Case Western Reserve University

Department of Computer and Data Sciences

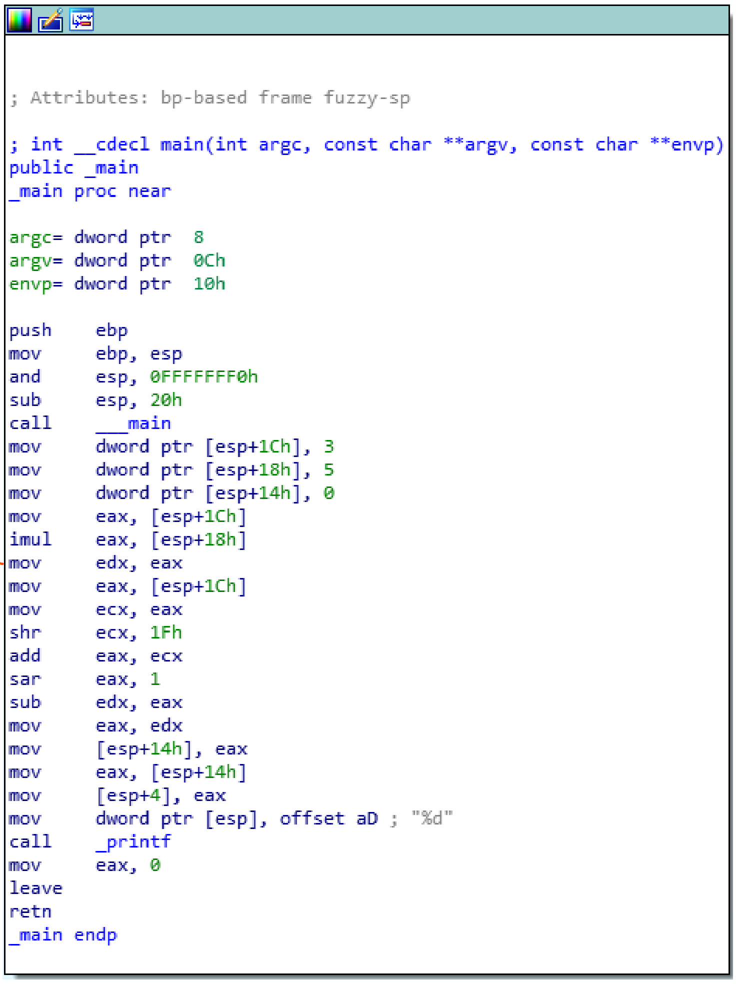
EECS 349&444: Computer Security

Assignment Date:	10/17/2019
Sumission Date:	10/23/2019@11:59pm
First Name:	Sabrina
Last Name:	Tay
Google Drive Link:	
Abstract of the feedback:	
I understand assembly is important but there's no class that goes so in depth w/ assemby Most of us only know the basics.	

* This is the third part of HW2 which contains 60 points. You are encouraged to finish independently. Any submitted work that it copied from any source or too similar to be an independent write-up will not be given credit. **Please post your solutions along with your detailed analysis and source codes in GitHub and provided your GitHub link for this submission on Canvas by 23:59pm on 10/23/2019.**

Q1: Assembly code is shown below. Please access its functionality and rewrite in C to printf() its output. (10pts)

```
push    ebp
mov     ebp, esp                  ebp = esp
and     esp, 0FFFFFFF0h           esp = 0
sub     esp, 20h                  esp-32
call    ___main
mov     dword ptr [esp+1Ch], 3
mov     dword ptr [esp+18h], 5
mov     dword ptr [esp+14h], 0
mov     eax, [esp+1Ch]            eax=3
imul    eax, [esp+18h]            eax = 3×5
mov     edx, eax                  edx = 15
mov     eax, [esp+1Ch]            eax = 3
mov     ecx, eax                  ecx = 3
shr     ecx, 1Fh                  ecx = 0 → because ecx was positive
add     eax, ecx                  eax = 3
sar     eax, 1                    eax = eax/2 = 1.
sub     edx, eax                  edx - eax = 14
mov     eax, edx                  eax = 14
mov     [esp+14h], eax            = 14
mov     eax= [esp+14h]
mov     [esp+4], eax              = 14
mov     dword ptr [esp], offset aD ; "%d"
call    _printf                   ecx = 0
mov     eax, 0                    eax = 14
leave
retn
_main endp
```



