המרכז לחינוך סייבר
CYBER EDUCATION CENTER

Assembly for Reverse Engineering

Workspace

Barak Gonen

```
call  2084
jmp   15CA
call  17B4
mov   si,7160
xor   di,di
mov   es,[7606]
mov   bx,000F
xor   cx,cx
mov   bp,0001
mov   dx,ds
```

Objectives

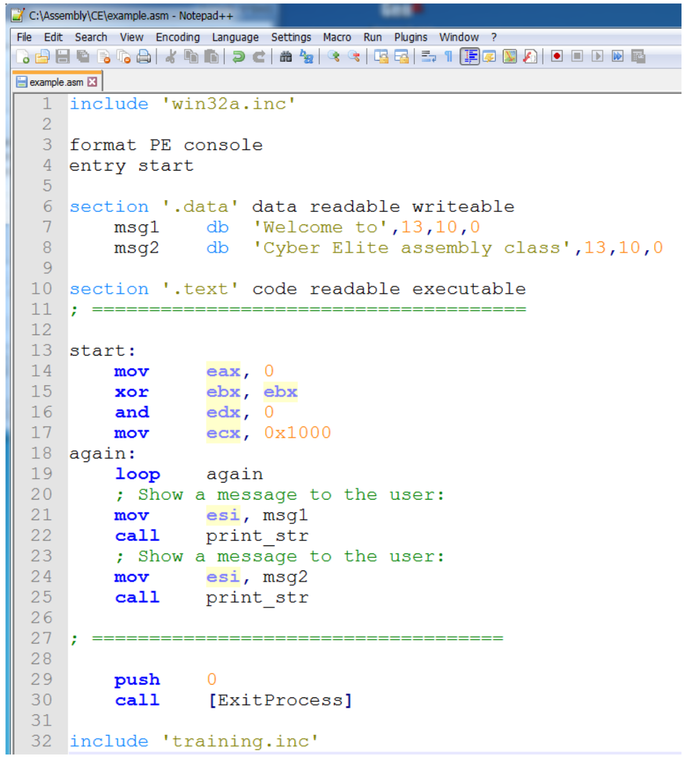
- Use the install guide to set workspace
- Be familiar with basic assembly file

Tools

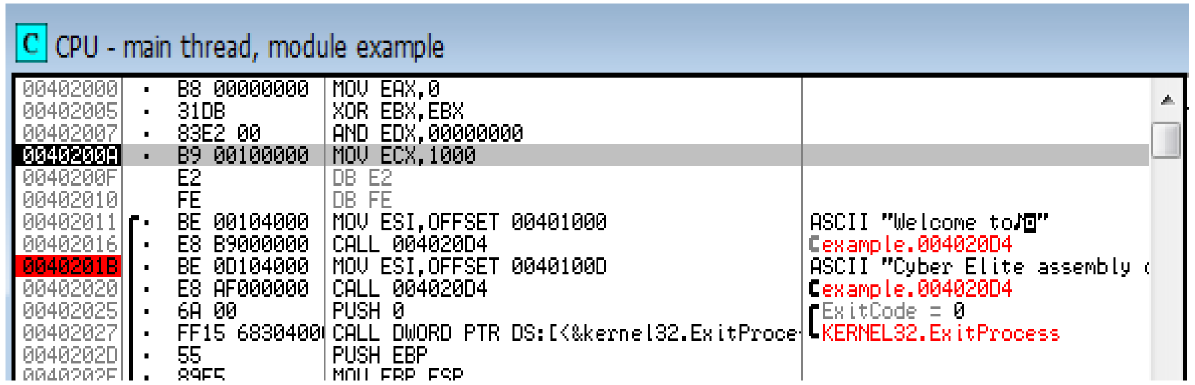
- Editor: write assembly files
  Notepad++
- Assembler : convert assembly files to machine language
  FASM
- Debugger: examine a program
  OllyDbg
- https://data.cyber.org.il/assembly/32bit/guide.pdf

Install Guide

- Use the Install Guide to convert example.asm to example.exe and disassemble it.


