1. Given a number, A, spans over registers %l1 to %l3 and a number, B, spans over registers %o1 to %o3. Write an assembly code segment to perform B = A + B (10 points.)

[A = %l1%l2%l3, B = %o1%o2%o3]

Explain in few lines the reason of using different addition instructions.

Answer:

```
addcc %l3, %o3, %o3          ! set the carry flag
addxcc %l2, %o2, %o2          ! set the carry flag and add the carry status
addx %l1, %o1, %o1           ! add the carry in evaluation
```
Assume you have only 8-bit registers available.

2. What is the content of register %l1 (in hexadecimal) at the end of the following instructions? (10 points)

a. mov 0x47, %l0
   orn %l0, 0xca, %l1

Answer:
   mov 0x47, %l0  !%l0 = 0100,0111
   orn %l0, 0xca, %l1  !0xca = 1100,1010  not 0xca = 0011,0101
                       !%l1 = 0111,0111

b. mov 0x55, %l1
   btog 127, %l1

Answer:
   mov 0x55, %l1  !%l1 = 0101,0101
   btog 127, %l1  !127 = 0111,1111
                      !%l1 = 0010,1010
3. Show the content of register ar in hexadecimal at the end of three labeled instructions, assuming ar is a 16 bit register. (10 points)

```plaintext
mov 0xff, %ar  ! (The initial value of ar is 0xff)
1:
bclr 0xaa, %ar
   andn %ar, 0x10, %ar
2:     
orn  %ar, %g0, %ar
   xnor %ar, %g0, %ar
3:         
bset 0x11, %ar
```

Answer:

1. %ar = 0000,0000,0101,0101 = 0x0055
2. %ar = 0000,0000,0100,0101 = 0x0045
3. %ar = 1111,1111,1011,1011 = 0xffbb
4. While converting a C program segment (given below) into an assembly program, I unknowingly introduced a bug. Help me find it and get a prize worth 10 points.

```c
while (a<=1)
{
    a = a + b;
}
```

Here is the buggy program. Rewrite the program to make it correct.

```assembly
test:
    cmp     %a_r, 1
    bg      1f

2:
    add     %a_r, %b_r, %a_r
    cmp     %a_r, 1
    ble     2b
    nop

1:
```

Answer:

```assembly
test:
    cmp     %a_r, 1
    bg      1f
    nop

2:
    add     %a_r, %b_r, %a_r
    cmp     %a_r, 1
    ble     2b
    nop

1:
```

or better answer:

```assembly
ba     test
nop

1:
    add     %a_r, %b_r, %a_r
```

```assembly
test:
    cmp     %a_r, 1
    ble     1b
    nop
```