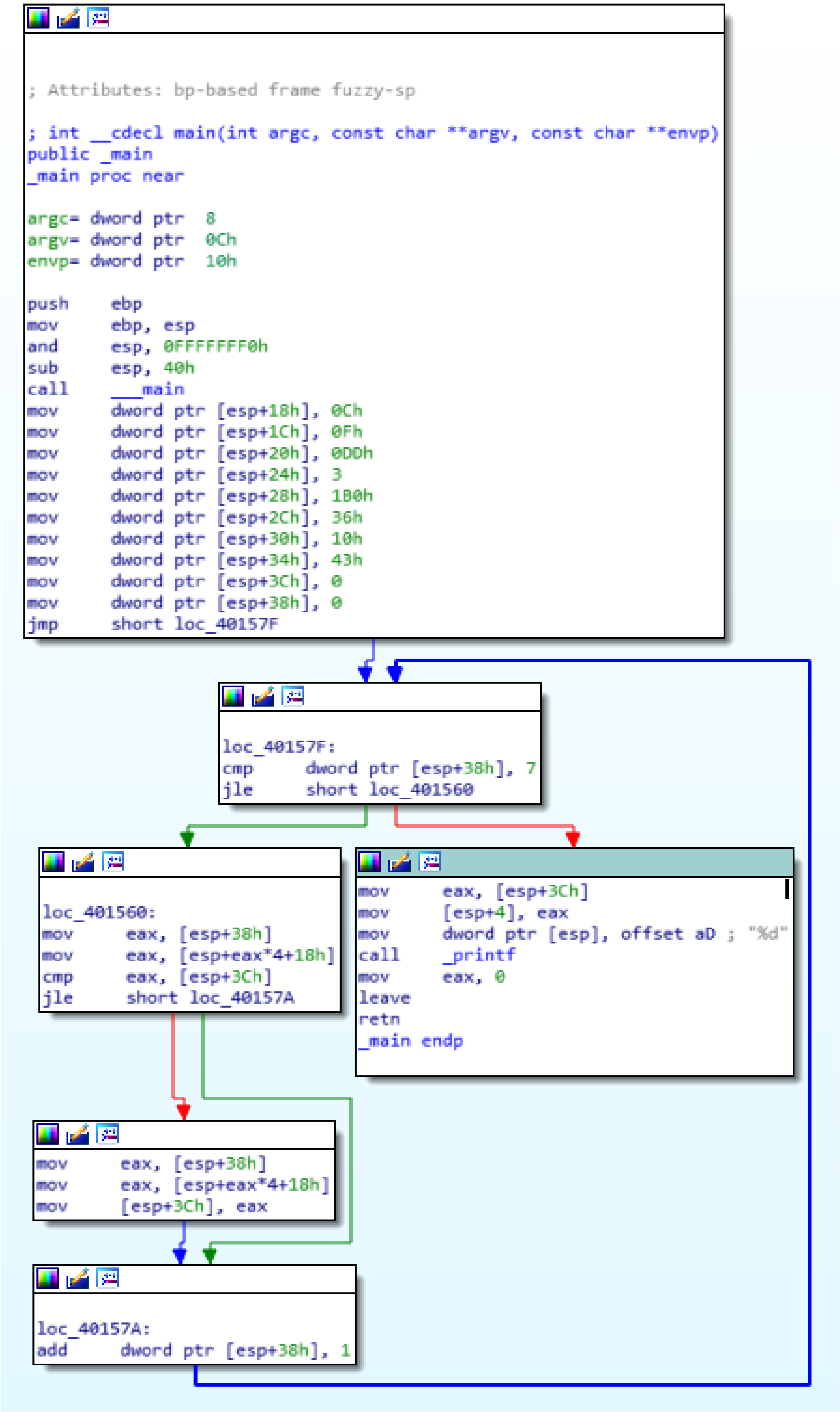
```
; Attributes: bp-based frame fuzzy-sp

; int __cdecl main(int argc, const char **argv, const char **envp)
public _main
_main proc near

argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF0h
sub     esp, 20h
call    ___main
mov     dword ptr [esp+1Ch], 3
mov     dword ptr [esp+18h], 5
mov     dword ptr [esp+14h], 0
mov     eax, [esp+1Ch]
imul    eax, [esp+18h]
mov     edx, eax
mov     eax, [esp+1Ch]
mov     ecx, eax
shr     ecx, 1Fh
add     eax, ecx
sar     eax, 1
sub     edx, eax
mov     eax, edx
mov     [esp+14h], eax
mov     eax, [esp+14h]
mov     [esp+4], eax
mov     dword ptr [esp], offset aD ; "%d"
call    _printf
mov     eax, 0
leave
retn
_main endp
```