C:\Assembly\CE\example.asm - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
example.asm

```
include 'win32a.inc'

format PE console
entry start

section '.data' data readable writeable
     msg1     db  'Welcome to',13,10,0
     msg2     db  'Cyber Elite assembly class',13,10,0

section '.text' code readable executable
; ======================================

start:
     mov      eax, 0
     xor      ebx, ebx
     and      edx, 0
     mov      ecx, 0x1000
again:
     loop     again
     ; Show a message to the user:
     mov      esi, msg1
     call     print_str
     ; Show a message to the user:
     mov      esi, msg2
     call     print_str

; ====================================

     push     0
     call     [ExitProcess]

include 'training.inc'
```

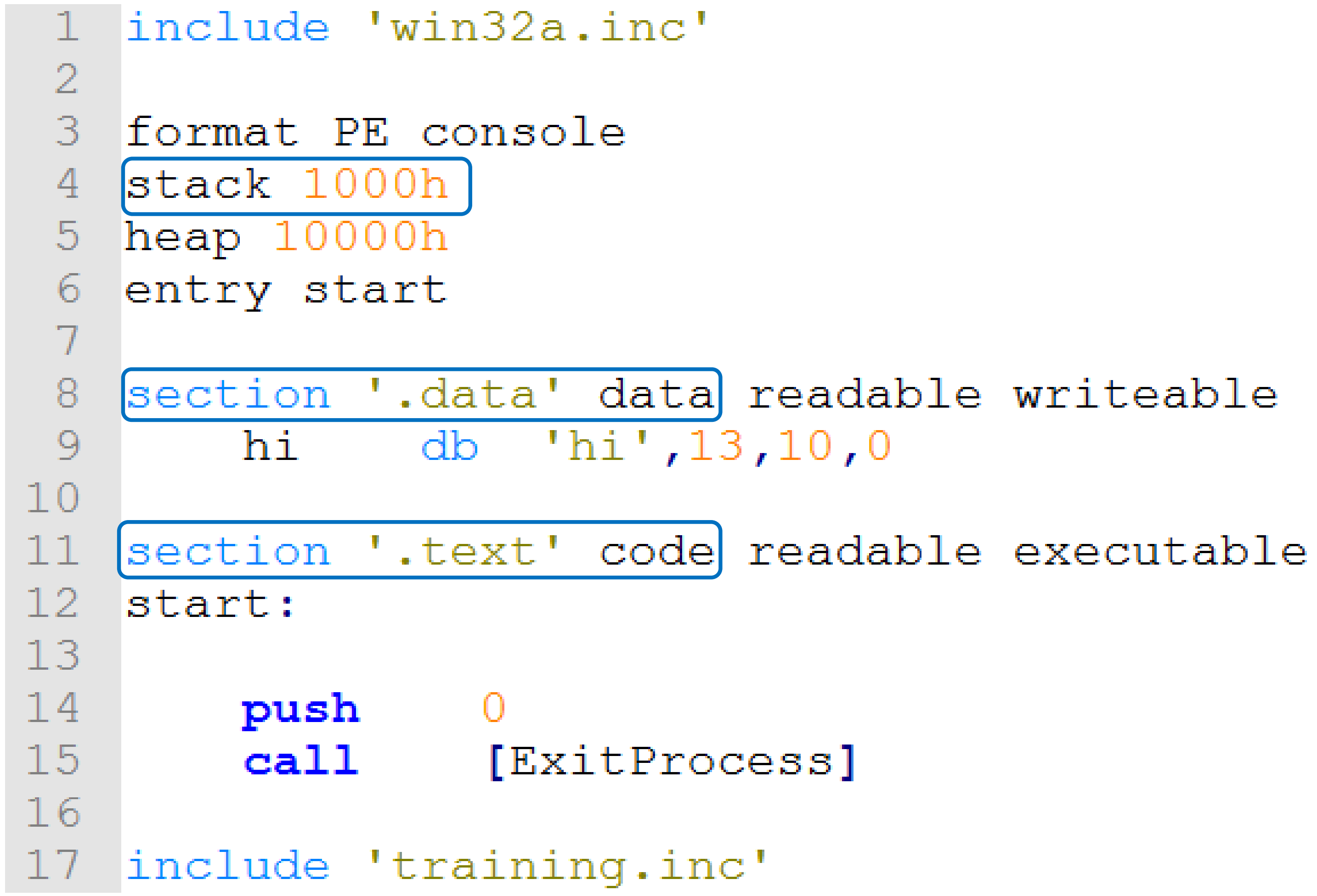


```
00402000  .  B8 00000000   MOV EAX,0
00402005  .  31DB          XOR EBX,EBX
00402007  .  83E2 00       AND EDX,00000000
0040200A  .  B9 00100000   MOV ECX,1000
0040200F     E2            DB E2
00402010     FE            DB FE
00402011 ┌.  BE 00104000   MOV ESI,OFFSET 00401000                ASCII "Welcome to♪◙"
00402016 │.  E8 B9000000   CALL 004020D4                          [example.004020D4
0040201B │.  BE 0D104000   MOV ESI,OFFSET 0040100D                ASCII "Cyber Elite assembly c
00402020 │.  E8 AF000000   CALL 004020D4                          [example.004020D4
00402025 │.  6A 00         PUSH 0                                 ┌ExitCode = 0
00402027 │.  FF15 6830400  CALL DWORD PTR DS:[<&kernel32.ExitProce └KERNEL32.ExitProcess
0040202D │.  55            PUSH EBP
0040202E │.  89E5          MOV EBP,ESP
```

