C Home_assignment

יהונתן ארמא 207938903

Part B:

1) ההבדל בין משתנים גלובליים ללוקאליים:
בשפת c, משתנים ניתן להגדיר בתוך פונקציות ומחוץ לפונקציות.
משתנים המוגדרים מחוץ לפונקציות נקראים 'משתנים גלובליים' ומשתנים המוגדרים בתוך פונקציות נקראים 'משתנים לוקאליים'.
ה-"scope" של המשתנים הגלובליים הוא כל התוכנית, כלומר כל פונקציה בתוכנית (יותר נכון ב-file כל עוד לא הגדרנו את המשתנה ב-extern) יכולה 'לראות' את המשתנה ולהשתמש בו.
ה-"scope" של המשתנים הלוקאליים הוא הפונקציה או "הסוגריים המסלוסלים" בהם הם הוגדרו, כלומר אם משתנה הוגדר בפונקציה מסוימת אז לא ניתן 'לראות' אותו בפונקציה אחרת, ואם משתנה הוגדר בתוך בלוק מסוים (לדוגמה "if") אז לא ניתן 'לראות' אותו מחוץ לבלוק.

עוד הבדל בין השניים – משתנים לוקאליים שמורים במחסים (Stack frame) בעוד שמשתנים גלובליים שמורים מחוץ ל-Stack. (BSS או Data)

בתוכנית שהוצגה לנו:
דוגמה למשתנה גלובלי:
maxTrace (שורה 34) – חוץ מזה שסומן לנו בגדול שזה החלק של המשתנים הגלובליים, ניתן להבין שזה משתנה גלובלי מכיוון שהוא לא בתוך אף פונקציה (הוא לא חלק משום "scope") ולכן ה-"scope" שלו הוא כל התוכנית.

דוגמה למשתנה לוקאלי:
Trace (שורה 101) – משתנה זה הוגדר לנו בתוך פונקציה (ComputeTrace) ולכן ה-"scope" שלו הוא רק בתוך הפונקציה הזאת.

2) ניתן לראות שכאשר עושים Debug לתוכנית מקבלים עליה נתונים בזמן אמת.
לדוגמה, בריצה זו קיבלנו שהערך של המשתנה Mat הוא "0x00d8fac8" שזה הכתובת של האיבר הראשון במערך.

Name	Value	Type
auxMat	0x00d8f930 {0x00d8f930 {-858993460, -858993460, -858993460, -858993460, -858993460, -858993460...	int[10][10]
ch	-52 '◆'	char
Mat	0x00d8fac8 {0x00d8fac8 {-858993460, -858993460, -858993460, -858993460, -858993460, -858993460...	int[10][10]
matTrace	-858993460	int
maxDiag	-858993460	int
offset	0	int
Selector	-52 '◆'	char
str	0x00d8f8e8 <Invalid characters in string.>	char[3]

בנוסף, הכתובת האחרונה של המערך תהיה הכתובת שקבלנו ועוד 4 (שזה גודל int) כפול M (אורך המטריצה) כפול M (רוחב המטריצה) פחות 1 (כי התחלנו לספור מ-0). ומכיוון שבמקרה שלנו M=10 אז נקבל שהאיבר האחרון במערך שמור ב-offset של 396 מההתחלה של המערך כלומר במרחק הקסה של 18C. ולכן יהיה שמור בכתובת "0x00d8fc54".

נבחין שכתובות אלו ישתנו עם כל הרצה של התוכנית ולא ישארו קבועות.

3) הפעם פתחנו את החלון של הרגיסטרים ב-Debug Mode, ושמנו BreakPoint בדיוק בהגעה לפונקציה ComputeTrace. קיבלנו את הערך הבא:

```
99   //-----------------------------------------------------------------
100  int ComputeTrace(int Mat[M][M]) {
101      int Trace = 0, i;
102      for (i = 0; i < M; i++) Trace += Mat[i][i];
103      return Trace;
104  }
105  //-----------------------------------------------------------------
106  //                    DiagonalsMax Computation
107  //-----------------------------------------------------------------
108  int DiagonalsMax(int Mat[M][M]) {
109      int Max = 0, i;
110      for (i = 0; i < M; i++) {
111          Max = max(Mat[i][i], Max);
112          Max = max(Mat[i][N - 1 - i], Max);
113      }
114      return Max;
115  }
116  //-----------------------------------------------------------------
117  //                    Matrix Transpose
118  //-----------------------------------------------------------------
100 %
Registers
EAX = 00FDF798 EBX = 011F1000 ECX = 00000003 EDX = 00000003 ESI = 00FDF424 EDI = 00FDF930 EIP = 00181780 ESP = 00FDF41C EBP = 00FDF930 EFL = 00000297
```

ניתן לראות שערך הרגיסטר ה- EIP (כלומר PC) הוא 00181780.

