1. Addressing Mode

Assuming we have the following initial state of registers and memory:

Memory Address	Value	Register	Value
0x10	0x20	%rax	0x10
0x20	0x25	%rbx	0x24
0x21	0x10	%rcx	0x1
0x22	0x26	%rdx	0x20
0x23	0x21		
0x24	0x24		

Please fill in the table(in **hexadecimal**)(Memory access will load one byte)

Operand	Value
%rax	0x10
(%rax)	0x20
0x21	0x10
$0x21	0x21
0x20(%rcx)	0x10
0x20(,%rcx,2)	0x26
(%rdx, %rcx)	0x10
0x10(%rax, %rcx, 4)	0x24

2. Data moving

Assume we have the following initial state of memory and registers, and we run on an x86-64 machine.

Memory Address	Value	Register	Value
0x20	0xDE	%r8	0xF0F1F2F3
0x21	0xAD	%r9	0x22
0x22	0xBE	%r10	0x1
0x23	0xEF	%r11	0x11223344556677
0x24	0x12	%rcx	0x0
0x25	0x34		
0x26	0x56		

0x27	0x78		

We execute the following assembly codes.

```
movl    (%r9, %r10, 2), %ecx
movw    $-1, %r11w
movl    $-1, %r11d
movabsq $-1, %r11
movq    $-1, %r11
movsbl  %r8b, %ecx
movzbl  %r8b, %ecx
```

Please fill the table which shows the state **after** each instruction being executed (In **hexadecimal**) (Please write "——" if the register or memory will not change):

Instructions	%rcx	%r11
movl	0x12	—
movw	—	0x1122334455FFFF
movl	—	0x112233FFFFFFFF
movabsq	—	0xFFFFFFFFFFFFFFFF
movq	—	0xFFFFFFFFFFFFFFFF
movsbl	0xFFFFFFF3	—
movzbl	0xF3	—