Basic assembly file

- While executing, the Operating System reserves memory for the program
- Which parts of a program require memory?
  - Code – the machine language instructions
  - Data – variables (Heap is allocated here)
  - Stack – functions arguments and local variables

Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi      db   'hi',13,10,0

section '.text' code readable executable
start:

    push     0
    call     [ExitProcess]

include 'training.inc'
```

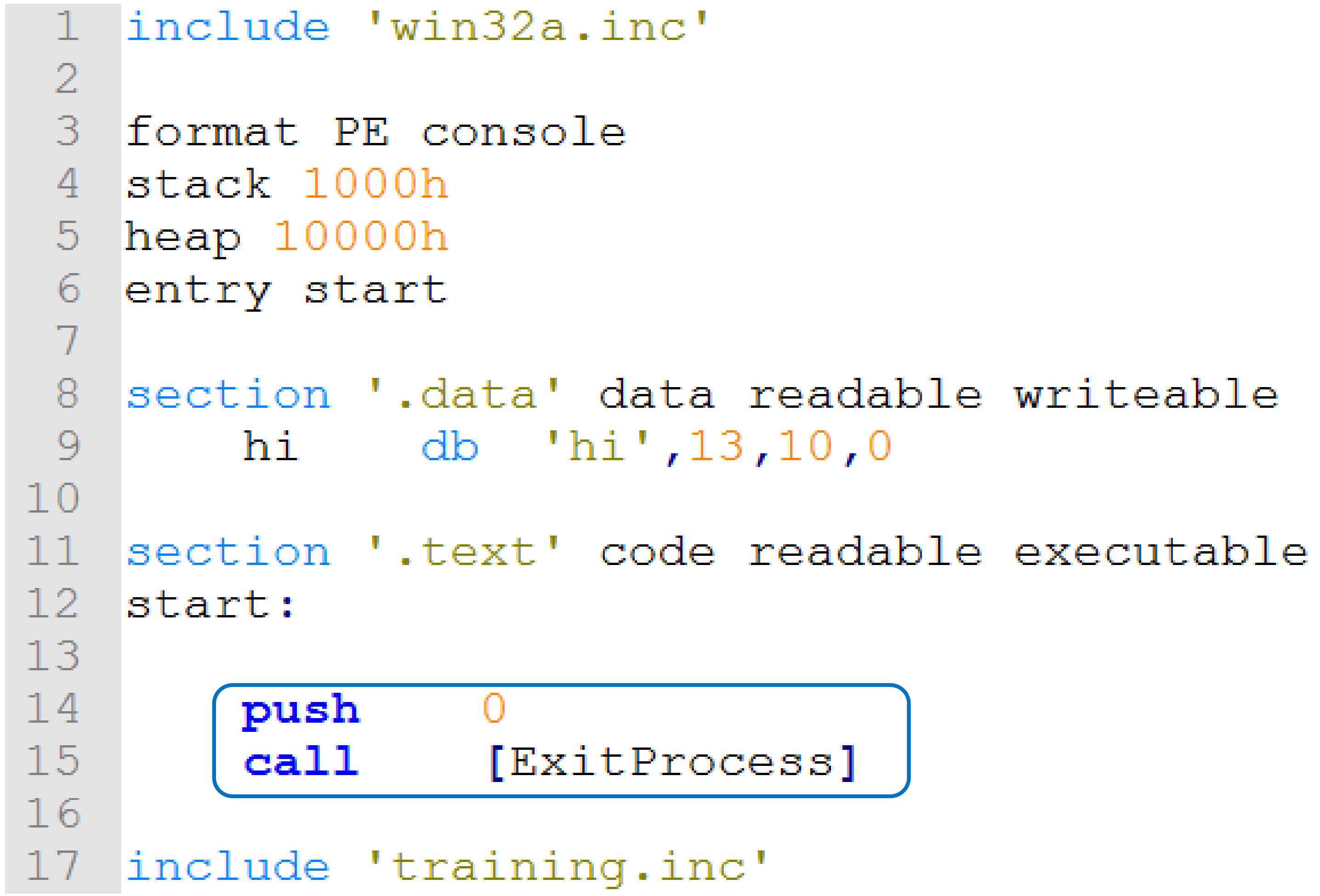
- The assembly file MUST define text section (code)
- data section – only if there are global variables
- Stack & Heap – OK if not defined, FASM will set defaults

Basic assembly file