4) נבחין שזוהי הפונקציה FillMatrix ב- DisAssembly:

```
//-----------------------------------------------------------------
void FillMatrix(unsigned int Mat[M][M], int offset) {
00181930  push        ebp
00181931  mov         ebp,esp
00181933  sub         esp,0D8h
00181939  push        ebx
0018193A  push        esi
0018193B  push        edi
0018193C  lea         edi,[ebp-0D8h]
00181942  mov         ecx,36h
00181947  mov         eax,0CCCCCCCCh
0018194C  rep stos    dword ptr es:[edi]
0018194E  mov         ecx,offset _AF62F785_source@c (018C003h)
00181953  call        @__CheckForDebuggerJustMyCode@4 (018122Bh)
    int i, j;
    for (i = 0; i < M; i++) {
00181958  mov         dword ptr [i],0
0018195F  jmp         FillMatrix+3Ah (018196Ah)
00181961  mov         eax,dword ptr [i]
00181964  add         eax,1
00181967  mov         dword ptr [i],eax
0018196A  cmp         dword ptr [i],0Ah
0018196E  jge         FillMatrix+7Bh (01819ABh)
        for (j = 0; j < M; j++) {
00181970  mov         dword ptr [j],0
00181977  jmp         FillMatrix+52h (0181982h)
00181979  mov         eax,dword ptr [j]
0018197C  add         eax,1
0018197F  mov         dword ptr [j],eax
00181982  cmp         dword ptr [j],0Ah
00181986  jge         FillMatrix+79h (01819A9h)
            Mat[i][j] = (offset + i * M + j) % CEIL;
00181988  imul        eax,dword ptr [i],0Ah
0018198C  add         eax,dword ptr [offset]
0018198F  add         eax,dword ptr [j]
00181992  cdq
00181993  mov         ecx,64h
00181998  idiv        eax,ecx
0018199A  imul        eax,dword ptr [i],28h
0018199E  add         eax,dword ptr [Mat]
001819A1  mov         ecx,dword ptr [j]
001819A4  mov         dword ptr [eax+ecx*4],edx
        }
001819A7  jmp         FillMatrix+49h (0181979h)
    }
001819A9  jmp         FillMatrix+31h (0181961h)
}
001819AB  pop         edi
001819AC  pop         esi
001819AD  pop         ebx
001819AE  add         esp,0D8h
001819B4  cmp         ebp,esp
001819B6  call        __RTC_CheckEsp (018123Ah)
001819BB  mov         esp,ebp
001819BD  pop         ebp
001819BE  ret
```

ניתן להבחין כי:
הכתובת הראשונה היא 00181930
וזוהי כתובת הפונצקיה.

שם הפונקציה FillMatrix מרמז על כך שזאת הפונקציה שממלאת לנו את המטריצה ואכן הפונקציה הזאת מקבלת מטריצה (תאכלס עושה שימוש בכתובת של המטריצה וממלאת שם ערכים) וממלאת את הכניסות במטריצה בערכים לפי נוסחה.

גודל הפונקציה:
ניתן לראות שהפונקציה מתחילה בכתובת 00181930 ונגמרת בכתובת 001819BE.
סה"כ לאחר חיסור בהקסא:
8E כלומר 142 בייטים.

הפונקציה נשמרת ב-RAM באזור שנקרא Code segment
או בשם אחר Text Segment.
שזהו מיקום ReadOnly ב- Ram שנמצא בתחילתו.

5) נבחין שמשתנה auxMat הוא משתנה לוקאלי שמוגדר בתוך פונקציית ה-main.
כלומר ניתן לגשת אליו בכל פונקציית ה-main ורק בתוכה.
יש לשים לב שהוא נשלח לפונקציה אחרת מה-main, וזה אולי יכול לבלבל, אבל בעצם מה שנשלח זה הכתובת של המערך (כי ערך של משתנה מסוג מערך הוא בעצם הכתובת בה הוא שמור) ובגלל שב-C השימוש הוא ב-Call By Value אז הכתובת מגיעה לפונקציה הנקראת והיא יכול לערוך את המערך. אבל, לא בגלל שניתן לגשת למשתנה דרכה.

כתובתו בבפונקציית ה-main:

Name	Value
auxMat	0x00b7f974 0x00b7f974 {-858993460, -858993460, -858993460, -858993460, -858993460, -858993460, -858993460, ...}, ...}
ch	52 '4'
Mat	0x00b7fb0c {0x00b7fb0c {-858993460, -858993460, -858993460, -858993460, -858993460, -858993460, -858993460, ...}, ...}
matTrace	-858993460
maxDiag	-858993460
offset	0
Selector	52 '4'
str	0x00b7f92c <Invalid characters in string.>

והוא שמור על ה-stack.

6) לפי Disassembly:

```
    a = a > b ? a : b;
00221D07  mov         eax,dword ptr [a]
00221D0D  cmp         eax,dword ptr [b]
00221D13  jle         main+73h (0221D23h)
00221D15  mov         ecx,dword ptr [a]
00221D1B  mov         dword ptr [ebp-518h],ecx
00221D21  jmp         main+7Fh (0221D2Fh)
    a = a > b ? a : b;
00221D23  mov         edx,dword ptr [b]
00221D29  mov         dword ptr [ebp-518h],edx
00221D2F  mov         eax,dword ptr [ebp-518h]
00221D35  mov         dword ptr [a],eax
```