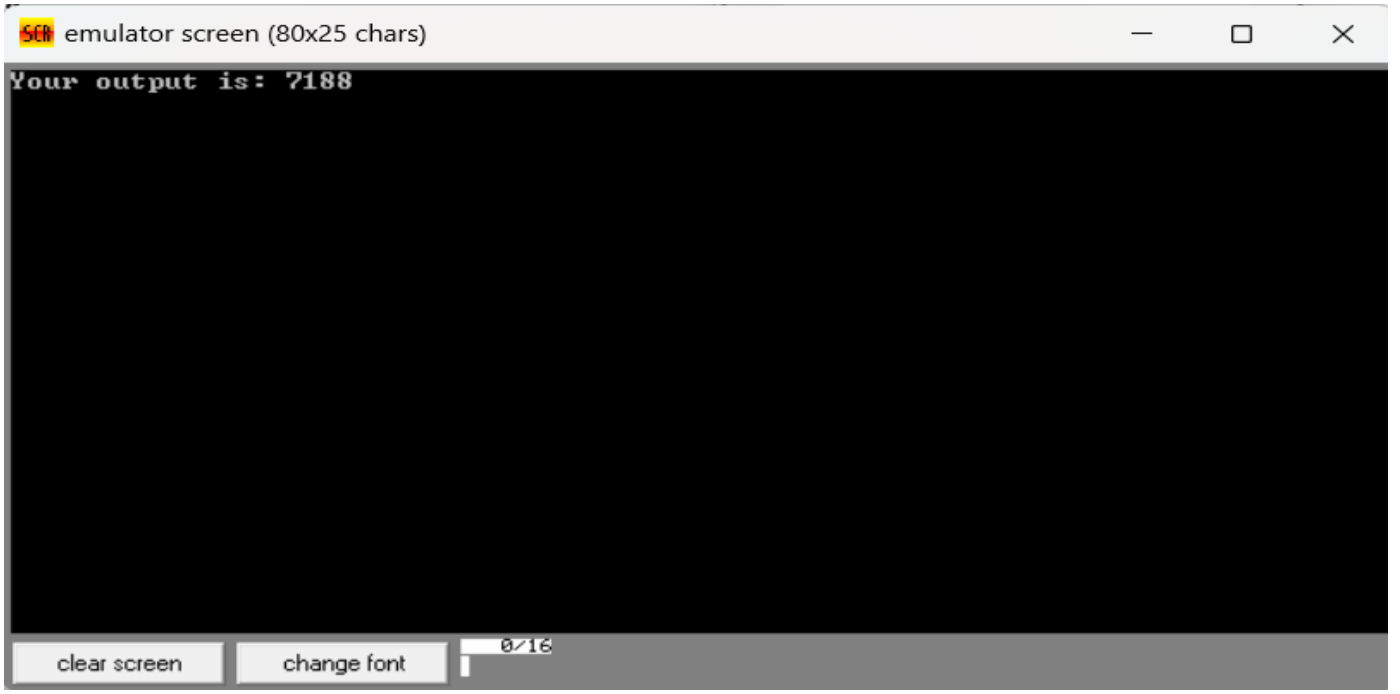
1).Write an assembly language program to perform multiplication of 8-bit data.

Code:-

```
.model small
.stack 100h
.data
   num1 db 5h
   num2 db 4h
   result dw 0
   msg db 'Your output is: $'

.code
main proc
   mov ax, @data
   mov ds, ax

   mov al, num1
   mov bl, num2
   xor cx, cx
   xor dx, dx

multiply:
   test bl, 1
   jz skip_add

   add cx, ax

skip_add:
   shl ax, 1
   shr bl, 1

   inc dx
   cmp dx, 8
   jl multiply

   mov result, cx

   lea dx, msg
   mov ah, 09h
   int 21h

   mov ax, result
   call DisplayResult

   mov ah, 4Ch
   int 21h
```

```
main endp

DisplayResult proc
   mov bx, 10
   xor cx, cx

convert_digit:
   xor dx, dx
   div bx
   add dl, '0'
   push dx
   inc cx
   test ax, ax
   jnz convert_digit

display_digit:
   pop dx
   mov ah, 02h
   int 21h
   loop display_digit

   ret
DisplayResult endp

end main
```

Output:-



2. Write a program in assembly language to perform multiplication of 16-bit data.

Code:-

```
.model small
.stack 100h
.data
   msg db 'Your final output is: $'
   num1 dw 4321h
   num2 dw 1234h
   result dw 0
   result_hi dw 0

.code
main proc
   mov ax, @data
   mov ds, ax

   mov ah, 09h
   lea dx, msg
   int 21h

   mov ax, num1
   mov dx, num2

   mul dx

   mov result, ax
   mov result_hi, dx

   call DisplayHex32

   mov ah, 4Ch
   int 21h

main endp

DisplayHex32 proc
   mov ax, result_hi
   call DisplayHex16
   mov ax, result
   call DisplayHex16
```

```
    ret
DisplayHex32 endp

DisplayHex16 proc
    push ax
    mov cx, 4

next_digit:
    rol ax, 4
    mov dx, ax
    and dx, 0Fh
    cmp dx, 0Ah
    jl display_digit
    add dl, 7h
display_digit:
    add dl, '0'
    mov ah, 02h
    int 21h
    loop next_digit
    pop ax
    ret
DisplayHex16 endp

end main
```

Output:-