1. Write a program to rotate left the data byte 84H four times.

PROGRAM STATEMENT

To write an assembly language program to rotate left the data byte 84H four times.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte 84H into accumulator.

STEP 2: Rotate left accumulator for four times

STEP 3: Store the content of accumulator in output port 05H.

STEP 4: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI A, 84H   // Load the data byte 84H into accumulator immediately.
RLC          // Rotate Left Accumulator
RLC
RLC
RLC
OUT 05H      // Store the result of accumulator into output port 05H.
HLT          // Terminate the program.
```

Input

A ← 84H

Output

Output port, 05H → 48H

2. Write a program to rotate right the data byte $F7_H$ two times.

PROGRAM STATEMENT

To write an assembly language program to rotate right the data byte F7H two times.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte F7H into accumulator.

STEP 2: Rotate accumulator right 2 times.

STEP 3: Store the content of accumulator in output port OSH.

ASSEMBLY LANGUAGE CODE

```
MVI A, F7H   // Load the data byte F7H into accumulator
                immediately.
RRC          // Rotate accumulator right
RRC          // Rotate accumulator right
HLT          // Terminate the program.
```

INPUT

A ← F7H

OUTPUT

Output port, OSH → FDH

3. Write a program to rotate left with carry the result of addition ($AA_H + EE_H$) three times.

PROGRAM STATEMENT

To write an assembly language program to rotate left with carry the result of addition ($AA_H + EE_H$) three times.

PROGRAM ALGORITHM:

ALGORITHM:

STEP 1: Load the data type AAH into accumulator.
STEP 2: Load the data type EEH into register B
STEP 3: Add the content of accumulator and register B.
STEP 4: Rotate accumulator left with carry, ~~four~~ three times.
STEP 5: Display the ~~ee~~ result of accumulator into output port 05H.
STEP 6: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI  A, AAH   // Load the data type AAH into accumulator.
MVI  B, EEH   // Load the data type EEH into register B.
ADD  B        // Add the content of accumulator and register B.
RAL           // Rotate accumulator left with carry.
RAL
RAL
RAL
OUT  05H      // Display result of accumulator into output port 05H
HLT           // Terminate the program.
```

Input

A ← AAH
B ← EEH

Output

Output port, 05H → C6H

4. Write a program to rotate right with carry the result of subtraction ($DE_H - FA_H$) two times.

PROGRAM STATEMENT

To write an assembly language program to rotate right with carry the result of subtraction ($DE_H - FA_H$) two times.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte DE_H into accumulator.

STEP 2: Load the data byte FA_H into register B.

STEP 3: Subtract the content of accumulator with register B.

STEP 4: Rotate accumulator right with carry 2 times.

STEP 5: Store the content of accumulator in output port 05H.

STEP 6: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI A, DEH   // Load the data type DE_H into accumulator.
MVI B, FAH   // Load the data type FA_H into register B.
SUB B        // Subtract the content of accumulator and
             register B.
RAR          // Rotate accumulator right with carry
RAR          // Rotate accumulator right with carry
OUT 05H      // Store the content of accumulator in output port
             05H.
HLT          // Terminate the program.
```

INPUT

A ← DEH

B ← FAH

OUTPUT

Output port, 05H → 78H

3. WAP to subtract $D7_H$ from $A9_H$

If borrow exists, rotate left with carry five times. Then subtract with borrow $A9_H$ from accumulator's content

Otherwise terminate.

PROGRAM STATEMENT

To write an assembly language program to subtract $D7_H$ from $A9_H$ and if borrow exists, rotate left with carry five times. Then subtract with borrow A9H from accumulator's content. Otherwise, terminate the program.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte A9 H into accumulator.

STEP 2: Load the data byte D7 H into register B.

STEP 3: Subtract the content of accumulator with register B.

STEP 4: Check for carry(borrow), if no carry jump to X.

STEP 5: Rotate accumulator left with carry for five times.

STEP 6: Store the result on accumulator into memory location 2000H.

STEP 7: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI A, A9H  // Load the data byte A9H into accumulator.
MVI B, D7H  // Load the data byte D7H into register B.
SUB B       // Subtract content of accumulator with register B.
JNC X       // Jump to X if no carry (borrow)
RAL         // Rotate accumulator left with carry.
RAL
RAL
RAL
RAL
X: STA 2000H // Store the result of accumulator
                into memory location 2000H.
HLT          // Terminate the program.
```

Input

A ← A9H

B ← D7H

Output

2000H → [redacted] H

5D

6. Comp to add two 1-byte integers 84_H and $F7_H$. If carry exists, rotate right with carry the accumulator result three times. Otherwise terminate.

