Hands-on Exercises

1. EXER1.ASM
   * CODE

*;Filename: EXER1.ASM*

*;display*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100

      start:

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,62h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,64h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,66h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,67h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,68h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Ah

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Bh

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,70h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,71h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,76h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,77h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,78h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,79h

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,7Ah

            mov dl,cl

            int 21h

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,41h

            mov dl,cl

            int 21h

*;B*

            mov ah,02h

            mov cl,42h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;D*

            mov ah,02h

            mov cl,44h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,45h

            mov dl,cl

            int 21h

*;F*

            mov ah,02h

            mov cl,46h

            mov dl,cl

            int 21h

*;G*

            mov ah,02h

            mov cl,47h

            mov dl,cl

            int 21h

*;H*

            mov ah,02h

            mov cl,48h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,49h

            mov dl,cl

            int 21h

*;J*

            mov ah,02h

            mov cl,4Ah

            mov dl,cl

            int 21h

*;K*

            mov ah,02h

            mov cl,4Bh

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,4Ch

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,4Dh

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,4Eh

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,4Fh

            mov dl,cl

            int 21h

*;P*

            mov ah,02h

            mov cl,50h

            mov dl,cl

            int 21h

*;Q*

            mov ah,02h

            mov cl,51h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,52h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,53h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,54h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,55h

            mov dl,cl

            int 21h

*;V*

            mov ah,02h

            mov cl,56h

            mov dl,cl

            int 21h

*;W*

            mov ah,02h

            mov cl,57h

            mov dl,cl

            int 21h

*;X*

            mov ah,02h

            mov cl,58h

            mov dl,cl

            int 21h

*;Y*

            mov ah,02h

            mov cl,59h

            mov dl,cl

            int 21h

*;Z*

            mov ah,02h

            mov cl,5Ah

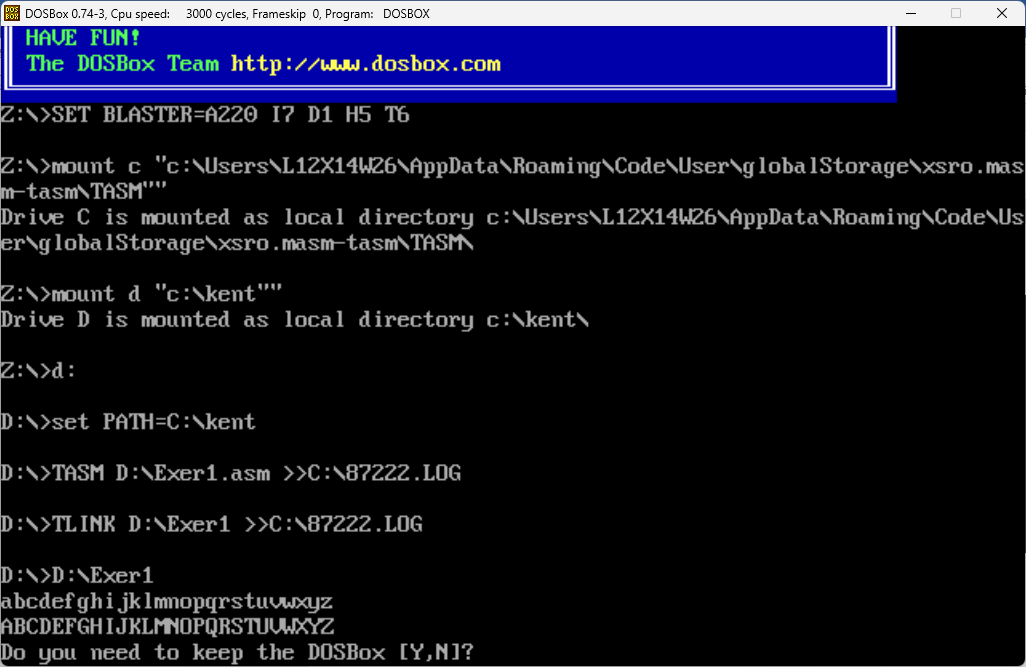
            mov dl,cl

            int 21h

            int 27h         *; terminate*

end start *; end program*

* + RUN SCREENSHOT



1. EXER2.ASM
   * CODE

*;Filename: EXER2.ASM*

*;display using single characters:*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100

      start:

*;K*

            mov ah,02h

            mov cl,4Bh

            mov dl,cl

            int 21h

*;e*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;n*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;t*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,41h

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,52h

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;B*

            mov ah,02h

            mov cl,42h

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;H*

            mov ah,02h

            mov cl,68h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;F*

            mov ah,02h

            mov cl,66h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,53h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,49h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