```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi      db   'hi',13,10,0

section '.text' code readable executable
start:

    push    0
    call    [ExitProcess]

include 'training.inc'
```

- A variable is defined
- Variable name – “hi”

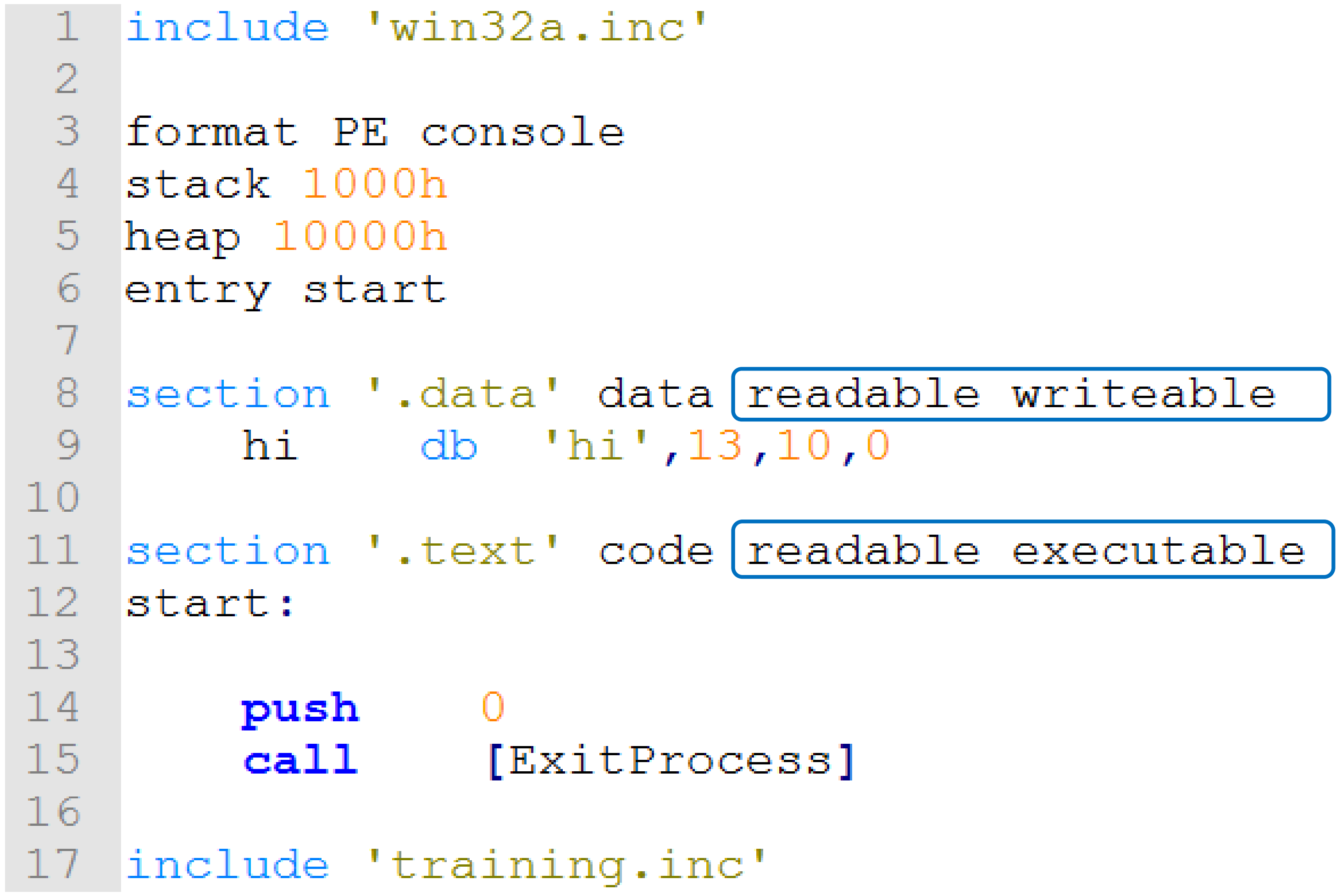
Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi     db  'hi',13,10,0

section '.text' code readable executable
start:

    push     0
    call     [ExitProcess]

include 'training.inc'
```