A B C D E F
10 11 12 13 14 15

Q2: Assembly code is shown below. Please access its functionality and rewrite in C to printf() its output. (15pts)

```
.text:00401500                  push      ebp
.text:00401501                  mov        ebp, esp                       esp -
.text:00401503                  and       esp, 0FFFFFFF0h
.text:00401506                  sub       esp, 40h
.text:00401509                  call     ___main
.text:0040150E                  mov        dword ptr [esp+18h], 0Ch        12
.text:00401516                  mov        dword ptr [esp+1Ch], 0Fh        15
.text:0040151E                  mov        dword ptr [esp+20h], 0DDh       221
.text:00401526                  mov        dword ptr [esp+24h], 3          3
.text:0040152E                  mov        dword ptr [esp+28h], 1B0h       432
.text:00401536                  mov        dword ptr [esp+2Ch], 36h        54
.text:0040153E                  mov        dword ptr [esp+30h], 10h        16
.text:00401546                  mov        dword ptr [esp+34h], 43h        67
.text:0040154E                  mov        dword ptr [esp+3Ch], 0          0
.text:00401556                  mov        dword ptr [esp+38h], 0          0
.text:0040155E                  jmp       short loc_40157F
.text:00401560 ; ------------------------------------------------------
.text:00401560
.text:00401560 loc_401560:                                  ; CODE XREF: _main+84↓j
.text:00401560                  mov        eax, [esp+38h]      0
.text:00401564                  mov        eax, [esp+eax*4+18h]  -> increment.
.text:00401568                  cmp        eax, [esp+3Ch]      <
.text:0040156C                  jle      short loc_40157A
.text:0040156E                  mov        eax, [esp+38h]
.text:00401572                  mov        eax, [esp+eax*4+18h]
.text:00401576                  mov        [esp+3Ch], eax
.text:0040157A
.text:0040157A loc_40157A:                                   ; CODE XREF: _main+6C↑j
.text:0040157A                  add       dword ptr [esp+38h], 1
.text:0040157F
.text:0040157F loc_40157F:                                   ; CODE XREF: _main+5E↑j
.text:0040157F                  cmp        dword ptr [esp+38h], 7      0
.text:00401584                  jle      short loc_401560     jump if < than 7
.text:00401586                  mov        eax, [esp+3Ch]
.text:0040158A                  mov        [esp+4], eax       eax has largest number
.text:0040158E                  mov        dword ptr [esp], offset aD ; "%d"
.text:00401595                  call     _printf           print eax.
.text:0040159A                  mov        eax, 0
```

```
.text:0040159F                    leave
.text:004015A0                    retn
.text:004015A0 _main              endp
```



```
; Attributes: bp-based frame fuzzy-sp

; int __cdecl main(int argc, const char **argv, const char **envp)
public _main
_main proc near

argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF0h
sub     esp, 40h
call    ___main
mov     dword ptr [esp+18h], 0Ch
mov     dword ptr [esp+1Ch], 0Fh
mov     dword ptr [esp+20h], 0DDh
mov     dword ptr [esp+24h], 3
mov     dword ptr [esp+28h], 1B0h
mov     dword ptr [esp+2Ch], 36h
mov     dword ptr [esp+30h], 10h
mov     dword ptr [esp+34h], 43h
mov     dword ptr [esp+3Ch], 0
mov     dword ptr [esp+38h], 0
jmp     short loc_40157F
```

```
loc_40157F:
cmp     dword ptr [esp+38h], 7
jle     short loc_401560
```

```
loc_401560:
mov     eax, [esp+38h]
mov     eax, [esp+eax*4+18h]
cmp     eax, [esp+3Ch]
jle     short loc_40157A
```

```
mov     eax, [esp+3Ch]
mov     [esp+4], eax
mov     dword ptr [esp], offset aD ; "%d"
call    _printf
mov     eax, 0
leave
retn
_main endp
```

