1.WRITE AN ASSEMBLY LANGUAGE PROGRAM TO ADD 2 NUMBERS

```
AREA MYCODE,CODE,READONLY
START
          LDR R0,= 0X112
          LDR R1,= 0X341
          ADD R2,R1,R0    ;R2=R1+R0 =  0X453
HERE      B   HERE
          END
```

2 .WRITE AN ASSEMBLY LANGUAGE PROGRAM TO SUBSTRACT 2 NUMBERS

```
AREA MYCODE,CODE,READONLY
START
          LDR R0,= 0X112
          LDR R1,= 0X341
          SUB R2,R1,R0
HERE       B  HERE
          END
```

3.. WRITE AN ASSEMBLY LANGUAGE PROGRAM TO MULTIPLY 2 NUMBERS

```
      AREA MYCODE,CODE,READONLY
START
          LDR R0,= 0X112
          LDR R1,= 0X341
          MUL R2,R1,R0
HERE      B  HERE
          END
```

4. WRITE AN ASSEMBLY LANGUAGE PROGRAM TO REVERSE SUBSTRACT 2 NUMBERS

```
          AREA MYCODE,CODE,READONLY
START
           LDR R0,= 0X112
```

```
      LDR R1,= 0X341
       SUB R2,R1,R0
HERE    B  HERE
       END
```

5 ANOTHER WAY OF DOING MULTIPLICATION

```
      AREA CONSTANT,DATA,READONLY
NUM1 EQU 0X0040
NUM2 EQU 0X0010
     AREA MYCODE,CODE,READONLY
ENTRY
     LDR R0,=NUM1
     LDR R1,=NUM2
      MUL R2,R1,R0
HERE    B    HERE
```

6. FACTORIAL OF A NUMBER

```
        AREA   FACRORIAL,CODE,READONLY
ENTRY
        MOV R0.#7
        MOV R1,R0
FACT     SUB   R1,#1
        CMP R1,#1
         BEQ      STOP
         MUL R3,R0,R1
        MOV R0,R3
         BNE     FACT
  STOP
          END
```

8 Develop an ALP to find the sum of first 10 integer numbers

```
            AREA ADDSUM,CODE,READONLY

ENTRY

            MOV R0,#10

           MOV R1,R0

ADDIT   SUBS R1,R1,#1

            CMP R1,#0

             BEQ    STOP

              ADD R3,R0,R1

               MOV R0,R3

               BNE   ADDIT

STOP

                END
```

9   ALP to perform logical operation

```
             AREA LOGIC,CODE,READONLY

ENTRY

              LDR R0,=0X0A5A5    ;0000  1010   0101   1010  0101

              LDR R1,=0XFAF56    ;1111  1010   1111   0101  0110

              AND R3,R0,R1         ;0000  1010   0101   0000  0100 = 0A504

              ORR R4,R0,R1         ;1111  1010   1111   1111  0111 = FAFF7

               EOR  R5,R0,R1       ;1111  0000   1010   1111  0011 = F0AF3

                BIC  R6,R0,R1       ;0000  0000   0000   1010  0001 =  IT WILL CLEAR ALL THE BITS OF R0 IN

                                     ;   WHICH R1 BITS ARE 1 WHEN R1=0 ,RO BIT NO CHANGE =000A1

STOP        B   STOP

                END
```

10 Develop an ALP to perform SMLAL operation

```
        AREA TEST, CODE, READONLY
ENTRY
START
        ldr R0,=0x5
        ldr R1,=0x5
        LDR R2,=0X2
        LDR R3,=0X4
        SMLAL R3,R2,R0,R1       ;  [R2 R3]=[R2 R3]+R0*R1
                                ;R0*R1=0X5 * 0X5 =  0X19
                                ;[R2 R3]=[OX2 0X4]+0X19=0204+19=21D
                                ; R3=1D  R2=02
BACK B BACK
        END
```