*;P*

            mov ah,02h

            mov cl,70h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,53h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;2*

            mov ah,02h

            mov cl,32h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;D*

            mov ah,02h

            mov cl,64h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;Y*

            mov ah,02h

            mov cl,79h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;A*

            mov ah,02h

            mov cl,61h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;G*

            mov ah,02h

            mov cl,67h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;F*

            mov ah,02h

            mov cl,66h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

*;P*

            mov ah,02h

            mov cl,70h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;D*

            mov ah,02h

            mov cl,64h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;B*

            mov ah,02h

            mov cl,62h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;F*

            mov ah,02h

            mov cl,66h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,63h

            mov dl,cl

            int 21h

*;H*

            mov ah,02h

            mov cl,68h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;G*

            mov ah,02h

            mov cl,67h

            mov dl,cl

            int 21h

*;Y*

            mov ah,02h

            mov cl,79h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;-*

            mov ah,02h

            mov cl,2Dh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;N*

            mov ah,02h

            mov cl,6Eh

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;V*

            mov ah,02h

            mov cl,76h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;Y*

            mov ah,02h

            mov cl,79h

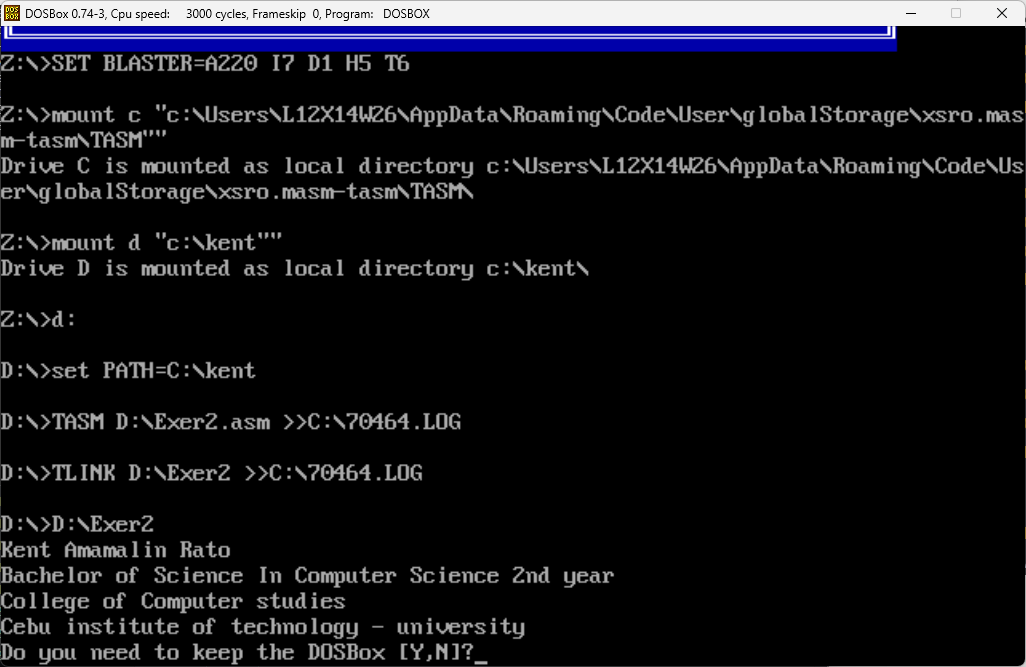
            mov dl,cl

            int 21h

            int 27h         *; terminate*

end start *; end program*

* + RUN SCREENSHOT



1. EXER3.ASM
   * CODE

*;Filename: EXER3.ASM*

*;Display*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100

      start:

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;0*

            mov ah,02h

            mov cl,30h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;1*

            mov ah,02h

            mov cl,31h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;2*

            mov ah,02h

            mov cl,32h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;3*

            mov ah,02h

            mov cl,33h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;4*

            mov ah,02h

            mov cl,34h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;5*

            mov ah,02h

            mov cl,35h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;6*

            mov ah,02h

            mov cl,36h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;7*

            mov ah,02h

            mov cl,37h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;8*

            mov ah,02h

            mov cl,38h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;9*

            mov ah,02h

            mov cl,39h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;!*

            mov ah,02h

            mov cl,21h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;\**

            mov ah,02h

            mov cl,2Ah

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;#*

            mov ah,02h

            mov cl,23h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;$*

            mov ah,02h

            mov cl,24h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;%*

            mov ah,02h

            mov cl,25h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;^*

            mov ah,02h

            mov cl,5Eh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;&*

            mov ah,02h

            mov cl,26h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;\**

            mov ah,02h

            mov cl,2Ah

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;(*

            mov ah,02h

            mov cl,28h

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;)*

            mov ah,02h

            mov cl,29h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;!*

            mov ah,02h

            mov cl,2Dh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;\_*

            mov ah,02h

            mov cl,5Fh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;+*

            mov ah,02h

            mov cl,2Bh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;=*

            mov ah,02h

            mov cl,3Dh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;{*

            mov ah,02h

            mov cl,7Bh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;}*

            mov ah,02h

            mov cl,7Dh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;[*

            mov ah,02h

            mov cl,5Bh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;]*

            mov ah,02h

            mov cl,5Dh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;<*

            mov ah,02h

            mov cl,3Ch

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;?*

            mov ah,02h

            mov cl,3Fh

            mov dl,cl

            int 21h

*;Space*

            mov ah,02h

            mov cl,20h

            mov dl,cl

            int 21h

*;;*

            mov ah,02h

            mov cl,3bh

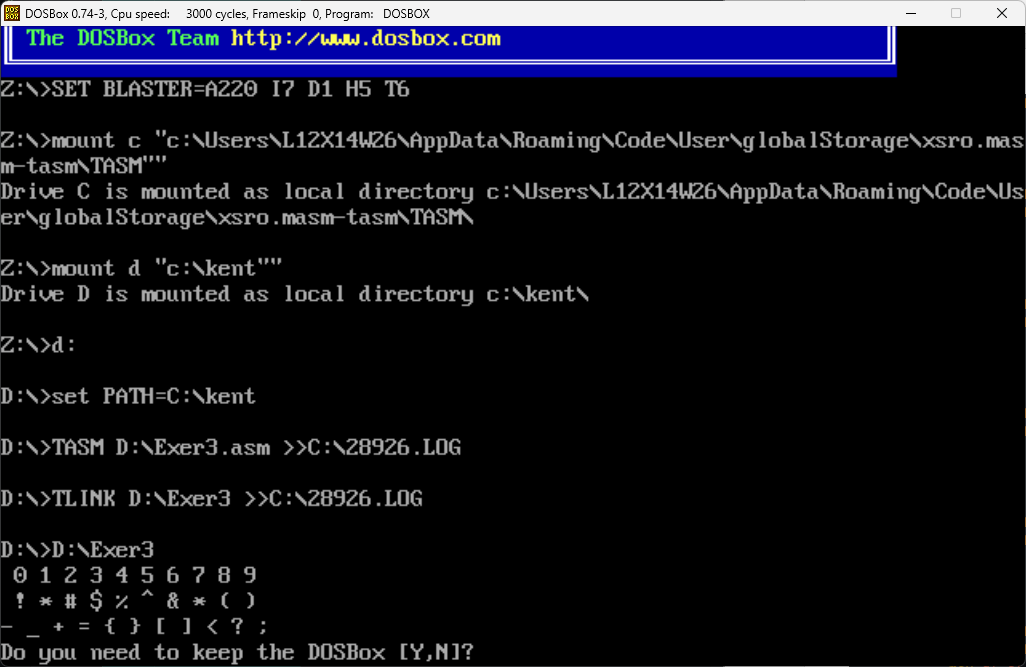
            mov dl,cl

            int 21h

            int 27h         *; terminate*

end start *; end program*

* + RUN SCREENSHOT



1. EXER4.ASM
   * CODE

*;Filename: EXER4.ASM*

*; display with different foreground and background colors*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100

      start:

            mov ah,09h

            mov bl,3Eh

            mov cx,7

            int 10h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;L*

            mov ah,02h

            mov cl,6Ch

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;G*

            mov ah,02h

            mov cl,67h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

            mov ah,09h

            mov bl,4Eh

            mov cx,2

            int 10h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;F*

            mov ah,02h

            mov cl,66h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;C*

            mov ah,02h

            mov cl,43h

            mov dl,cl

            int 21h

*;O*

            mov ah,02h

            mov cl,6Fh

            mov dl,cl

            int 21h

*;M*

            mov ah,02h

            mov cl,6Dh

            mov dl,cl

            int 21h

*;P*

            mov ah,02h

            mov cl,70h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;R*

            mov ah,02h

            mov cl,72h

            mov dl,cl

            int 21h

*;\n*

            mov ah,02h

            mov cl,0Ah

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

            mov dl,cl

            int 21h

*;T*

            mov ah,02h

            mov cl,74h

            mov dl,cl

            int 21h

*;U*

            mov ah,02h

            mov cl,75h

            mov dl,cl

            int 21h

*;D*

            mov ah,02h

            mov cl,64h

            mov dl,cl

            int 21h

*;I*

            mov ah,02h

            mov cl,69h

            mov dl,cl

            int 21h

*;E*

            mov ah,02h

            mov cl,65h

            mov dl,cl

            int 21h

*;S*

            mov ah,02h

            mov cl,73h

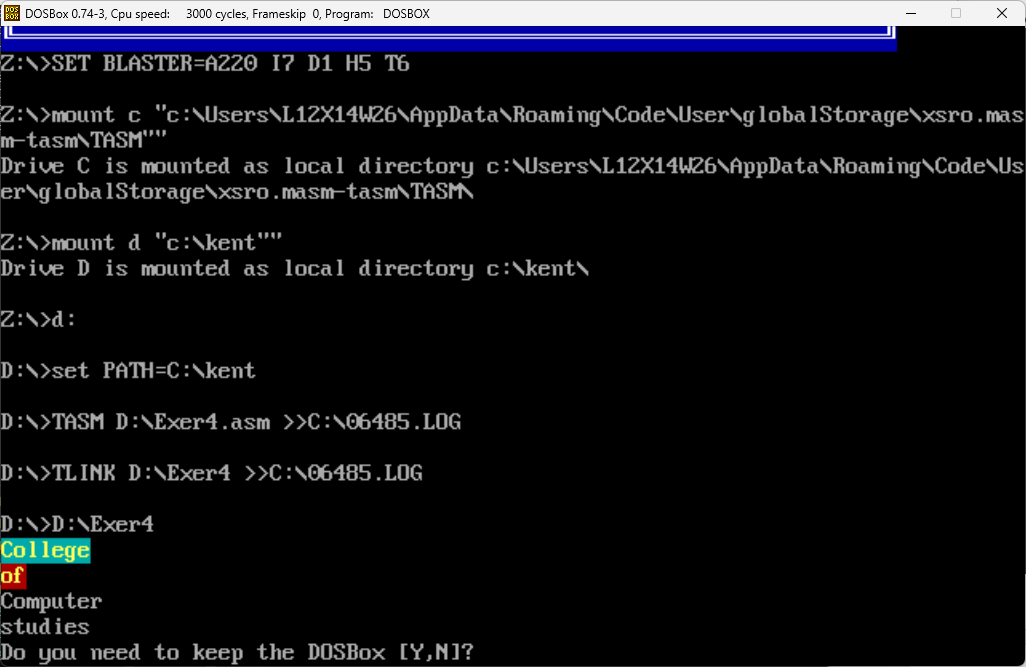
            mov dl,cl

            int 21h

            int 27h         *; terminate*

end start *; end program*

* + RUN SCREENSHOT



1. EXER5.ASM
   * CODE

*;Filename: EXER5.ASM*

*;Displaying single characters, numbers, and symbols*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100h

    start:

*;1st row color*

          mov ah,09h

          mov bl,11h

          mov cx,8

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;\n*

          mov ah,02h

          mov cl,0Ah

          mov dl,cl

          int 21h

*;2nd row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,3

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;2nd row color red*

          mov ah,09h

          mov bl,44h

          mov cx,2

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;2nd row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,3

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;\n*

          mov ah,02h

          mov cl,0Ah

          mov dl,cl

          int 21h

*;3rd row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,1

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;3rd row color red*

          mov ah,09h

          mov bl,44h

          mov cx,6

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;3rd row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,1

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;\n*

          mov ah,02h

          mov cl,0Ah

          mov dl,cl

          int 21h

*;4th row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,3

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;4th row color red*

          mov ah,09h

          mov bl,44h

          mov cx,2

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;4th row color blue*

          mov ah,09h

          mov bl,11h

          mov cx,3

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;\n*

          mov ah,02h

          mov cl,0Ah

          mov dl,cl

          int 21h

*;5th row color*

          mov ah,09h

          mov bl,11h

          mov cx,8

          int 10h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

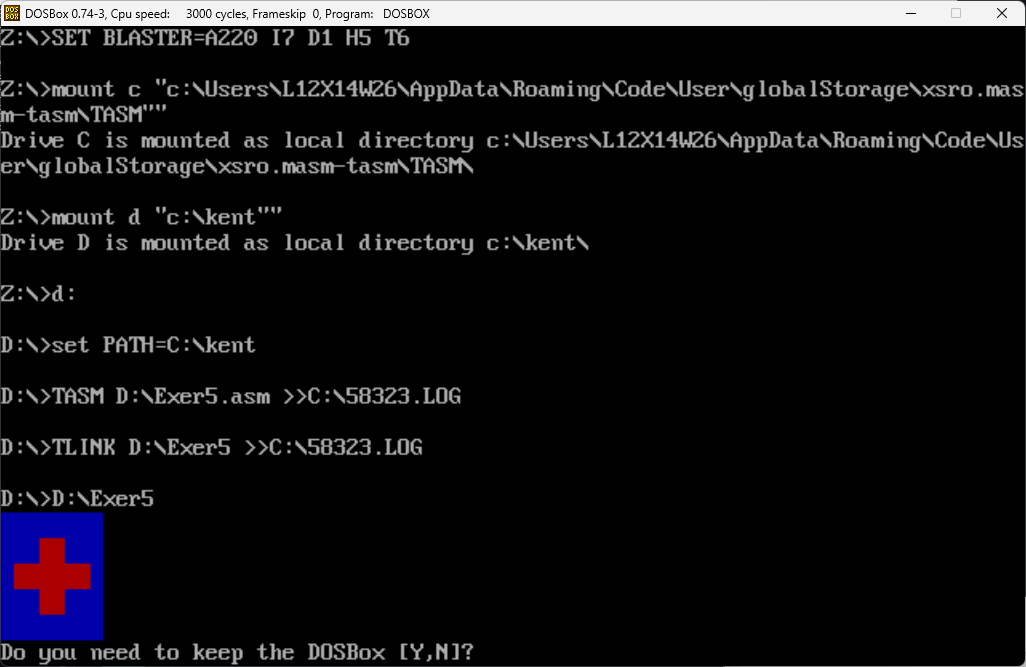
          mov dl,cl

          int 21h

          int 27h

end start

* + RUN SCREENSHOT



1. EXER6.ASM
   * CODE

*;Filename: EXER6.ASM*

*;Displaying single characters, numbers, and symbols*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.code

