207205972

שאלה מספר 1-

לצורך מענה על שאלה זו השתמשתי בשני כילים : **nm, objdump** (תמונות של הרצת הפקודות הללו מצורפות מטה).

```
char globBuf[65536];        // 1. Uninitialized data section   .1
```

```
00000000009c8060 B globBuf
```

1. where is allocated?

כפי שניתן לראות בטרמינל, עבור הסמל **globBuf** קיבלתי את האות **'B'**
והמשמעות היא שהסמל globBuf נמצא ב- **uninitialized data section** .

```
int primes[] = { 2, 3, 5, 7 };  // 2. Initialized data section.2
```

```
0000000000004010 D primes
```

2. Where is allocated?

עבור הסמל **primes** קיבלתי את האות **'D'** והמשמעות היא שהסמל נמצא ב- **initialized data section**.

```
square(int x)        //3. allocated in frame for square().3
```

```
0000000000001169 t square
```

3. Where is allocated?

עבור הסמל **square** קיבלתי את האות **'t'** והמשמעות היא שהסמל נמצא ב- **Text section**.

```
int result; // 4. allocated in the Stack of square() function   .4
```

```
4:   55                    push   %rbp
5:   48 89 e5              mov    %rsp,%rbp
```

4. Where is allocated?

בשאלה זו השתמשתי בפקודה **objdump -d** . כפי שניתן לראות בטרמינל - ב frame של square() כאשר int result הוצהר- מה שקרה מאחורי הקלעים זה ש result הוכנס לתוך המחסנית של הפונקציה square() (ע"י הפקודה push) **ולכן ניתן להסיק שהמשתנה result ממוקם על stack.**

```
return result;   // 5. return value is passed via register   .5
```

```
4:   55                    push   %rbp
5:   48 89 e5              mov    %rsp,%rbp
```

```
14:   8b 45 fc             mov    -0x4(%rbp),%eax
17:   5d                   pop    %rbp
18:   c3                   retq
```

5. How the return value is passed ?

בשאלה זו השתמשתי בפקודה **objdump -d** . בסעיף 4 דחפנו את result לתוך רגיסטר %rbp ע"י push ובסוף כפי שניתן לראות בטרמינל - מתבצע pop למשתנה result **ולכן ניתן להסיק כי הוא מועבר ע"י רגיסטר.**

```
doCalc(int val) // 6. allocated in frame for doCalc()   .6
```

```
0000000000001182 t doCalc
```

6. where is allocated?

עבור הסמל **doCalc** קיבלתי את האות **'t'** והמשמעות היא שהסמל נמצא ב- **Text section.**

```
int t;  // 7. allocated in the Stack of doCalc() function    .7
```

```
  51:   7f 28                   jg     7b <doCalc+0x62>
  53:   8b 45 ec                mov    -0x14(%rbp),%eax
```

7.Where is allocated?

כפי שניתן לראות בטרמינל, jg זה התנאי if שלפני הצהרת המשתנה t.
כאשר נכנסנו לתנאי (if), ניתן לראות כי מתבצע mov למשתנה t **ולכן אפשר להסיק כי המשתנה t נמצא על הstack.**

```
main(int argc, char* argv[]) // 8. allocated in frame for main()   .8
```

```
00000000000011e7 T main
```

8. Where is allocated?

עבור הסמל **main** קיבלתי את האות **'T'** והמשמעות היא שהסמל נמצא ב-**Text section.**

```
static int key = 9973;// 9. Initialized data section    .9
```

```
0000000000004020 d key.2841
```

9. Where is allocated?

עבור הסמל **key** קיבלתי את האות **'d'** והמשמעות היא שהסמל נמצא ב-**initialized data section.**

```
static char mbuf[10240000];  // 10. Uninitialized data section   .10
```

```
0000000000004060 b mbuf.2842
```

10. Where is allocated?

עבור הסמל **mbuf** קיבלתי את האות **'b'** והמשמעות היא שהסמל נמצא ב-**uninitialized data.**

```
char* p;                    // 11. allocated in the Stack   .11
```

```
82:   55                    push   %rbp
83:   48 89 e5              mov    %rsp,%rbp
86:   48 83 ec 10           sub    $0x10,%rsp
8a:   89 7d fc              mov    %edi,-0x4(%rbp)
8d:   48 89 75 f0           mov    %rsi,-0x10(%rbp)
```

