Problem 1:

Write an ARM7 assembly program to find the greatest common divisor (GCD) of 2 integers, where the GCD is the largest positive integer that divides the numbers without a remainder. For example, the GCD of 8 and 12 is 4, the GCD of 54 and24 is 6. The below high level programming code is a simple way to find the greatest common divisor (GCD)of **a** and **b**.

```
int a = 54;
int b =24;
while(a != b)
{
        if(a>b)
          a=a-b;
        else
            b=b-a;
        }
```

Solution

```
        AREA GCD, CODE
        ENTRY
        MOV R0, #54
        MOV R1, #24
Loop    CMP R0, R1
        SUBGT R0, R0, R1
        SUBLT R1, R1, R0
        BNE Loop
        END
```

Problem2:

Write an ARM7 assembly program that calculates the value of 2^3 given the high level programming language code below.

```
int a =1;
for (int i=0; i<3; i++)
{
        a*=2;
}
```

Solution

```
        AREA Power, CODE
        ENTRY
        MOV R0, #1
        MOV R1, #0
        MOV R2, #2          ; Multiplication Constant
Loop    MUL R3, R0, R2
        MOV R0, R3
        ADD R1, R1, #1
        CMP R1, #3
        BLT Loop
        END
```

Problem3:

Write an ARM7 assembly program that sums up the numbers from 1to 10 starting from number 10, given the high level programming language code below.

```
int sum =0;
for (int i=10; i>0; i--)
{
        sum +=i;
}
```

Solution

```
        AREA adding, CODE
        ENTRY
        MOV R0, #0
        MOV R1, #10
Loop    ADD R0, R0, R1
        SUB R1, R1, #1
        CMP R1, #0
        BGT loop
        END
```

Problem4:

Write an ARM7 assembly program to check a number in R0 is even or odd using AND instruction. If the number is even copy it into R2, if it is odd copy it in R3.

Solution

```
        AREA check, CODE
        MOV R0, #0xF3
        ANDS R1, R0, #0x01
        BEQ even
        MOV R3, R0
        B EXT
even    MOV R2, R0
EXT
        END
```

Problem 5:

The table below holds some logical operations that are not included in the ARM7 instruction set. How can these instructions be implemented using the available ARM7 instructions?

a.	ANDN R1, R2, R3 // bit-wise AND of R2 and !R3
b.	XNOR R1, R2, R3 // bit-wise exclusive-NOR

Solution

```
; ANDN  R1, R2, R3
AREA ANDN, Code
ENTRY
 MOV R2, #0xAA
 MOV R3, #0xF3
 MVN R4, R3
 AND R1, R2, R4
 END
```

```
; XNOR   R1, R2, R3
 AREA XNOR, Code
 ENTRY
 MOV R2, #0xAA
 MOV R3, #0xF3
 EOR R4, R2, R3
 MVN R1, R4
 END
```

Problem 6:

Write an ARM7 assembly program that given the value 0xBD in register R1, 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

```
        AREA insert, CODE
        ENTRY
        MOV R1, #0xBD
        AND R1, #0x0F
        ORR R1, #0x50
        END
```

Problem 7:

Write an ARM7 assembly program that perform the following equation a = (b+ 4c) without using the MUL instruction. Assume that a, b and c are values in R0, R1, and R2 respectively.

Solution

```
AREA equation, CODE
ENTRY
MOV R1,#3
MOV R2,#5
ADD R0, R1, R2,LSL#2
END
```

Problem 8:

Write an ARM7 assembly program that perform the following equation A = B – C/8 without looping. Assume that A, B and C are values in R0, R1, and R2 respectively.

Solution

```
AREA equation, CODE
ENTRY
MOV R1,#5
MOV R2,#16
 SUB R0, R1, R2, ASR#3
END
```

Problem 9:

Write an ARM7 assembly program that perform the following equation A = (B+ $Cx2^D$) without looping or using the MUL instruction. Assume that A, B, C and D are values in R0, R1, R2 and R3 respectively.

Solution:

```
        AREA equation, CODE
        ENTRY
        MOV R1, #5
        MOV R2, #4
        MOV R3, #3
        ADD R0, R1, R2, LSL R3
        END
```

Problem 10:

If we set R0 to (0x00000022) and Rotate Right by 2, what will be the Result in hexadecimal?

Solution:

It will be (0x80000008)