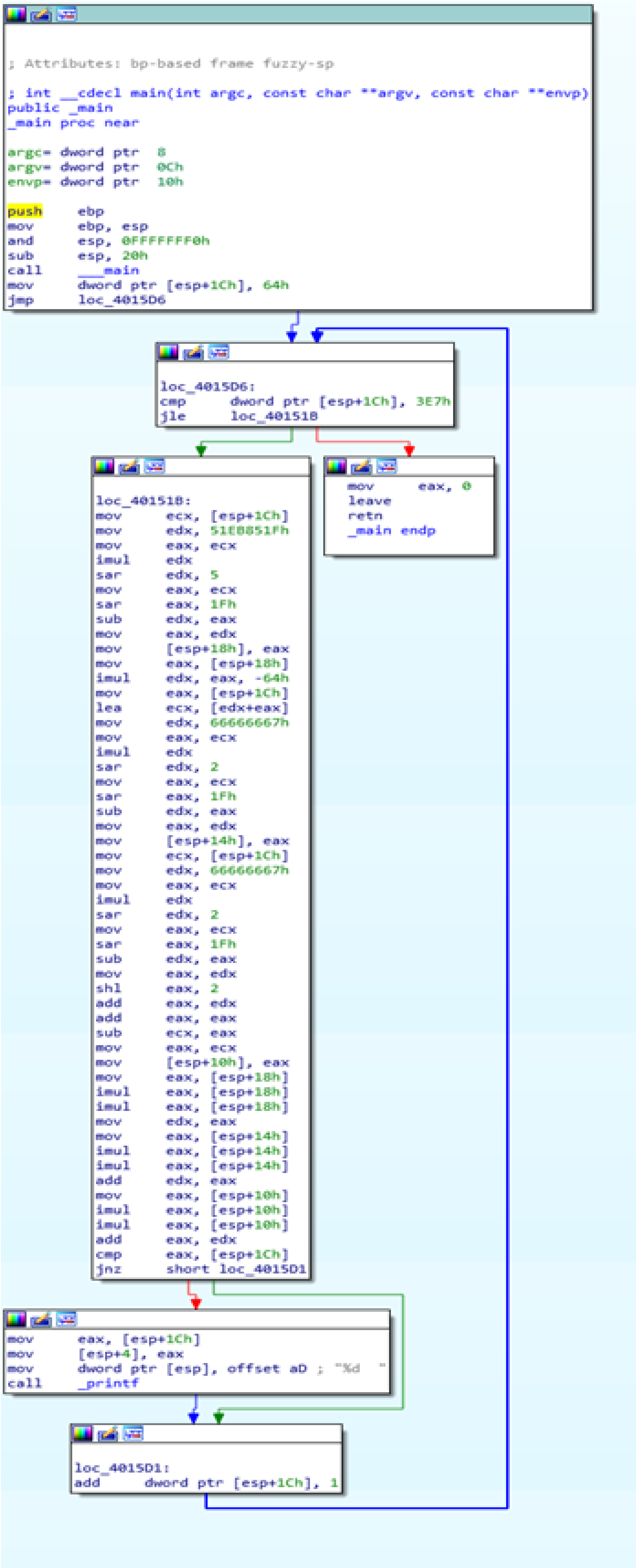
```
mov     eax, [esp+38h]
mov     eax, [esp+eax*4+18h]
mov     [esp+3Ch], eax
```

```
loc_40157A:
add     dword ptr [esp+38h], 1
```