11 DEVELOP AN ALP TO PERFORM MLA OPERATION

```
    AREA TEST, CODE, READONLY
ENTRY

START
        MOV R0,#6
        MOV R1,#5
        MOV R2,#1
        MLA R3,R0,R1,R2  ;R3=(R0*R1) + R2
                         ;R3=6*5 + 1 =   31  =  1F
BACK   B BACK
        END
```

12 DEVELOP AN ALP FOR BLOCK TRANSFER

```
AREA Program, CODE, READONLY

ENTRY
   MOV    R5,#6
   LDR    R0,=BLOCK1
   LDR    R1,=BLOCK2

NEXT
   LDRB   R2,[R0],#1
   STRB   R2,[R1],#1
   SUBS   R5,#1

BLOCK1  DCB  0x11,0x22,0x33,0x44,0x55,0x66
    AREA    Data1, DATA, READWRITE
BLOCK2  DCB  0
       END
```

13 Develop an ALP TO COUNT THE NUMBER OF ONES AND ZEROS IN TWO CONSECUTIVE MEMORY LOCATION

```
                      AREA ONEZERO, CODE, READONLY  ; Define a code section named ONEZERO

ENTRY                                                       ; Mark first instruction to execute

START
                            MOV R2, #0              ; Initialize counter for ones to 0
                              MOV R3, #0              ; Initialize counter for zeros to 0
                               MOV R7, #2              ; Counter to get two words
                               LDR R6, =VALUE          ; Load the address of VALUE into R6
LOOP
                               MOV R1, #32             ; Initialize a counter for 32 bits
                                LDR R0, [R6], #4        ; Load the 32-bit value from memory address in R6 into R0
LOOP0
                                MOVS R0, R0, ROR #1      ; Rotate right R0 by 1 bit and update flags
                                    BHI ONES               ; If carry bit is 1 (indicating a 1), branch to ONES
                                    ADD R3, R3, #1          ; Otherwise, increment the zero counter
                                    B LOOP1                ; Branch to LOOP1


ONES
                                     ADD R2, R2, #1          ; Increment the ones counter
                                      B LOOP1                ; Branch to LOOP1
LOOP1
                                      SUBS R1, R1, #1         ; Decrement the bit counter
                                       BNE LOOP0              ; If R1 is not zero, continue the loop at LOOP0


                                          SUBS R7, R7, #1         ; Decrement the word counter
                                           CMP R7, #0             ; Compare R7 to 0
                                BNE LOOP          ; If R7 is not zero, continue the loop at LOOP

BACK                            B BACK            ; Infinite loop to end the program

VALUE           DCD 0x11111111, 0xAA55AA55 ; Define two 32-bit values in memory

                                END               ; Mark end of file
```

14 DEVELOP AN ALP TO FIND THE LARGEST NUMBER IN AN ARRAY OF 32 NUMBERS

```
      AREA LARGEST, CODE, READONLY
        EXPORT __main


__main   MOV R5, #6         ; INITIALISE COUNTER TO 6 (i.e. N-7)
         LDR R1, =ARRAY      ; LOADS THE ADDRESS OF FIRST VALUE
        LDR R2, [R1], #4     ; WORD ALIGN TO ARRAY ELEMENT
LOOP   LDR R4, [R1], #4 ; WORD ALIGN TO ARRAY ELEMENT
        CMP R2, R4          ; COMPARE NUMBERS
         BHI LOOP1          ; IF THE FIRST NUMBER IS HIGHER THEN GOTO LOOP1
         MOV R2, R4          ; IF THE FIRST NUMBER IS < THEN MOV CONTENT R4 TO R2
LOOP1   SUBS R5, R5, #1      ; DECREMENT COUNTER
         CMP R5, #0          ; COMPARE COUNTER TO 0
         BNE LOOP           ; LOOP BACK TILL ARRAY ENDS
          LDR R4, =RESULT      ; LOADS THE ADDRESS OF RESULT
          STR R2, [R4]         ; STORES THE RESULT IN R1


ARRAY        DCD 0x44444444, 0xB00BDEFF, 0x11111111, 0x33333333, 0xAAAAAAAA, 0x88888888,
0x99999999
             AREA DATA2, DATA, READWRITE
   RESULT      DCD 0x0


               END
```