- Two assembly instructions

Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi      db  'hi',13,10,0

section '.text' code readable executable
start:

    push    0
    call    [ExitProcess]

include 'training.inc'
```

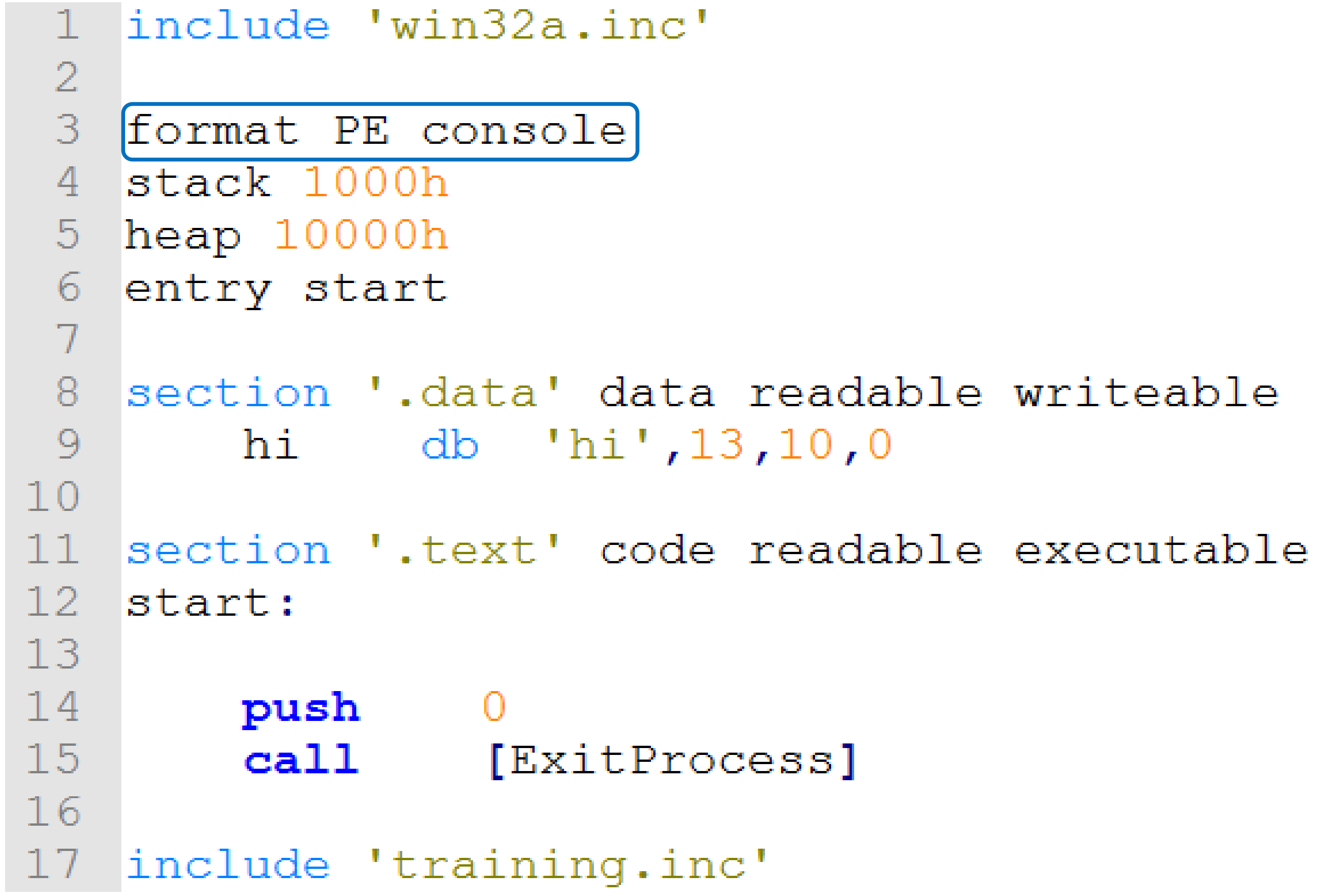
- OS should be told how to let the program access memory sections
- What if data memory wouldn't be writable? Will be executable?

Basic assembly file

