Problem 1

```
int main() {
    int a = __3__;
    int result = 5;
    switch (a) {
    case 0:
        result++;
        break;
    case 1:
        result __*=_ 2;
        break;
    case 3:
        _result_ = 10;
    case 5:
        result += 5;
        break;
    case 6:
        __a__--;
        break;
    default:
        result = __7__;
        break;
    }
    return result;
}
```

```
<main>:
    push   %rbp
    mov    %rsp,%rbp
    movl   $0x3,-0x4(%rbp)
    movl   $0x5, -0x8(%rbp)
    cmpl   $0x6,-0x4(%rbp)
    _ja__   __.L6___
    mov    -0x4(%rbp),%eax
    mov    _.tbl (, %eax, 8)_,%rax
    jmpq   *%rax
.L1:
    addl   $0x1,-0x8(%rbp)
    jmp    __.L7__
.L2:
    shl    $1, -0x8(%rbp)
    jmp    .L7
.L3:
    movl   $10,-0x8(%rbp)
.L4:
    addl   $5,-0x8(%rbp)
    jmp    .L7
.L5:
    subl   $1,-0x4(%rbp)
    jmp    .L7
.L6:
    movl   $7,-0x8(%rbp)
.L7:
    mov    -0x8(%rbp),_%eax_
    pop    %rbp
    retq

.tbl:
    _____.L1_____
    _____.L2_____
    _____.L6_____
    _____.L3_____
    _____.L6_____
    _____.L4_____
    _____.L5_____
```

1. Please fill the blanks.
2. What's the return value of main()? 15

Problem 2

executed on a **64-bit little endian** machine.

```
#include <stdio.h>
int main(void){
  char array[6] = {0,0,1,4,6,7};int i;
  for(i = 0; i < 3; i++)
    foo(array + i);}
```

```
void foo(char* n){
  *(int*)n += 0x10100;
  char c = n[*n];
  printf("foo: %d\n", c);
}
```

```
0000000000400526 <foo>:
```

```
400526:   55                      push   %rbp
400527:   48 89 e5                mov    %rsp,%rbp
40052a:   48 83 ec 20             sub    $0x20,%rsp
40052e:   48 89 7d e8             mov    %rdi,-0x18(%rbp)
400532:   48 8b 45 e8             mov    -0x18(%rbp),%rax
400536:   8b 00                   mov    _(%rax)_,%eax
400538:   8d 90 00 01 01 00       lea    0x10100(%rax),%edx
40053e:   48 8b 45 e8             mov    -0x18(%rbp),%rax
400542:   89 10                   mov    %edx,(%rax)
400544:   48 8b 45 e8             mov    -0x18(%rbp),%rax
400548:   0f b6 00                movzbl (%rax),%eax
40054b:   48 0f be d0             movsbq %al,%rdx
40054f:   48 8b 45 e8             mov    -0x18(%rbp),%rax
400553:   48 01 d0                add    %rdx,%rax
400556:   0f b6 00                movzbl(%rax),%eax
400559:   88 45 ff                mov    %al,-0x1(%rbp)
40055c:   0f be 45 ff             movsbl -0x1(%rbp),%eax
400560:   89 c6                   mov    %eax,_ %esi
400562:   bf 54 06 40 00          mov    $0x400654,%edi

400567:   b8 00 00 00 00          mov    $0x0,%eax
40056c:   e8 8f fe ff ff          callq   400400   <printf@plt>
400571:   90                      nop
400572:   c9                      leaveq //restore %rbp and %rsp
400573:   c3                      retq
0000000000400574 <main>:
400574:   55                      push   %rbp
400575:   48 89 e5                mov    %rsp,%rbp
400578:   48 83 ec 10             sub    $0x10,%rsp
40057c:   c6 45 f0 00             movb   $0x0,-0x10(%rbp)
400580:   c6 45 f1 00             movb   $0x0,-0xf(%rbp)
400584:   c6 45 f2 01             movb   $0x1,-0xe(%rbp)
400588:   c6 45 f3 04             movb   $0x4,-0xd(%rbp)
40058c:   c6 45 f4 06             movb   $0x6,-0xc(%rbp)
400590:   c6 45 f5 07             movb   $0x7,-0xb(%rbp)
400594:   c7 45 fc 00 00 00 00    movl   $0x0,-0x4(%rbp)
40059b:   eb 18                   jmp    4005b5 <main+0x41>
40059d:   8b 45 fc                mov    -0x4(%rbp),%eax
```

```
4005a0:   48 98                    cltq
4005a2:   48 8d 55 f0              lea    _-0x10(%rbp),%rdx
4005a6:   48 01 d0                 add    %rdx,%rax
4005a9:   48 89 c7                 mov    %rax,%rdi
4005ac:   e8 75 ff ff ff           callq  400526 <foo>
4005b1:   83 45 fc 01              addl   $0x1,-0x4(%rbp)
4005b5:   83 7d fc 02              cmpl   $0x2,-0x4(%rbp)
4005b9:   7e e2                    jle    40059d <main+0x29>
4005bb:   b8 00 00 00 00           mov    $0x0,%eax
4005c0:   c9                       leaveq
4005c1:   c3                       retq
```