.stack 100h

    start:

*;blinking*

          mov ah,09h

          mov bl,8Eh

          mov cx,1

          int 10h

*;I*

          mov ah,02h

          mov cl,49h

          mov dl,cl

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;blinking red*

          mov ah,09h

          mov bl,84h

          mov cx,1

          int 10h

*;red heart*

          mov ah,02h

          mov dl,03h

          int 21h

          mov ah,02h

          mov cl,20h

          mov dl,cl

          int 21h

*;blinking*

          mov ah,09h

          mov bl,8Eh

          mov cx,1

          int 10h

*;U*

          mov ah,02h

          mov cl,55h

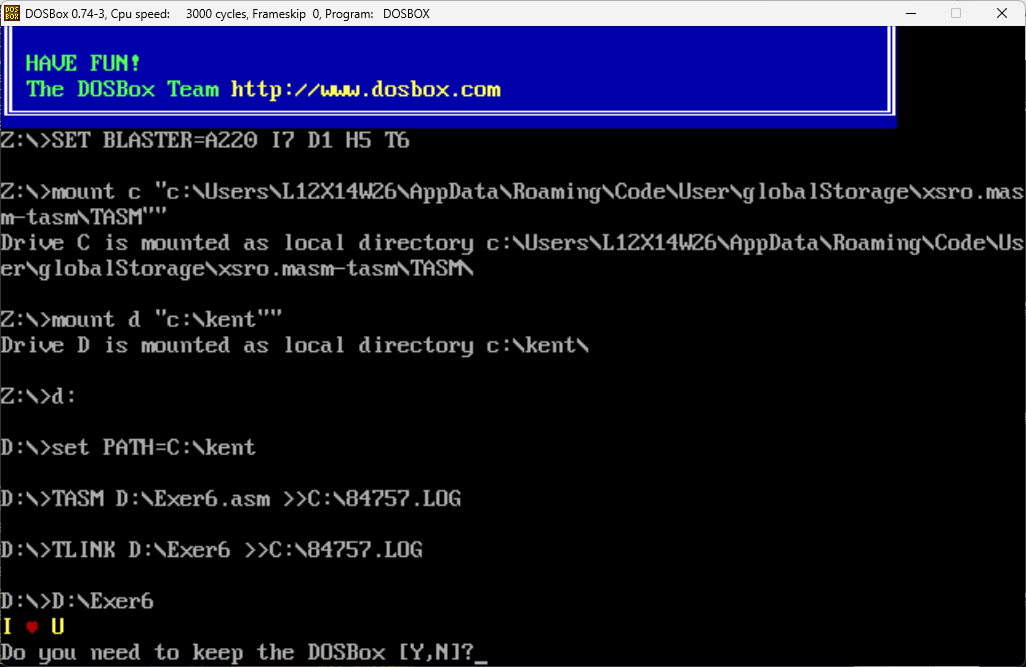
          mov dl,cl

          int 21h

          int 27h

end start

* + RUN SCREENSHOT



1. EXER7.ASM
   * CODE

.model small

.code

.stack 100h

    start:

*; 1 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 2 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 3 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 4 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 5 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 6 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 7 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 8 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,77h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,77h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,77h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 9 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 10 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,44h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,44h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,44h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,44h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 11 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,22h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 12 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,11h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 13 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 14 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 15 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*; 16 row colors*

