Problem1:

Write MIPS assembly program to check a number in $s0 is even or odd using AND instruction. If the number is even copy it into $s2, if it is odd copy it in $s3.

Solution

```
.globl main
.text
main:
  li $s0, 0xF3
  andi $t0, $s0,1
  beq $t0, $zero, even
  move $s3, $s0
  j exit

even:
 move $s2, $s0

exit:
 li $v0, 10
 syscall
```

Problem 2:

The table below holds some logical operations that are not included in the MIPS instruction set. How can these instructions be implemented using the available MIPS instructions that you know?

a.	ANDN $s1, $s2, $s3 // bit-wise AND of $s2 and !s3
b.	XNOR $s1, $s2, $s3 // bit-wise exclusive-NOR

Solution

```
.globl main
.text
main:
  li $s2, 0xA3
  li $s3, 0xB2
  not $t1, $s3
 and $s1, $s2, $t1
  li $v0, 10
  syscall
```

```
.globl main
.text
main:
  li $s2, 0xA3
  li $s3, 0xB2
  xor $t1, $s2, $s3
  not $s1, $t1
  li $v0, 10
  syscall
```

Problem 3:

Write MIPS assembly program that given the value 0xBD in register $s1, it replaces the bits from bit 4 to bit 7 to be 0x5 instead of 0xB.

Note that in binary to replace bits by another value this is done in two steps:

1) Masking: this clears the unrequired bits (using AND operation)
2) Inserting: this inserts the required bits (using OR operation)

Solution

```
.globl main
.text
main:
  li $s1,0xBD
  andi $s1,0x0F
  ori $s1,0x50

  li $v0,10
  syscall
```

Problem 4:

Write MIPS assembly program that performs the following equation a = (b+ 4c) without using the **mul** instruction. Assume that a, b, and c are values in $s0, $s1, and $s2 respectively.

Solution

```
.globl main
.text
main:
  li $s1, 3
  li $s2, 5
  sll $t1, $s2, 2
  add $s0, $t1, $s1

  li $v0,10
  syscall
```

Problem 5:

Write MIPS assembly program that performs the following equation A = B – C/8 without using **div** instruction. Assume that A, B and C are values in $s0, $s1, and $s2 respectively.

Solution

```
.globl main
.text
main:
  li $s1, 5
  li $s2, 16
  sra $t1, $s2, 3
  sub $s0, $s1, $t1

  li $v0,10
  syscall
```

Problem 6:

Write MIPS assembly program that performs the following equation A = ($B+ Cx2^D$) without using **mul** instruction. Assume that A, B, C and D are values in $s0, $s1, $s2 and $s3 respectively.

Solution:

```
.globl main
.text
main:
 li $s1, 5
 li $s2, 4
 li $s3, 3
 sll $t1, $s2, $s3
 add $s0, $s1, $t1

 li $v0,10
 syscall
```

Problem 7:

Write an equivalent MIPS assembly program to the following high-level programming code.

```
i=1
j =1
sum =0
for (int i=1; i<=5; i++)
{
   for (int j =1; j<=20; j++)
   {
          sum = sum + (i*j)
  }
}
```

Solution:

```
.globl main
.text
main:
  li $s1, 1
  li $s2, 1
  li $s3, 0

outer:
  slti $t1, $s1, 6
  beq  $t1, $zero, exitOuter

inner:
  slti $t2, $s2, 21
  beq $t2, $zero, exitInner
  mul $t3, $s1, $s2
  add $s3, $s3, $t3
  addi $s2,1
  j inner

exitInner:
 addi $s1, 1
 j outer

exitOuter:
  li $v0,10
  syscall
```

Problem 8:

Write an equivalent MIPS assembly program to the following pseudocode.

```
i=1
j =1
sum =0
for (int i=1; i<=5; i++)
{
   for (int j =1; j<=20; j++)
   {
          sum = sum + (the anding between the 1st complement of i with the value of j)
  }
}
```

Solution:

```
.globl main
.text
main:
  li $s1, 1
  li $s2, 1
  li $s3, 0

outer:
  slti $t1, $s1, 6
  beq  $t1, $zero, exitOuter
inner:
  slti $t2, $s2, 21
  beq $t2, $zero, exitInner
  not $t3, $s1
  and $t2, $t3, $s2
  add $s3, $s3, $t2
  addi $s2,1
  j inner
```

```
exitInner:
 addi $s1, 1
 j outer

exitOuter:
  li $v0,10
  syscall
```