11. Where is allocated?

כפי שניתן לראות בטרמינל - כאשר המשתנה p הוצהר , מה שקרה מאחורי הקלעים זה ש- p הוכנס לתוך הstack (ע"י הפקודה push) **ולכן ניתן להסיק שהמשתנה p ממוקם על stack.**

הרצת הפקודה objdump –d :

```
Activities   Terminal ▾                    Aug 9  15:57 ●
yarden@osboxes: ~/fwork_207205972/q_1
yarden@osboxes:~/fwork_207205972/q_1$ objdump -d a.o

a.o:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <square>:
   0:   f3 0f 1e fa             endbr64
   4:   55                      push   %rbp
   5:   48 89 e5                mov    %rsp,%rbp
   8:   89 7d ec                mov    %edi,-0x14(%rbp)
   b:   8b 45 ec                mov    -0x14(%rbp),%eax
   e:   0f af c0                imul   %eax,%eax
  11:   89 45 fc                mov    %eax,-0x4(%rbp)
  14:   8b 45 fc                mov    -0x4(%rbp),%eax
  17:   5d                      pop    %rbp
  18:   c3                      retq

0000000000000019 <doCalc>:
  19:   f3 0f 1e fa             endbr64
  1d:   55                      push   %rbp
  1e:   48 89 e5                mov    %rsp,%rbp
  21:   48 83 ec 20             sub    $0x20,%rsp
  25:   89 7d ec                mov    %edi,-0x14(%rbp)
  28:   8b 45 ec                mov    -0x14(%rbp),%eax
  2b:   89 c7                   mov    %eax,%edi
  2d:   e8 ce ff ff ff          callq  0 <square>
  32:   89 c2                   mov    %eax,%edx
  34:   8b 45 ec                mov    -0x14(%rbp),%eax
  37:   89 c6                   mov    %eax,%esi
  39:   48 8d 3d 00 00 00 00    lea    0x0(%rip),%rdi        # 40 <doCalc+0x27>
  40:   b8 00 00 00 00          mov    $0x0,%eax
```

207205972

```
0000000000000019 <doCalc>:
  19:   f3 0f 1e fa             endbr64
  1d:   55                      push   %rbp
  1e:   48 89 e5                mov    %rsp,%rbp
  21:   48 83 ec 20             sub    $0x20,%rsp
  25:   89 7d ec                mov    %edi,-0x14(%rbp)
  28:   8b 45 ec                mov    -0x14(%rbp),%eax
  2b:   89 c7                   mov    %eax,%edi
  2d:   e8 ce ff ff ff          callq  0 <square>
  32:   89 c2                   mov    %eax,%edx
  34:   8b 45 ec                mov    -0x14(%rbp),%eax
  37:   89 c6                   mov    %eax,%esi
  39:   48 8d 3d 00 00 00 00    lea    0x0(%rip),%rdi        # 40 <doCalc+0x27>
  40:   b8 00 00 00 00          mov    $0x0,%eax
  45:   e8 00 00 00 00          callq  4a <doCalc+0x31>
  4a:   81 7d ec e7 03 00 00    cmpl   $0x3e7,-0x14(%rbp)
  51:   7f 28                   jg     7b <doCalc+0x62>
  53:   8b 45 ec                mov    -0x14(%rbp),%eax
  56:   0f af c0                imul   %eax,%eax
  59:   8b 55 ec                mov    -0x14(%rbp),%edx
  5c:   0f af c2                imul   %edx,%eax
  5f:   89 45 fc                mov    %eax,-0x4(%rbp)
  62:   8b 55 fc                mov    -0x4(%rbp),%edx
  65:   8b 45 ec                mov    -0x14(%rbp),%eax
  68:   89 c6                   mov    %eax,%esi
  6a:   48 8d 3d 00 00 00 00    lea    0x0(%rip),%rdi        # 71 <doCalc+0x58>
  71:   b8 00 00 00 00          mov    $0x0,%eax
  76:   e8 00 00 00 00          callq  7b <doCalc+0x62>
  7b:   90                      nop
  7c:   c9                      leaveq
  7d:   c3                      retq
```

```
  40:   b8 00 00 00 00          mov    $0x0,%eax
  45:   e8 00 00 00 00          callq  4a <doCalc+0x31>
  4a:   81 7d ec e7 03 00 00    cmpl   $0x3e7,-0x14(%rbp)
  51:   7f 28                   jg     7b <doCalc+0x62>
  53:   8b 45 ec                mov    -0x14(%rbp),%eax
  56:   0f af c0                imul   %eax,%eax
  59:   8b 55 ec                mov    -0x14(%rbp),%edx
  5c:   0f af c2                imul   %edx,%eax
  5f:   89 45 fc                mov    %eax,-0x4(%rbp)
  62:   8b 55 fc                mov    -0x4(%rbp),%edx
  65:   8b 45 ec                mov    -0x14(%rbp),%eax
  68:   89 c6                   mov    %eax,%esi
  6a:   48 8d 3d 00 00 00 00    lea    0x0(%rip),%rdi        # 71 <doCalc+0x58>
  71:   b8 00 00 00 00          mov    $0x0,%eax
  76:   e8 00 00 00 00          callq  7b <doCalc+0x62>
  7b:   90                      nop
  7c:   c9                      leaveq
  7d:   c3                      retq

000000000000007e <main>:
  7e:   f3 0f 1e fa             endbr64
  82:   55                      push   %rbp
  83:   48 89 e5                mov    %rsp,%rbp
  86:   48 83 ec 10             sub    $0x10,%rsp
  8a:   89 7d fc                mov    %edi,-0x4(%rbp)
  8d:   48 89 75 f0             mov    %rsi,-0x10(%rbp)
  91:   8b 05 00 00 00 00       mov    0x0(%rip),%eax        # 97 <main+0x19>
  97:   89 c7                   mov    %eax,%edi
  99:   e8 7b ff ff ff          callq  19 <doCalc>
  9e:   bf 00 00 00 00          mov    $0x0,%edi
  a3:   e8 00 00 00 00          callq  a8 <main+0x2a>
yarden@osboxes:~/fwork_207205972/q_1$
```

