימצא או ב uninitialized data לפי מבנה הזכרון של תכניות בשפת c initialized data segment כאשר המשתנה ממוקם בחלק הראשון אם המתכנת לא איתחל אותו ובחלק השני כאשר המתכנת איתחל את המשתנה.

.uninitialized data segment לא אותחל אז הוא ממוקם ב mbuf לא אותחל אז

*מופיעים בסוף הקובץ צילומי מסך שמאשרים קביעה זו.

```
char* p; /* 11. Where is allocated? */
```

11. המשתנה p לא ממוקם בשום אזור זכרון.

תחילה נשים לב ש p הינו משתנה מקומי ששייך ל main ולכן אינו יכול להיות שייך ל uninitialized data segment (מכיוון שרק משתנים סטטיים uninitialized data segment) ול text segment (כי רק פקודות ממוקמות שם), בנוסף ניתן לראות by במופיע בפלטים של הפקודות שמופיעים בסוף הקובץ שמראים את מה שנמצא באזורי הזכרון הללו.

כמו כן p לא נמצא על ה heap מאחר ולא התבצעה הקצאת זכרון דינמית, לכן האפשרות היחידה שנותרה היא ש p נמצא על המחסנית.

על מנת להראות ש p לא נמצא על המחסנית הוספתי צילום מסך של קוד האסמבלי של ה main פעם אחת עם ההצהרה של המשתנה p (כמו התכנית המקורית) ופעם אחרת ללא הצהרה של המתשנה p, ניתן לראות ששניהם הקודים של האסמבלי זהים לחלוטין ולכן ניתן להסיק מכך שלמשתנה p לא מוקצה כלל זכרון.

.p עם ההצרה על המשתנה p הצילום הבא הוא של קוד האסמבלי של ה

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork_312488596/q_1
File Edit View Search Terminal Help
  70:
        e8 00 00 00 00
                                  callq 75 <doCalc+0x5f>
        90
  75:
                                  nop
  76:
        c9
                                  leaveq
                                  retq
0000000000000078 <main>:
  78:
                                  push
                                          %гьр
  79:
        48 89 e5
                                          %rsp,%rbp
                                  MOV
        48 83 ec 10
                                          $0x10,%rsp
  7c:
                                  sub
  80:
        89 7d fc
                                  mov
                                          %edi,-0x4(%rbp)
                                          %rsi,-0x10(%rbp)
0x0(%rip),%eax
        48 89 75 f0
  83:
                                  mov
        8b 05 00 00 00 00
                                                                 # 8d <main+0x15>
  87:
                                  mov
  8d:
        89 c7
                                  MOV
                                          %eax,%edi
  8f:
        e8 82 ff ff ff
                                  callq
                                         16 <doCalc>
        bf 00 00 00 00
  94:
                                          $0x0,%edi
                                  mov
                                  callq 9e <main+0x26>
        e8 00 00 00 00
Disassembly of section .data:
0000000000000000 <primes>:
                                          (%rax),%al
   0:
        02 00
                                  add
        00 00
                                          %al,(%rax)
                                  add
   2:
        03 00
                                          (%rax),%eax
   4:
                                  add
                                          %al,(%rax)
        00 00
                                  add
   8:
        05 00 00 00 07
                                  add
                                          $0x7000000, %eax
        00 00
                                          %al,(%rax)
   d:
                                  add
```

הצילום הבא הוא של קוד האסמבלי של ה main ללא ההצהרה של המשתנה p,ניתן לראות שהקודים זהים לחלוטין ומכאן ניתן להסיק שהמשתנה p לא נמצא על המחסנית.

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork 312488596/q 1
File Edit View Search Terminal Help
                                          0x0(%rip),%rdi
        48 8d 3d 00 00 00 00
                                                                  # 6b <doCalc+0x55>
  64:
                                  lea
  6b:
        b8 00 00 00 00
                                  mov
                                          $0x0,%eax
        e8 00 00 00 00
                                  callq 75 <doCalc+0x5f>
  70:
  75:
        90
                                  nop
        c9
  76:
                                   leaveq
        с3
                                  retq
00000000000000078 <main>:
                                   push
        48 89 e5
                                          %rsp,%rbp
  79:
                                  MOV
                                          $0x10,%rsp
%edi,-0x4(%rbp)
        48 83 ec 10
  7c:
                                  sub
        89 7d fc
  80:
                                  mov
  83:
        48 89 75 f0
                                  mov
                                          %rsi,-0x10(%rbp)
        8b 05 00 00 00 00
                                                                  # 8d <main+0x15>
  87:
                                  mov
                                          0x0(%rip),%eax
        89 c7
                                          %eax,%edi
  8d:
                                  MOV
        e8 82 ff ff ff
  8f:
                                  callq 16 <doCalc>
  94:
        bf 00 00 00 00
                                  MOV
                                          $0x0,%edi
  99:
        e8 00 00 00 00
                                  callq 9e <main+0x26>
Disassembly of section .data:
00000000000000000 <primes>:
        02 00
                                          (%rax),%al
   0:
                                   add
        00 00
                                   add
                                          %al,(%rax)
                                          (%rax),%eax
%al,(%rax)
        03 00
                                   add
        00 00
                                   add
   6:
        05 00 00 00 07
                                   add
                                          $0x7000000, %eax
```