PROGRAM STATEMENT

To write an assembly language program to add two 1-byte integers 84_H and $F7_H$. If carry exists, rotate right with carry the accumulator result three times. Otherwise terminate the program.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data integer 84H into accumulator.
STEP 2: Load the data integer F7H into register B.
STEP 3: Add the content of accumulator with register B.
STEP 4: Check for carry, if no carry jump to X.
STEP 5: Rotate accumulator right with carry for three times.
STEP 6: Store the result on accumulator into memory location 2000H.
STEP 7: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI A, 84H  // Load the data integer 84H into accumulator.
MVI B, F7H  // Load the data integer F7H into register B.
ADD B       // Add the content of accumulator with register B.
JNC X       // Jump to X if no carry
RAR         // Rotate right accumulator with carry
RAR
RAR
X: STA 2000H // Store the result of accumulator into memory location
                2000H.
HLT         // Terminate the program.
```

INPUT

A ← 84H
B ← F7H

OUTPUT

2000H → EF■H

7. Write a program to subtract CD_H from AB_H.

If result is negative, express 2's complement result in original diffrence form in register H. Also, save the sign bit in register L. Then copy H's content to E. Perform necessary operations to shift E's bit right five times. Perform necessary operations to add E's content with FF H. After that, add with carry whatever is in A's content with H's content. Then, rotate right with carry A's content 2 times. Otherwise terminate.

PROGRAM STATEMENT

To write an assembly language program to subtract CD_H from AB_H. If result is negative, express 2's complement result in original diffrence form in register H. Also, save the sign bit in register L. Then copy H's content to E. Perform necessary operations to shift E's bit right five times. Perform necessary operations to add E's content with FFH. After that, add with carry whatever is in A's content with its content. Then, rotate right with carry A's content 2 times. Otherwise, terminate.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte AB H into accumulator.

STEP 2: Load the data byte CD H into register B.

STEP 3: Subtract the content of accumulator with register B.

STEP 4: Check for result if positive or negative.

STEP 5: Take 2's complement of A

STEP 6: Copy result to H.

STEP 7: Set L to 1 the sign bit.

STEP 8: Copy content of H to register E.

STEP 9: Shift E's bits right 5 times.

STEP 10: Add E's content with FFH.

STEP 11: Add with carry whatever is in A's content with H's content.

STEP 12: Rotate right A's content 2 times.

STEP 13: Terminate the program.

1. ASSEMBLY LANGUAGE CODE

```
MVI A, ABH // Load the data byte ABH into accumulator.
MVI B, CDH // Load the data byte CDH into register B.
SUB B      // Subtract CDH from ABH

JM negative  // If result is negative, jump to negative level.
JMP end      // If result is positive, jump to end level.

negative: CMA   // Complement content of Accumulator.
ADI 01H         // Add 01H with accumulator immediately.
MOV H, A        // Copy content of accumulator to register H.
MVI L, 01H      // Set L to 1, indicating the negative result.
JMP end         // Unconditional jump to end ~~level~~ label.

end: MOV E, H   // Move the content of register H to register E.
MOV A, E        // Move the content of register E to accumulator
RRC             // Rotate right accumulator
RRC
RRC
RRC
RRC
ADI FFH         // Add A's content immediate with value FFH.
ADC H           // Add with carry whatever is in A's content with H's
                content.
RAR             // Rotate right accumulator with carry
RAR             // Rotate right accumulator with carry
HLT             // Terminate the program
```

Input

A ← ABH

B ← CDH

Output

A → BCH

1. Write a program to multiply two 8-bit numbers
   a. by Repetition Addition Method
   b. by Rotation Method

Repetion Additon Method

PROGRAM STATEMENT

To write an assembly language program to multiply two 8-bit numbers 05_H and 03_H.

a. by Repetition Method

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the data byte 00H into accumulator.

STEP 2: Load the data byte 05H into register B.

STEP 3: Load the data byte 03H into accumulator.

STEP 4: Add the content of register B with accumulator on label X.

STEP 5: Decreased the content of register C by 1.

STEP 6: Jump to X if not zero in register C.

STEP 7: Store the content of accumulator on memory location 2055H.

STEP 8: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI A, 00H      //Load 00H into accumulator.
MVI B, 05H      // Load 05H into register B
MVI C, 03H      // Load 03H into register C.
X: ADD B        // Add content of accumulator and register B.
DCR C           // Decrease content of register C by 1.
JNZ X           // Jump to X if content of C is not zero.
STA 2055H       // Store the result of accumulator in 2055H.
HLT             // Terminate the program.
```

Input

A← 00H B← 05H C← 03H

Output

9. Rotation Method

2. Write a program to divide OF_H data byte by 07_H data byte by repeated subtraction method.

PROGRAM STATEMENT

To write an assembly language program to divide OF_H data byte by 07_H data byte by repeated subtraction method.