```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi     db   'hi',13,10,0

section '.text' code readable executable
start:

    push     0
    call     [ExitProcess]

include 'training.inc'
```

- Parts covered so far

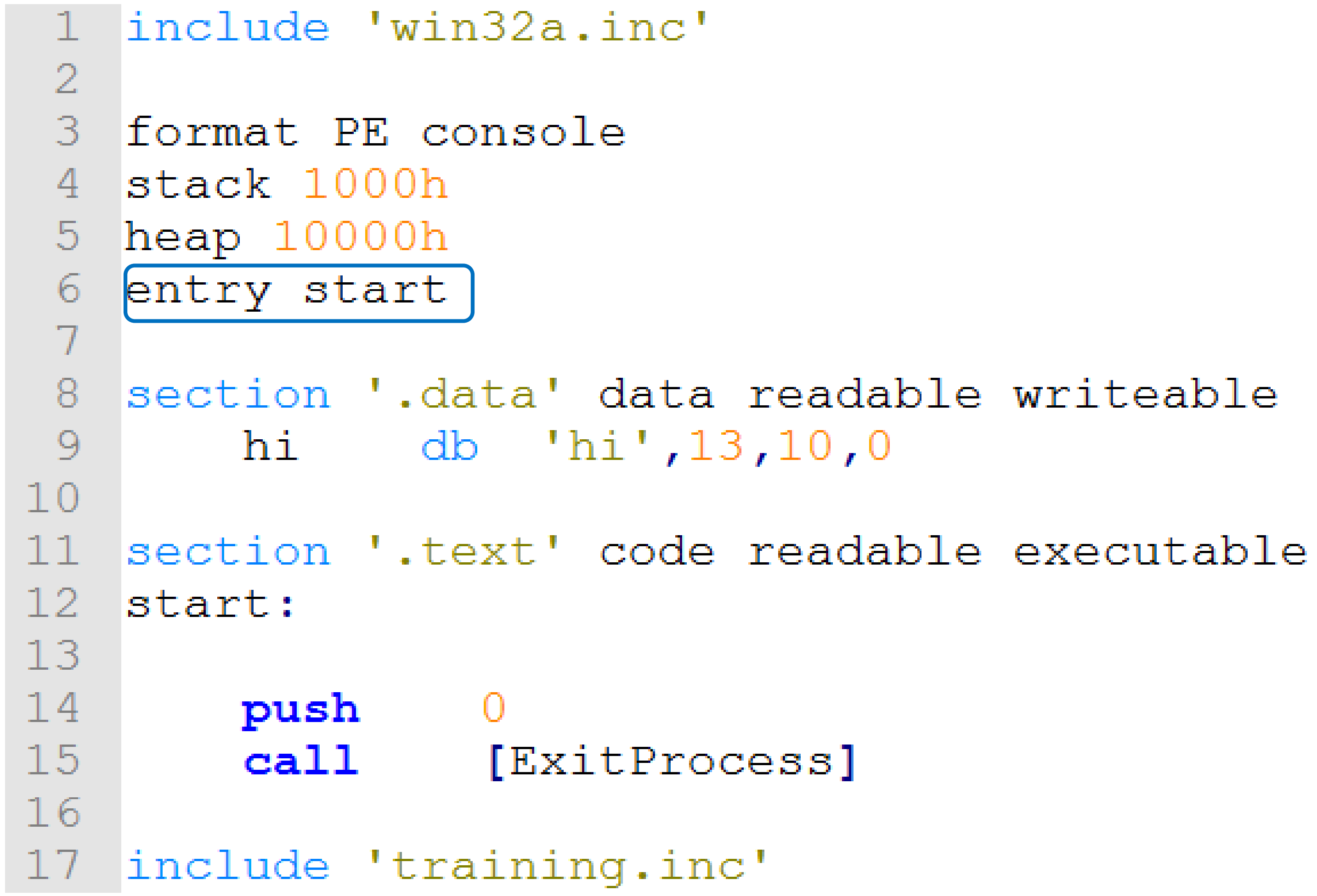
Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi     db  'hi',13,10,0

section '.text' code readable executable
start:

    push    0
    call    [ExitProcess]

include 'training.inc'
```

- PE – Portable Executable. Microsoft format.
- Console – use console (not GUI)