בסופו של דבר קיבלנו שהמשתנה p לא נמצא באף אזור זכרון.

מצורפים כאן צילומי מסך של שימוש בכלים בלינוקס שמאשרים את התשובות שלי עבור שאלות 1,2,3,6,8,9,10

בתמונה א ניתן לראות את הפלט של הפקודה nm -n על הקובץ המקומפל של התכנית, עבור כל אחד מהשמות של המשתנים/הפונקציות שמופיעות בפלט מופיעה אות באנגלית שמציינת את המקום שבו נמצא המשתנה/הפונקציה בזיכרון (על פי ה man) –

b/C - uninitialized data section

D/d - initialized data section

T/t - text section

בתמונה ב ניתן לראות את הפלט של הפקודה objdump -x עבור את הפלט של התוכנית, עבור square, doCalc, key, mbuf,) כל אחד מהשמות של המשתנים/הפונקציות שמופיעות בפלט (globBuf, primes, main) מופיע בפירוש המקום שבו נמצא המשתנה/הפונקציה בזיכרון

ב.

```
ori@ori-VirtualBox: ~/Desktop/OS-FinalWork/fwork_312488596/q_1
File Edit View Search Terminal Help
ori@ori-VirtualBox:~/Desktop/OS-FinalWork/fwork_312488596/q_1$ objdump -x q1_312488596.o
q1_312488596.o:
               file format elf64-x86-64
q1 312488596.o
architecture: i386:x86-64, flags 0x00000011:
HAS_RELOC, HAS_SYMS
start address 0x0000000000000000
Sections:
Idx Name
              Size
                                                   File off
                                                           Algn
              0000009e
                                    0000000000000000
                                                           2**0
 0 .text
                      0000000000000000
                                                   00000040
              CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
                                                           2**4
 1 .data
              00000014 0000000000000000
                                    00000000000000000
                                                   000000e0
              CONTENTS, ALLOC, LOAD, DATA
 2 .bss
              00000100 2**5
              ALLOC
 3 .rodata
              00000100 2**0
              CONTENTS, ALLOC, LOAD, READONLY, DATA
 4 .comment
              0000012e 2**0
              CONTENTS, READONLY
 CONTENTS, READONLY
              00000158 2**3
 6 .eh_frame
              CONTENTS, ALLOC, LOAD, RELOC, READONLY, DATA
SYMBOL TABLE:
00000000000000000000001
                 df *ABS*
                         0000000000000000 q1_312488596.c
d .text
                         000000000000000 .text
                   .data
0000000000000000 l
                         000000000000000 .data
0000000000000000 .bss
                    .bss
                  F.text 000000000000016 square
.rodata
0000000000000000 l
                               0000000000000000
                                             .rodata
                    .text 0000000000000062 doCalc
00000000000000016 l
                  0000000000000010 l
.note.GNU-stack
                                      0000000000000000 .note.GNU-stack
d
                               0000000000000000 .eh_frame
00000000000000000000001
                 d
                   .eh_frame
0000000000000000000001
                   .comment
                               0000000000000000 .comment
                  0 *COM*
0 .data
0000000000010000
                         00000000000000020 globBuf
00000000000000000000
                         0000000000000010 primes
                    0000000000000000
0000000000000000
00000000000000078 g
                  F.text 0000000000000026 main
00000000000000000
                    *UND*
                         00000000000000000 exit
```