Suppose **BEFORE** the execution of instruction at **400574 (push %rbp)**, the register values are: **%rsp = 0x7fffffffdb58 %rbp = 0x7fffffffdb58**

1. Fill in the blanks in the Assembly Code.

2. According to the %rsp, %rbp **BEFORE** the execution of instruction at **400574 (push %rbp)**. Please show the value of %rsp and %rbp under the following conditions.

AFTER executing the instruction **"push %rbp" (400574)**

%rsp = 0x7fffffffdb50 %rbp = 0x7fffffffdb58

AFTER executing the instruction **"call <foo>" (4005ac)**

%rsp = 0x7fffffffdb38 %rbp = 0x0x7fffffffdb50

BEFORE executing the instruction **"leave" (400572)**

%rsp = 0x7fffffffdb10 %rbp = 0x0x7fffffffdb30

AFTER executing the instruction **"ret" (400573)**

%rsp = 0x7fffffffdb40 %rbp = 0x0x7fffffffdb50

3. After that, we restart the execution. Now, we stop **After** the execution of instruction at 400536: mov ___, %eax

We find that the value of %eax is 0x4010000.Then we continue the execution. Please fill the table below.

NOTE: "After **400536**" means "after executing the instruction in the address **400536**".

Phase	Register or Address	Value	Meaning
After 400538	**%edx**	0x4020100	*(int*)n+0x10100
After 400548	**%eax**	0	n[0]
After 400559	**-0x1(%rbp)**	0	c

4. Please write the output of the program.

foo: 0

foo: 3

foo: 7

Problem 3

Assume we have a function f:

```
void f() {
    int arr[10][_5__];
    for (int i = 0; i < 10; i++) {
        arr[i][0] = _i+1__;
        *(&arr[0][i] + 5) = i;
    }
}
```

The assembly code is:

```
<f>:
        push   %rbp
        mov    %rsp,%rbp
        sub    $0x58,%rsp
        movl   $0,-4(%rbp)
        jmp    .L0
.L1:    mov    -4(%rbp),%eax
        lea    1(%rax),%ecx
        movslq %eax,%rdx
        mov    %rdx,%rax
        shl    $2,%rax
        add    %rdx,%rax
        lea    -0xd0(%rbp,%rax,4),%rax
        mov    %ecx,(%rax)
        lea    -0xd0(%rbp),%rax
        mov    -4(%rbp),%edx
        movslq %edx,%rdx
```

```
        shl     $2,%rdx
        add     %rdx,%rax
        lea     __20_(%rax),%rdx
        mov         -4(%rbp),%eax
        mov         %eax,(%rdx)
        addl    $1,-4(%rbp)
.L0     cmpl    $9,-4(%rbp)
        jle     .L1
        leaveq
        retq
```

1. Please fill the blanks
2. Please add label .L0 to right position
3. Please give a statement that can replace "*(&arr[0][i] + 5) = i;"
   arr[_i/5 + 1_][__i%5__] = i;
4. Which elements in arr will be touched by "*(&arr[0][i] + 5) = i;"
   arr[1][0]….arr[1][4]arr[2][0]…arr[2][4]