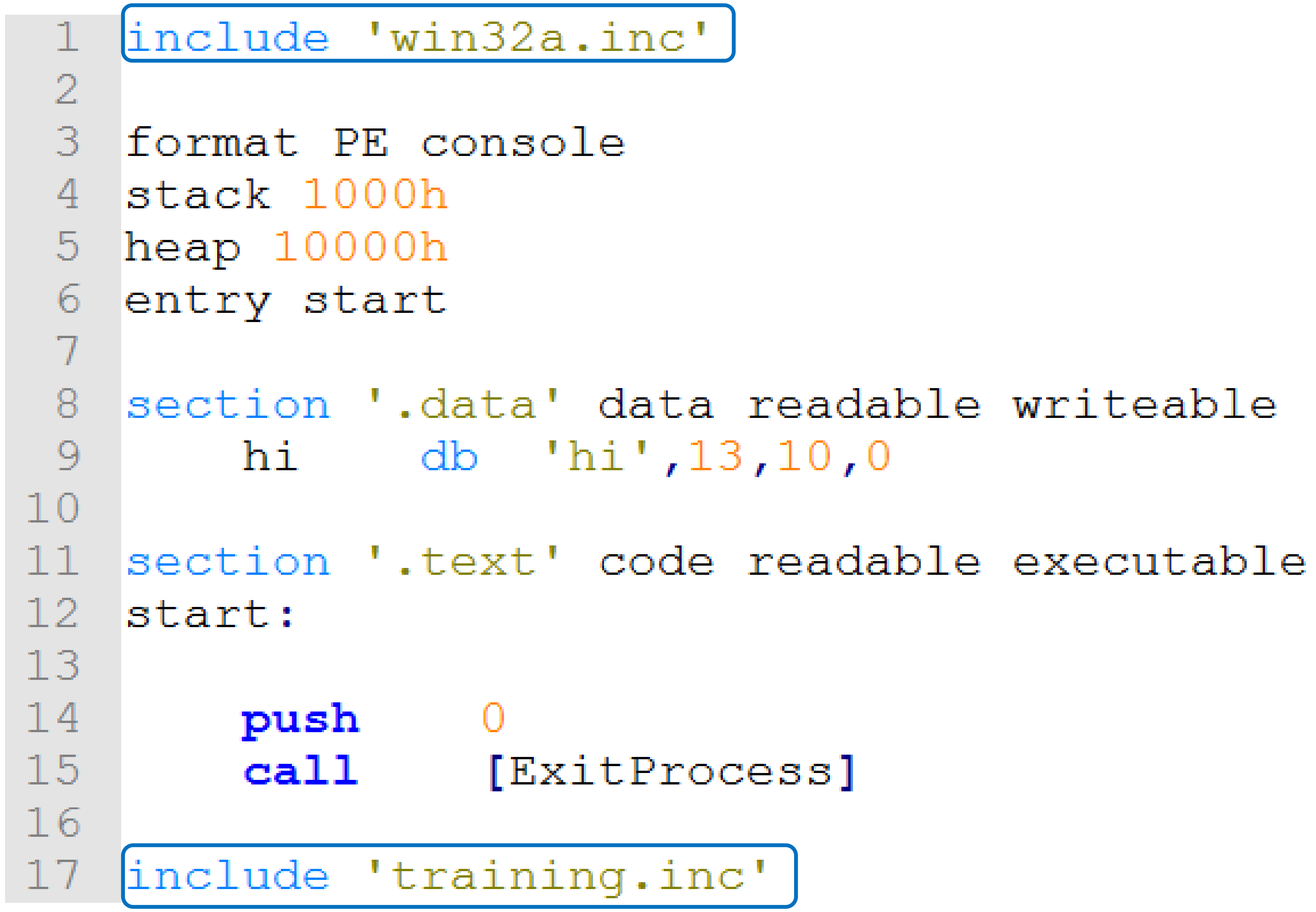
Basic assembly file



```
 1  include 'win32a.inc'
 2
 3  format PE console
 4  stack 1000h
 5  heap 10000h
 6  entry start
 7
 8  section '.data' data readable writeable
 9      hi     db   'hi',13,10,0
10
11  section '.text' code readable executable
12  start:
13
14      push    0
15      call    [ExitProcess]
16
17  include 'training.inc'
```

- Entry – where is the 1st instruction?
- Example: You may open a book on page 20
- We can move “start” from line 12 to another line, program execution will start there