הרצת הפקודה nm-

```
yarden@osboxes:~/fwork_207205972/q_1$ nm run
0000000000004024 B __bss_start
0000000000004040 b completed.8055
                 w __cxa_finalize@@GLIBC_2.2.5
0000000000004000 D __data_start
0000000000004000 W data_start
00000000000010b0 t deregister_tm_clones
0000000000001182 t doCalc
0000000000001120 t __do_global_dtors_aux
0000000000003db8 d __do_global_dtors_aux_fini_array_entry
0000000000004008 D __dso_handle
0000000000003dc0 d _DYNAMIC
0000000000004024 D _edata
00000000009d8060 B _end
                 U exit@@GLIBC_2.2.5
0000000000001298 T _fini
0000000000001160 t frame_dummy
0000000000003db0 d __frame_dummy_init_array_entry
00000000000021cc r __FRAME_END__
0000000000003fb0 d _GLOBAL_OFFSET_TABLE_
00000000009c8060 B globBuf
                 w __gmon_start__
0000000000002034 r __GNU_EH_FRAME_HDR
0000000000001000 t _init
0000000000003db8 d __init_array_end
0000000000003db0 d __init_array_start
0000000000002000 R _IO_stdin_used
                 w _ITM_deregisterTMCloneTable
                 w _ITM_registerTMCloneTable
0000000000004020 d key.2841
0000000000001290 T __libc_csu_fini
0000000000001220 T __libc_csu_init
                 U __libc_start_main@@GLIBC_2.2.5
```

```
0000000000004008 D __dso_handle
0000000000003dc0 d _DYNAMIC
0000000000004024 D _edata
00000000009d8060 B _end
                 U exit@@GLIBC_2.2.5
0000000000001298 T _fini
0000000000001160 t frame_dummy
0000000000003db0 d __frame_dummy_init_array_entry
00000000000021cc r __FRAME_END__
0000000000003fb0 d _GLOBAL_OFFSET_TABLE_
00000000009c8060 B globBuf
                 w __gmon_start__
0000000000002034 r __GNU_EH_FRAME_HDR
0000000000001000 t _init
0000000000003db8 d __init_array_end
0000000000003db0 d __init_array_start
0000000000002000 R _IO_stdin_used
                 w _ITM_deregisterTMCloneTable
                 w _ITM_registerTMCloneTable
0000000000004020 d key.2841
0000000000001290 T __libc_csu_fini
0000000000001220 T __libc_csu_init
                 U __libc_start_main@@GLIBC_2.2.5
00000000000011e7 T main
0000000000004060 b mbuf.2842
0000000000004010 D primes
                 U printf@@GLIBC_2.2.5
00000000000010e0 t register_tm_clones
0000000000001169 t square
0000000000001080 T _start
0000000000004028 D __TMC_END__
yarden@osboxes:~/fwork_207205972/q_1$
```