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,09h

    mov bl,00h

    mov cx,1

    int 10h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,20h

    mov dl,cl

    int 21h

    mov ah,02h

    mov cl,0Ah

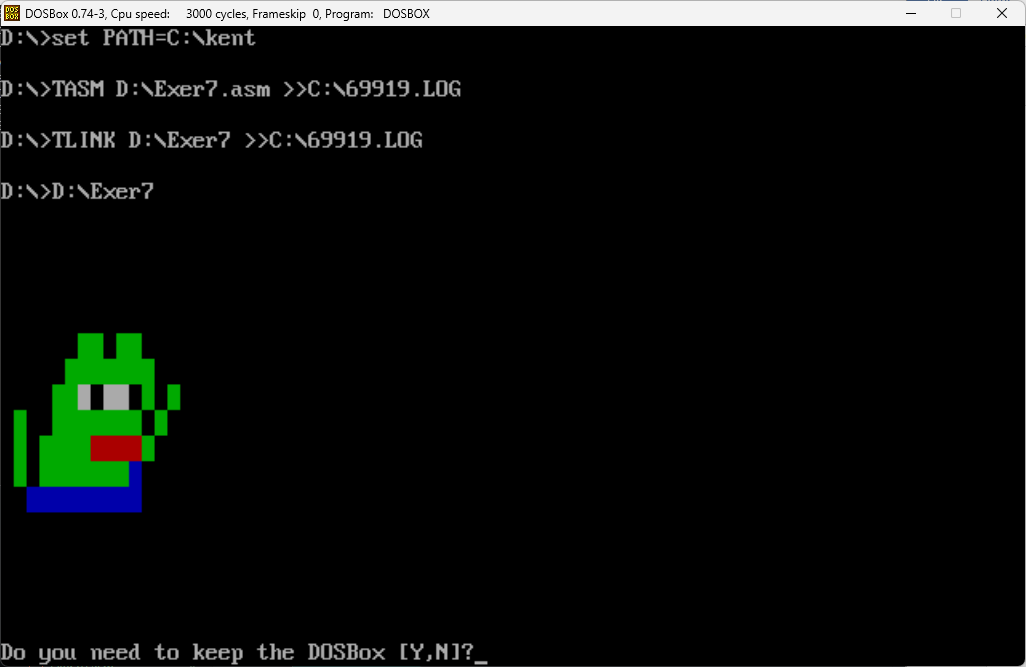
    mov dl,cl

    int 21h

    int 27h

end start

* + RUN SCREENSHOT



1. EXER8.ASM
   * CODE

*;Filename: EXER8.ASM*

*;Displaying strings of text*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.stack 100h

.data

    line1 db 'Kent A Rato$'

    line2 db 'Bachelor of Science in Computer Science 2nd Year$'

    line3 db 'College of Computer Studies$'

    line4 db 'Cebu Institute of Technology - University$'

.code

main:

*;set up data*

    mov ax,@data

    mov ds, ax

*;line1*

    mov ah,09h

    mov dx,offset line1

    int 21h

*;\n*

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*;line2*

    mov ah,09h

    mov dx,offset line2

    int 21h

*;\n*

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*;line3*

    mov ah,09h

    mov dx,offset line3

    int 21h

*;\n*

    mov ah,02h

    mov cl,0Ah

    mov dl,cl

    int 21h

*;line4*

    mov ah,09h

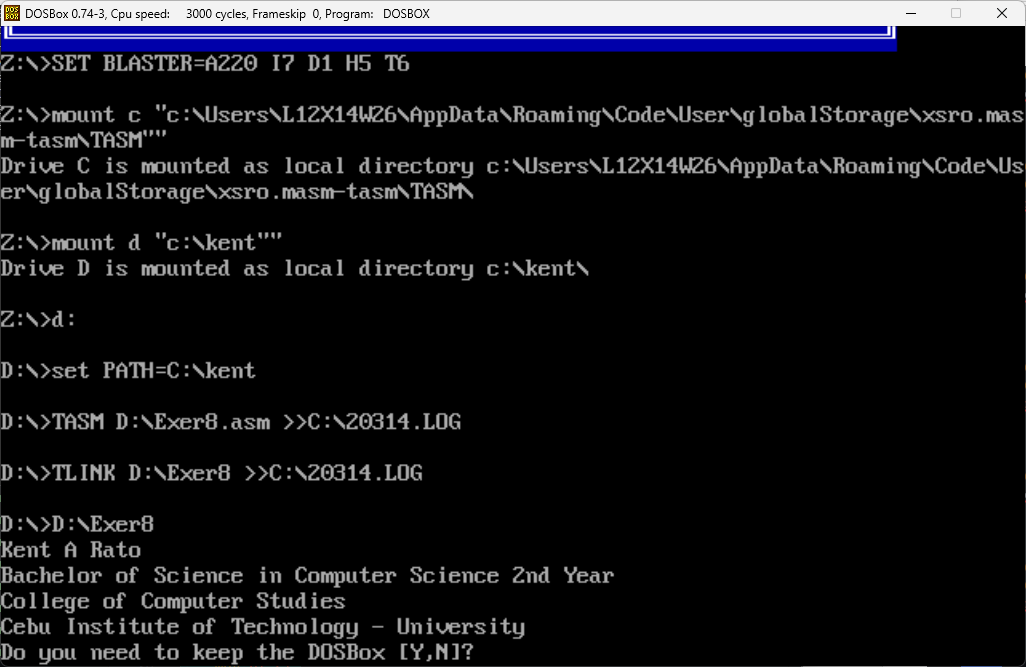
    mov dx,offset line4

    int 21h

int 27h

end

* + RUN SCREENSHOT



1. EXER9.ASM
   * CODE

*;Filename: EXER9.ASM*

*;Displaying  calendar*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.stack 100h