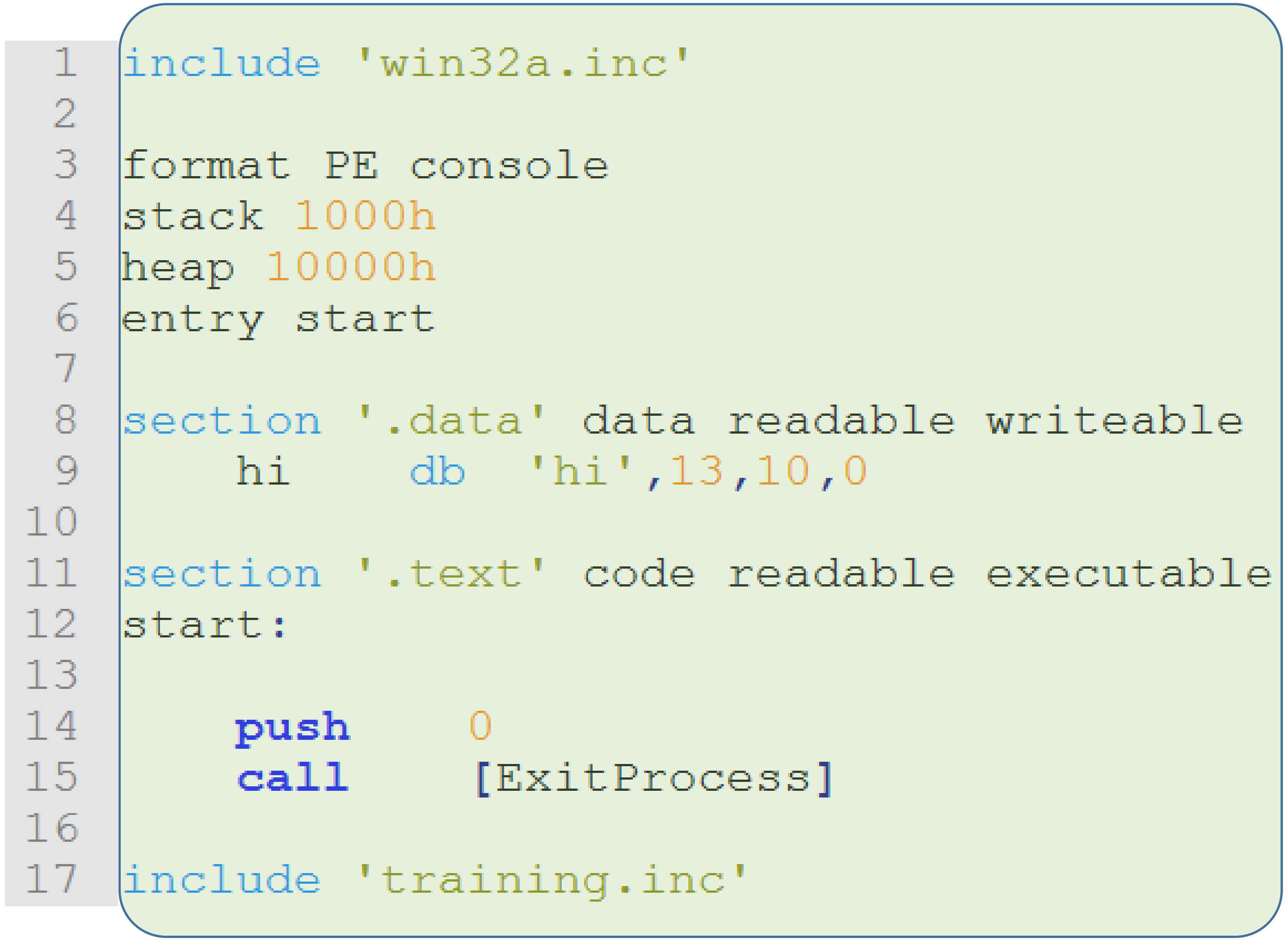
Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi     db   'hi',13,10,0

section '.text' code readable executable
start:

    push    0
    call    [ExitProcess]

include 'training.inc'
```

- Include win32a.inc, training.inc– helpful definitions and functions

Basic assembly file



```
include 'win32a.inc'

format PE console
stack 1000h
heap 10000h
entry start

section '.data' data readable writeable
    hi     db   'hi',13,10,0

section '.text' code readable executable
start:

    push    0
    call    [ExitProcess]

include 'training.inc'
```

- Parts covered so far – all ☺