Q3: Assembly code is shown below. Please access its functionality and rewrite in C to printf() its output. (15pts)

```
.text:00401500          push    ebp
.text:00401501          mov     ebp, esp
.text:00401503          and     esp, 0FFFFFFF0h
.text:00401506          sub     esp, 20h
.text:00401509          call    ___main                 100
.text:0040150E          mov     dword ptr [esp+1Ch], 64h
.text:00401516          jmp     loc_4015D6
.text:0040151B ; ---------------------------------------------------------------------------
.text:0040151B
.text:0040151B loc_40151B:                              ; CODE XREF: _main+DE↓j
.text:0040151B          mov     ecx, [esp+1Ch]
.text:0040151F          mov     edx, 51EB851Fh
.text:00401524          mov     eax, ecx
.text:00401526          imul    edx  edx * eax
.text:00401528          sar     edx, 5  right shift edx.by 5 → 51E
                                                           = 1310
.text:0040152B          mov     eax, ecx  eax = 1310       edx = 1310
.text:0040152D          sar     eax, 1Fh
.text:00401530          sub     edx, eax
.text:00401532          mov     eax, edx
.text:00401534          mov     [esp+18h], eax
.text:00401538          mov     eax, [esp+18h]
.text:0040153C          imul    edx, eax, -64h -100
.text:0040153F          mov     eax, [esp+1Ch]
.text:00401543          lea     ecx, [edx+eax] → eax stores the
                                                 mem location.
.text:00401546          mov     edx, 66666667h
.text:0040154B          mov     eax, ecx  eax = ecx
.text:0040154D          imul    edx
.text:0040154F          sar     edx, 2  right shif edx by 2.
.text:00401552          mov     eax, ecx
.text:00401554          sar     eax, 1Fh
.text:00401557          sub     edx, eax
.text:00401559          mov     eax, edx
.text:0040155B          mov     [esp+14h], eax
.text:0040155F          mov     ecx, [esp+1Ch]
.text:00401563          mov     edx, 66666667h
.text:00401568          mov     eax, ecx
.text:0040156A          imul    edx
.text:0040156C          sar     edx, 2
```

```
.text:0040156F                  mov       eax, ecx
.text:00401571                  sar       eax, 1Fh
.text:00401574                  sub       edx, eax
.text:00401576                  mov       eax, edx
.text:00401578                  shl       eax, 2
.text:0040157B                  add       eax, edx
.text:0040157D                  add       eax, eax
.text:0040157F                  sub       ecx, eax
.text:00401581                  mov       eax, ecx
.text:00401583                  mov       [esp+10h], eax
.text:00401587                  mov       eax, [esp+18h]
.text:0040158B                  imul      eax, [esp+18h]
.text:00401590                  imul      eax, [esp+18h]
.text:00401595                  mov       edx, eax
.text:00401597                  mov       eax, [esp+14h]
.text:0040159B                  imul      eax, [esp+14h]
.text:004015A0                  imul      eax, [esp+14h]
.text:004015A5                  add       edx, eax
.text:004015A7                  mov       eax, [esp+10h]
.text:004015AB                  imul      eax, [esp+10h]
.text:004015B0                  imul      eax, [esp+10h]
.text:004015B5                  add       eax, edx
.text:004015B7                  cmp       eax, [esp+1Ch]
.text:004015BB                  jnz       short loc_4015D1
.text:004015BD                  mov       eax, [esp+1Ch]
.text:004015C1                  mov       [esp+4], eax
.text:004015C5                  mov       dword ptr [esp], offset aD ; "%d    "
.text:004015CC                  call      _printf
.text:004015D1
.text:004015D1 loc_4015D1:                                   ; CODE XREF: _main+BB↑j
.text:004015D1                  add       dword ptr [esp+1Ch], 1
.text:004015D6
.text:004015D6 loc_4015D6:                                   ; CODE XREF: _main+16↑j
.text:004015D6                  cmp       dword ptr [esp+1Ch], 3E7h     goes from 100
.text:004015DE                  jle       loc_40151B               999             999
.text:004015E4                  mov       eax, 0
.text:004015E9                  leave
.text:004015EA                  retn
.text:004015EA _main            endp
```



```
; Attributes: bp-based frame fuzzy-sp

; int __cdecl main(int argc, const char **argv, const char **envp)
public _main
_main proc near

argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF0h
sub     esp, 20h
call    ___main
mov     dword ptr [esp+1Ch], 64h
jmp     loc_4015D6
```

```
loc_4015D6:
cmp     dword ptr [esp+1Ch], 3E7h
jle     loc_401518
```

```
loc_401518:
mov     ecx, [esp+1Ch]
mov     edx, 51EB851Fh
mov     eax, ecx
imul    edx
sar     edx, 5
mov     eax, ecx
sar     eax, 1Fh
sub     edx, eax
mov     eax, edx
mov     [esp+18h], eax
mov     eax, [esp+18h]
imul    edx, eax, -64h
mov     eax, [esp+1Ch]
lea     ecx, [edx+eax]
mov     edx, 66666667h
mov     eax, ecx
imul    edx
sar     edx, 2
mov     eax, ecx
sar     eax, 1Fh
sub     edx, eax
mov     eax, edx
mov     [esp+14h], eax
mov     ecx, [esp+1Ch]
mov     edx, 66666667h
mov     eax, ecx
imul    edx
sar     edx, 2
mov     eax, ecx
sar     eax, 1Fh
sub     edx, eax
mov     eax, edx
shl     eax, 2
add     eax, edx
add     eax, eax
sub     ecx, eax
mov     eax, ecx
mov     [esp+10h], eax
mov     eax, [esp+18h]
imul    eax, [esp+18h]
imul    eax, [esp+18h]
mov     edx, eax
mov     eax, [esp+14h]
imul    eax, [esp+14h]
imul    eax, [esp+14h]
add     edx, eax
mov     eax, [esp+10h]
imul    eax, [esp+10h]
imul    eax, [esp+10h]
add     eax, edx
cmp     eax, [esp+1Ch]
jnz     short loc_4015D1
```

```
mov     eax, 0
leave
retn
_main endp
```

```
mov     eax, [esp+1Ch]
mov     [esp+4], eax
mov     dword ptr [esp], offset aD ; "%d  "
call    _printf
```

```
loc_4015D1:
add     dword ptr [esp+1Ch], 1
```

Q4: Given a binary (i.e., HW2-P3-Q4), please use IDA to access its functionality and rewrite in C to printf() its output. (20pts)