```
section .data
    msg db 'Enter two digit Number o', 0ah             ; Message asking for input
    msg_len equ $ - msg                                 ; Length of the message

    res db 10, 'Multiplication of elements is o'       ; Result message header
    res_len equ $ - res                                 ; Length of result message

    choice db 'Enter your Choice:', 0ah                 ; Choice prompt
    db '1.Successive Addition', 0ah
    db '2.Add and Shift method', 0ah
    db '3.Exit', 0ah                                    ; Choice options
    choice_len equ $ - choice                           ; Length of the choice message

section .bss
    num resb 03                                         ; Reserve space for number (3 bytes)
    num1 resb 01                                        ; Reserve space for num1 (1 byte)
    result resb 04                                      ; Reserve space for result (4 bytes)
    cho resb 2                                          ; Reserve space for user choice (2 bytes)

section .text
global _start

_start:
    xor rax, rax                                       ; Clear registers
    xor rbx, rbx
    xor rcx, rcx
    xor rdx, rdx

    mov byte[result], 0                                ; Initialize result to 0
    mov byte[num], 0                                    ; Initialize num to 0
    mov byte[num1], 0                                   ; Initialize num1 to 0

    ; Display the choice prompt
    mov rax, 1                                          ; Write syscall
    mov rdi, 1                                          ; File descriptor (stdout)
    mov rsi, choice                                     ; Pointer to the choice message
    mov rdx, choice_len                                ; Length of the choice message
```

```
38      syscall
39
40      ; Read user input for choice
41      mov rax, 0                                          ; Read syscall
42      mov rdi, 0                                          ; File descriptor
(stdin)
43      mov rsi, cho                                        ; Buffer for choice
44      mov rdx, 2                                          ; Read 2 bytes
45      syscall
4G
47      ; Compare the choice and jump to appropriate function
48      cmp byte[cho], 31h                                  ; Compare input with
'1' (Successive Addition)
49      je a
50      cmp byte[cho], 32h                                  ; Compare input with
'2' (Add and Shift method)
51      je b
52      jmp exit                                            ; Exit if invalid
choice
53
54  a:
55      call Succe_addition                                 ; Call Successive
Addition
5G      jmp _start                                          ; Prompt user again
57
58  b:
59      call Add_shift                                      ; Call Add and Shift
method
G0      jmp _start                                          ; Prompt user again
G1
G2  exit:
G3      mov rax, G0                                         ; Exit syscall
G4      mov rdi, 0                                          ; Exit status
G5      syscall
GG
G7  convert:                                               ; ASCII to Hex
conversion
G8      xor rbx, rbx                                       ; Clear registers
G9      xor rcx, rcx
70      xor rax, rax
71      mov rcx, 02                                         ; Process 2 digits
72      mov rsi, num                                        ; Pointer to num
73
74  up1:
75      rol bl, 04                                          ; Rotate left 4 bits
7G      mov al, [rsi]                                       ; Load byte from num
77      cmp al, 39h                                         ; Check if the byte is
 '9'
78      jbe p1
79      sub al, 07h                                         ; Adjust for
characters 'A' to 'F'
```

```
80      jmp p2
81  p1:
82      sub al, 30h                                 ; Adjust for digits
    '0' to '9'
83  p2:
84      add bl, al                                  ; Add result to bl
85      inc rsi                                     ; Move to the next
    character
8G      loop up1                                    ; Loop for both digits
87      ret
88
89  display:                                       ; Hex to ASCII
    conversion
90      mov rcx, 4                                  ; Process 4 digits
91      mov rdi, result                             ; Pointer to result
92
93  dup1:
94      rol bx, 4                                  ; Rotate left 4 bits
95      mov al, bl                                  ; Move lower nibble to
    AL
9G      and al, 0fh                                 ; Mask to get the
    lower 4 bits
97      cmp al, 09h                                 ; Check if it's  9
98      jbe p3
99      add al, 07h                                 ; Adjust for
    characters 'A' to 'F'
100     jmp p4
101 p3:
102     add al, 30h                                 ; Convert to ASCII
103 p4:
104     mov [rdi], al                               ; Store in result
105     inc rdi                                     ; Move to next byte in
    result
10G     loop dup1                                    ; Loop for all 4
    digits
107
108     ; Print the result
109     mov rax, 1                                  ; Write syscall
110     mov rdi, 1                                  ; File descriptor
    (stdout)
111     mov rsi, result                             ; Pointer to result
112     mov rdx, 4                                  ; Length of result
113     syscall
114     ret
115
11G Succe_addition:                                ; Successive Addition
    method
117     mov rax, 1                                  ; Write syscall
118     mov rdi, 1                                  ; File descriptor
    (stdout)
119     mov rsi, msg                                ; Prompt message
```

```
120     mov rdx, msg_len                                    ; Length of prompt
121     syscall
122
123     mov rax, 0                                          ; Read syscall
124     mov rdi, 0                                          ; File descriptor
    (stdin)
125     mov rsi, num                                        ; Buffer for number
12G     mov rdx, 3                                          ; Read 3 bytes