PROGRAM ANALYSIS

Algorithm:

STEP 1: Load accumulator with data byte OFH.
STEP 2: Load register B with data byte 07H.
STEP 3: Load register C with data byte 00H.
STEP 4: Compare A with B on label X.
STEP 5: Jump to Y if carry.
STEP 6: Subtract content of A with B.
STEP 7: Increament content of register C by 1.
STEP 8: Unconditional jump to X.
STEP 9: Store the result of accumulator on mem. location 2050H.
STEP 10: Move the content of register C to accumulator.
STEP 11: Store the result of accumulator on mem. location 2055H.
STEP 12: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI  A, OF H   // Load accumulator with data byte OFH.
MVI  B, 07H    // load register B with data byte 07H.
MVI  C, 00H    // Load register C with data byte 00H.
X: CMP B       // Compare A with B.
JC Y           // Jump to label Y if carry.
SUB B          // Subtract B's content with A's content.
INR C          // Increament the content of register C by 1.
JMP X          // Unconditional Jump to X
```

MOV A, C // [illegible] to [illegible]

STA 2055H // Store the content of accumulator to memory location 2055H.

HLT // Terminate the program.

INPUT

A ← 00H

B ← 05H

C ← 03H

OUTPUT

2055H → 02H

2056H → 01H

3. Write a program to multiply 16 bit number 2022H by 8-bit number 03H.

PROGRAM STATEMENT

To write a assembly language program to multiply 16 bit number 2022H by 8-bit number 03H.

PROGRAM ANALYSIS

ALGORITHM:

STEP 1: Load the 8 bit-multiplier into register B.

STEP 2: Initialize 16 bit result to zero.

STEP 3: Load the 16 bit-number into register HL.

STEP 4: Load the high byte of the 16-bit number into accumulator.

STEP 5: Save a copy of the high byte in register D.

STEP 6: Load the lower-byte of the 16-bit number into accumulator.

STEP 7: Save a copy of the lower byte in register E.

STEP 8: Load 8-bit multiplier into register A.

STEP 9: Double the value of the 8-bit multiplier, for 2 times.

STEP 10: Add the high-byte of the 16-bit number to the result.

STEP 11: Add the high-byte of the 16-bit number to the result with carry.

STEP 12: Load the 8-bit multiplier into register A.

STEP 13: Add the low byte of the 16 bit number to the result.

STEP 14: Add the low byte of the 16 bit number to the result with carry.

STEP 15: Store the low byte of the result in memory.

STEP 16: Increase the memory pointer.

STEP 17: Move the content of register H to accumulator.

STEP 18: Store the high byte of the result in memory location 200

STEP 20: Terminate the program.

ASSEMBLY LANGUAGE CODE

```
MVI B, 03H  // Load the 8 bit multiplier into register B
MVI C, 00H  // Initialize the 16 bit result to zero.
LXI H, 2020H  // Load the 16-bit number into register HL.
MOV A,H  // Load the high byte of 16-bit number into accumulator.
MOV D,A  // Save a copy of the high byte in register D.
MOV A,L  // Load the lower byte of 16-bit number into accumulator.
MOV E,A  // Save a copy of the lower byte in register E.
MOV A,B  // Load 8-bit multiplier into accumulator.
ADD A  // Double the value of the 8-bit multiplier.
ADD A  // Double the value of the 8-bit multiplier.
ADD H  // Add the high of the 16-bit number to the result.
ADC D  // Add the high byte of the 16-bit number to result with carry.
MOV A,B  // Load the 8-bit multiplier into register A.
ADD E  // Add the low-byte of the 16-bit number to the result.
ADC C  // Add the low-byte of the 16-bit number to the result with carry
MOV M,L  // Store the low-byte of the result in memory
INX H  // Increament the memory point
MOV A,H  // Move the result of register H to accumulator.
STA 2000H  // Store the result of accumulator in memory location
           2000H.
HLT  // Terminate the program.
```

INPUT

B ← 03H

HL ← 2020H

C ← 00H

OUTPUT

2000H → 20H