I. Shift and Type Casting

1. Calculate the C expressions:

Assume we are using **64-bit machine**, and variables have following values.

```
char c = -9;
unsigned char uc = c;
short s = uc * 3;
unsigned short us = s;
```

Fill in the table below.

Expression	Value	Hexadecimal
c	-9	F7
uc	247	F7
s		
us		
s >> 2		
us >> 2		

2. Compare the following values (Assume the value is in 16-bit)

A	Relation	B
-1	>	-2
0		-1
0U		-1
-32768		32768U
-1		65535U
-3U		-4U

II.x86_64 Instructions

1. Addressing Mode

Assume we have following initial states 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 only one byte):

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

2. Stack and data moving

Assume we have following initial states of registers and memory, and the machine is in **64-bit** and **little-endian**:

Memory Address	Value	Register	Value
0x10	0x20	%rax	0x10
0x20	0xFE	%rbx	0xFFFFFFFFFFFFFFFF
0x21	0xFF	%rcx	0x1
0x22	0x01	%rsp	0x20
0x23	0xFE		
0x24	0x02		
0x25	0xFD		
0x26	0x03		
0x27	0xFC		

And the CPU executes following assembly code:

```
pop %rcx
mov %ecx, %ebx
push %rbx
movsbl %bl, %ecx
movzbl %bl, %ecx
push %rcx
movsbl %bh, %ecx
push %rcx
movzbl %bh, %ecx
push %rcx
movb (%rax), %bl
```

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

Instruction	%rbx	%rcx	%rsp	0x10	0x20
pop	——			——	——
mov					
push					
movsbl					

movzbl					
push					
movsbl					
push					
movzbl					
push					
movb					