Question 1

a) This is a MARIE program to calculate the Fibonacci series of the given term. The value of 'n' will determine the number of Fibonacci terms of the series that the user wants to print. The program given below calculates the series based on a given set of values which the programmer has decoded to implement.

The MARIE assembly level language program is thus:

```
ORG 100
Input Input number
Store N        // The user inputs the number of terms to be printed
Store Ctr      // It stores the counter variable in the accumulator
Loop1, Clear
Load Ctr       // The counter is loaded into the processor form the accumulator
Subt C1        // The counter is subtracted by c1
Store Ctr
Load F2        // F2 is loaded into the accumulator
Add F1         // F1 is added
Store F3       // F3 is stored into the accumulator
Load F1        // F1 is loaded into the accumulator
Store F2       // F2 is stored into the accumulator
Load F3
Store F1
Load Ctr
```

```
Skipcond 400        // Check condition is skipped here
Jump Loop1          // Jump condition to the location 400
Load Ctr
Output
Load N              // Output is N
Output
Load F1             // the Final output is placed
Output
Halt                // control halts
N, DEC 0
Ctr, DEC 0
C1, DEC 1
F1, DEC 0
F2, DEC 1
F3, DEC 0
```

b) For some values of n, the program is not producing appropriate results as it should behave. This abnormality is occurring due to the limit of the integers being crossed by the system. There is a limit to the value that can be stored by data type. In this case after careful analysis of the output we have seen that the output produced is correct for values of n up to n= 24. After this threshold is reached or crossed the program starts displaying results in the negative value ranges.

Question 2:

We can still accommodate,

Total Number. of possible instruction encoding schemes = 2^{11}=2048

the number of encoding schemes taken by the 2 address instruction = $5\times2^4\times2^4$=1280

1 address instruction encoding number of address = 32×2^4=512

therefore total number of zero address instructions are = 2048−(1280+512)= 256

Question 3:

The zero address code for the same expression is:

ADD R1, B, C
ADD R2, D, E
MULT A, R1, R2

In a 2 address machine

LOAD R1, B
ADD R1, C
LOAD R2, E
ADD R2, E
MULT R2, R1
STORE A, R2

In a one address machine

Load B
Add C
Store Temp
Load D
Add E
Mult Temp
Store A

In zero address machine

Push B
Push C
Add
Push D
Push E
Add

Mult
Store A