Middle-term examination: System programs

Time: 60 minutes

(Write down to papers and submit a soft copy, i.e. pictures, at the end of the examination)

Part 1.

Given the following C function.

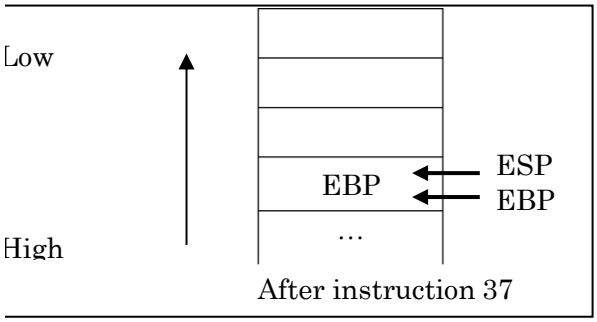
```
int f(int a, int b, int n) {
       if (n == 0) return a;
       else return f(a + b, b, n - 1);
}
```

1. Write assembly code to call f(1,2,3) in the following situations:
    a. The function f is written using cdecl calling convention
    b. The function f is written using pascal calling convention
    c. The function f is written using fastcall calling convention
2. Write the assembly code of the f function in the following situations.
    a. The function f is written using cdecl calling convention
    b. The function f is written using fastcall calling convention

Part 2.

A C program is compiled into assembly languages as shown in the figure below. Describe the stack state when the program is running at instructions with the line number (6), (11), (26), (42) (note. before instruction execution).

```
1  #include <stdio.h>
2
3  int a[5] = {1,2,3,4,5};
4
5  int count(int n, int x) {
6    int m;
7    if (n < 0) return 0;
8    else {
9      m = n - 1;
10     if (a[n] > x) return (1 + count(m, x));
11     else return count(m, x);
12   }
13 }
14
15 void main() {
16   printf("%d\n", count(5, 3));
17 }
18
```




```
1  .text
2  count:
3      pushl   %ebp
4      movl    %esp, %ebp
5      subl    $40, %esp
6      cmpl    $0, 8(%ebp)
7      jns .L2
8      movl    $0, %eax
9      jmp .L3
10 .L2:
11     movl    8(%ebp), %eax
12     subl    $1, %eax
13     movl    %eax, -12(%ebp)
14     movl    8(%ebp), %eax
15     movl    a(,%eax,4), %eax
16     cmpl    12(%ebp), %eax
17     jle .L4
18     movl    12(%ebp), %eax
19     movl    %eax, 4(%esp)
20     movl    -12(%ebp), %eax
21     movl    %eax, (%esp)
22     call    count
23     addl    $1, %eax
24     jmp .L3
25 .L4:
26     movl    12(%ebp), %eax
27     movl    %eax, 4(%esp)
28     movl    -12(%ebp), %eax
29     movl    %eax, (%esp)
30     call    count
31 .L3:
32     leave
33     ret
```

```
34 .text
35 main:
36     pushl   %ebp
37     movl    %esp, %ebp
38     andl    $-16, %esp
39     subl    $16, %esp
40     movl    $3, 4(%esp)
41     movl    $5, (%esp)
42     call    count
43     movl    $.LC0, %edx
44     movl    %eax, 4(%esp)
45     movl    %edx, (%esp)
46     call    printf
47     leave
48     ret
49 .data
50     .type   a, @object
51     .size   a, 20
52 a:
53     .long   1
54     .long   2
55     .long   3
56     .long   4
57     .long   5
58 .section   .rodata
59 .LC0:
60     .string "%d\n"
```

CORRECT:

C code (line 16): **printf("%d¥n", count(4,3));**

ASM code (line 41): **movl $5, (%esp)**

Part 3.

A linker links three object files (X, Y , Z) whose layout are described in the following table:

	X	Y	Z
Text	0x2047	0x1243	0x857
Data	0x3123	0x2016	0x209
BSS	0x6501	0x3994	0x463

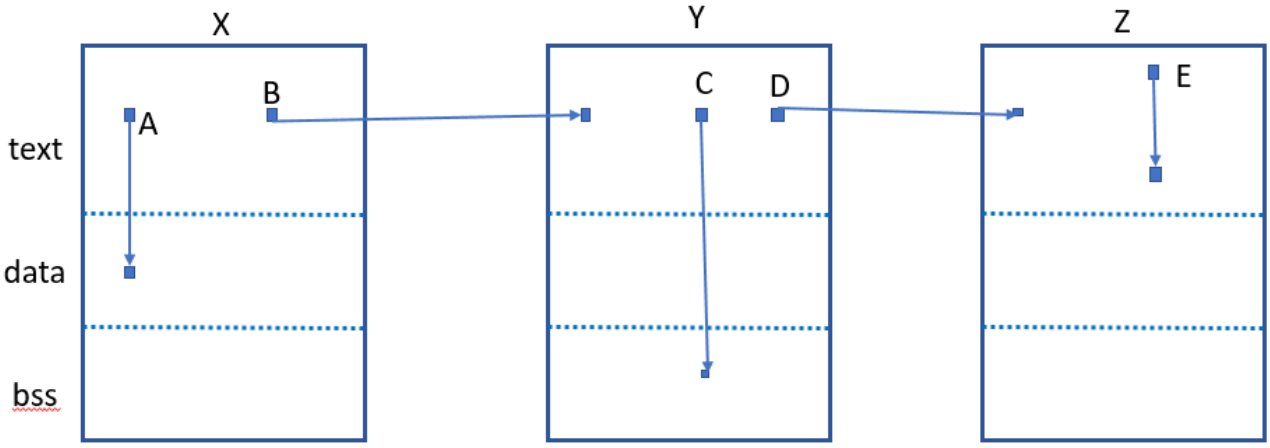
1. Identify base address for each segment by filling in the following table:

	X	Y	Z
Text			
Data			
BSS			

Note:

- Object code starts from page 1 (0x1000)
- Page size is 0x1000

2. There are several references in module X, Y and Z as follows:



- (A) is an absolute reference which has value 0x2610 (data base) before relocation.
- (B) is an absolute reference which has value 0x380 (text base) before relocation.
- (C) is an absolute reference which has value 0x3048 (data base) before relocation.
- (D) is an absolute reference which has value 0x280 (text base) before relocation.
- (E) is a relative reference which has value 0x288 (data base) before relocation.

Identify values of references (A), (B), (C), (D) và (E) after relocation.