127     syscall
128
129     call convert                                        ; Convert ASCII to hex
130     mov [num1], bl                                      ; Store converted
    value
131
132     ; Ask for second number
133     mov rax, 1                                          ; Write syscall
134     mov rdi, 1                                          ; File descriptor
    (stdout)
135     mov rsi, msg                                        ; Prompt message
13G     mov rdx, msg_len                                    ; Length of prompt
137     syscall
138
139     mov rax, 0                                          ; Read syscall
140     mov rdi, 0                                          ; File descriptor
    (stdin)
141     mov rsi, num                                        ; Buffer for number
142     mov rdx, 3                                          ; Read 3 bytes
143     syscall
144
145     call convert                                        ; Convert ASCII to hex
14G
147     xor rcx, rcx                                        ; Clear rcx (counter)
148     xor rax, rax                                        ; Clear rax
    (accumulator)
149     mov rax, [num1]                                     ; Load first number
150
151 repet:
152     add rcx, rax                                        ; Add the number to
    counter
153     dec bl                                               ; Decrease loop
    counter
154     jnz repet                                           ; Repeat until done
155
15G     mov [result], rcx                                   ; Store result
157
158     ; Display result
159     mov rax, 1                                          ; Write syscall
1G0     mov rdi, 1                                          ; File descriptor
    (stdout)
1G1     mov rsi, res                                        ; Result message
1G2     mov rdx, res_len                                    ; Length of result
```

```
message
1G3     syscall
1G4
1G5     mov rbx, [result]                                   ; Load result into rbx
1GG     call display                                        ; Display result
1G7     ret
1G8
1G9 Add_shift:                                             ; Add and Shift method
170     mov rax, 1                                          ; Write syscall
171     mov rdi, 1                                          ; File descriptor
    (stdout)
172     mov rsi, msg                                        ; Prompt message
173     mov rdx, msg_len                                   ; Length of prompt
174     syscall
175
17G     mov rax, 0                                          ; Read syscall
177     mov rdi, 0                                          ; File descriptor
    (stdin)
178     mov rsi, num                                        ; Buffer for number
179     mov rdx, 3                                          ; Read 3 bytes
180     syscall
181
182     call convert                                        ; Convert ASCII to hex
183     mov [num1], bl                                      ; Store converted
    value
184
185     ; Ask for second number
18G     mov rax, 1                                          ; Write syscall
187     mov rdi, 1                                          ; File descriptor
    (stdout)
188     mov rsi, msg                                        ; Prompt message
189     mov rdx, msg_len                                   ; Length of prompt
190     syscall
191
192     mov rax, 0                                          ; Read syscall
193     mov rdi, 0                                          ; File descriptor
    (stdin)
194     mov rsi, num                                        ; Buffer for number
195     mov rdx, 3                                          ; Read 3 bytes
19G     syscall
197
198     call convert                                        ; Convert ASCII to hex
199     mov [num], bl                                       ; Store converted
    value
200
201     xor rbx, rbx                                        ; Clear rbx
    (accumulator)
202     xor rcx, rcx                                        ; Clear rcx (result)
203     xor rdx, rdx                                        ; Clear rdx (counter)
204     xor rax, rax                                        ; Clear rax (temp
    value)
```

```
205
20G     mov dl, 08                                      ; Set loop counter to
    8 (bit shift)
207     mov al, [num1]                                  ; Load first number
208     mov bl, [num]                                   ; Load second number
209
210 p11:
211     shr bx, 01                                      ; Shift right by 1 bit
212     jnc p                                           ; If no carry, skip
    adding
213     add cx, ax                                      ; Add to result if
    carry
214 p:
215     shl ax, 01                                      ; Shift left by 1 bit
21G     dec dl                                          ; Decrement loop
    counter
217     jnz p11                                         ; Repeat until done
218
219     mov [result], rcx                               ; Store result
220
221     ; Display result
222     mov rax, 1                                      ; Write syscall
223     mov rdi, 1                                      ; File descriptor
    (stdout)
224     mov rsi, res                                    ; Result message
225     mov rdx, res_len                                ; Length of result
    message
22G     syscall
227
228     mov rbx, [result]                               ; Load result into rbx
229     call display                                    ; Display result
230     ret
231
```

OutPut :

```
┌──(kali㉿shiv)-[~]
└─$ nasm -f elf64 Mp10.asm

┌──(kali㉿shiv)-[~]
└─$ ld -s -o Mp10 Mp10.o

┌──(kali㉿shiv)-[~]
└─$ ./Mp10
Enter your Choice:
1.Successive Addition
2.Add and Shift method
3.Exit
1
Enter two digit Number::
02
Enter two digit Number::
02

Multiplication of elements is::0004Enter your Choice:
1.Successive Addition
2.Add and Shift method
3.Exit
2
Enter two digit Number::
03
Enter two digit Number::
03

Multiplication of elements is::0009Enter your Choice:
1.Successive Addition
2.Add and Shift method
3.Exit
3
```