.data

    CALENDAR      db '\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_', 13, 10

                  db '|                      AUGUST 2024                      |', 13, 10

                  db '|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|', 13, 10

                  db '|  MON  |  TUE  |  WED  |  THU  |  FRI  |  SAT  |  SUN  |', 13, 10

                  db '|     29|     30|     31|      1|      2|      3|      4|', 13, 10

                  db '|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|', 13, 10

                  db '|      5|      6|      7|      8|      9|     10|     11|', 13, 10

                  db '|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|', 13, 10

                  db '|     12|     13|     14|     15|     16|     17|     18|', 13, 10

                  db '|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|', 13, 10

                  db '|     19|     20|     21|     22|     23|     24|     25|', 13, 10

                  db '|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|', 13, 10

                  db '|     26|     27|     28|     29|     30|     31|      1|', 13, 10

                  db '|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|\_\_\_\_\_\_\_|', 13, 10, '$'

.code

print proc

    mov ah, 09h

    int 21h

    ret

print endp

start:

    mov  ax, @data

    mov  ds, ax

    mov ax, 3

    int 10h

    lea dx, CALENDAR

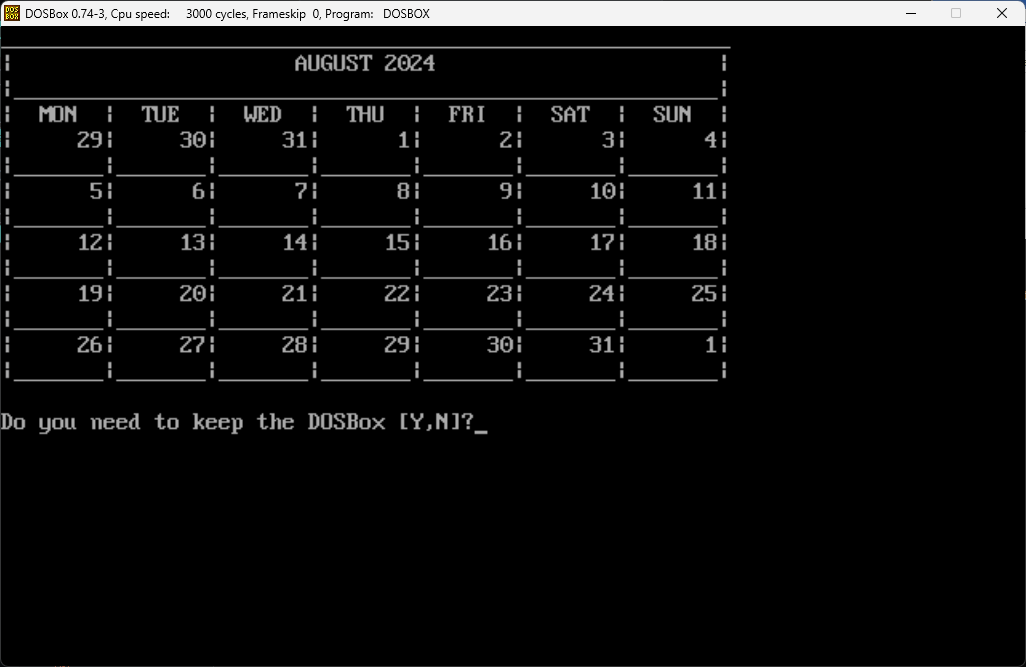
    call print

    mov ax, 4C00h

    int 21h

end start

* + RUN SCREENSHOT



1. EXER10.ASM
   * CODE

*;Filename: EXER8.ASM*

*;Displaying strings of text*

*;KENT A. RATO*

*;Date: August 30, 2024*

.model small

.stack 100h

.data

    stitle        db  'College enrollment form$'

    sName         db  'First name             Last name$'

    sAddress      db  'Street Address$'

    slocation     db  'City                   State$'

    sBday         db  'Month           Day              Year$'

.code

*; Procedure to print a single character*

printChar proc

    mov ah, 02h          *; DOS function to print character*

    int 21h              *; Call DOS interrupt*

    ret                  *; Return from procedure*

printChar endp

*; Procedure to print a character a specific number of times*

printCharNTimes proc

*; Parameters:*

*;   DL - character to print*

*;   CX - number of times to print*

printLoop:

    call printChar       *; Print the character*

    loop printLoop       *; Repeat until CX is zero*

    ret                  *; Return from procedure*

printCharNTimes endp

start:

*; Initialize data segment*

    mov ax, @data

    mov ds, ax

*; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Title*

    mov  ah,09h

    mov  dx,offset stitle

    int  21h

*;\n*

    mov dl, 0Ah

    call printChar

*;Name*

    mov  ah,09h

    mov  dx,offset sName

    int  21h

*;\n*

    mov dl, 0Ah

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*;Print Space*

    mov dl, 20h

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*; Print newline*

    mov dl, 0Ah

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;Space*

    mov dl, 20h

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;\n*

    mov dl, 0Ah

    call printChar

*; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Address*

    mov  ah,09h

    mov  dx,offset sAddress

    int  21h

*;\n*

    mov dl, 0Ah

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 43

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*;\n*

    mov dl, 0Ah

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 43

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;\n*

    mov dl, 0Ah

    call printChar

*; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> location*

    mov  ah,09h

    mov  dx,offset slocation

    int  21h

*;\n*

    mov dl, 0Ah

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*;Print Space*

    mov dl, 20h

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*; Print newline*

    mov dl, 0Ah

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;Space*

    mov dl, 20h

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 20

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;\n*

    mov dl, 0Ah

    call printChar

*; >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Birthday*

    mov  ah,09h

    mov  dx,offset sBday

    int  21h

*;\n*

    mov dl, 0Ah

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*;Print Space*

    mov dl, 20h

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*;Print Space*

    mov dl, 20h

    call printChar

*; Print ┌ (218)*

    mov dl, 218

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┐ (191)*

    mov dl, 191

    call printChar

*; Print newline*

    mov dl, 0Ah

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;Space*

    mov dl, 20h

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;Space*

    mov dl, 20h

    call printChar

*; Print └ (192)*

    mov dl, 192

    call printChar

*; Print loop char*

    mov dl, 196

    mov cx, 13

    call printCharNTimes

*; Print ┘ (217)*

    mov dl, 217

    call printChar

*;\n*

    mov dl, 0Ah

    call printChar

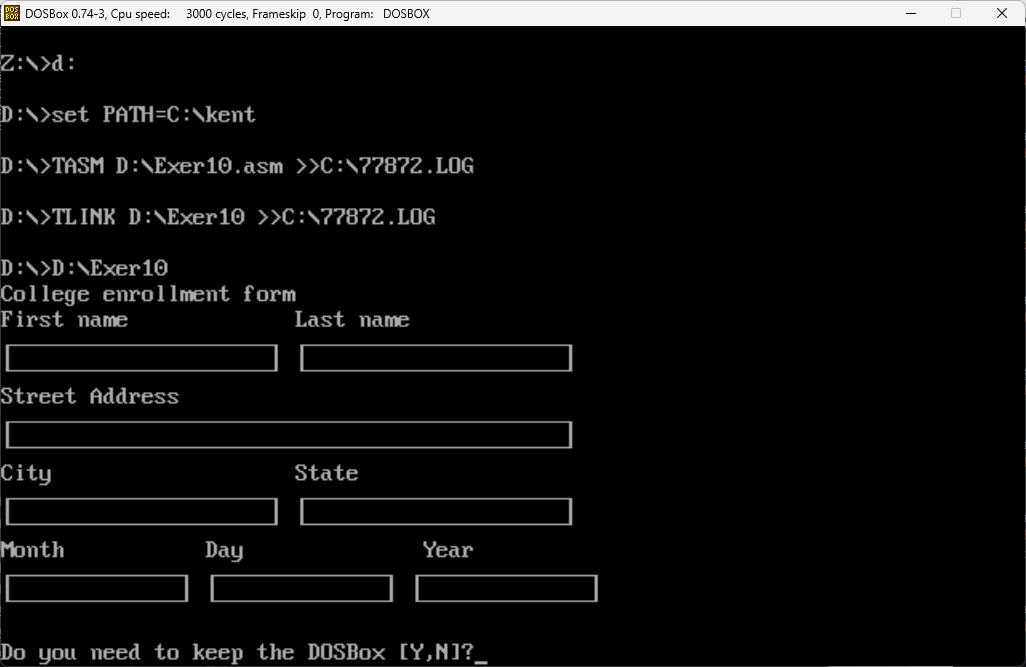
*; Terminate program*

    mov ah, 4Ch

    int 21h

end start

* + RUN SCREENSHOT



1. EXER11.ASM
   * CODE

*;Filename: EXER11.ASM*

*; make an atm machine*

*;KENT A. RATO*

*;Date: August 30, 2024*

.MODEL SMALL

.STACK 100

.DATA

TEXT    db '   Log Out '

        db 13, 10, 10, '                        Dear Customers, Welcome to Bank X             ', 13, 10, 10

        db 13, 10, 10, '        <    Withdrawal                                   Payment    >', 13, 10, 10

        db 13, 10, 10, '        <    Balance Inquiry                   Recent Transaction    >', 13, 10, 10

        db 13, 10, 10, '        <    Transfer                      Credit Card Operations    >', 13, 10, 10

        db 13, 10, 10, '        <    Change PIN                                   Deposit    >', 13, 10, 10, 10, 10, '$'

.CODE

START:

    mov ax, 3

    int 10h

    mov ah, 06h

    xor al, al

    xor cx, cx

    mov ch, 1

    mov cl, 0

    mov dh, 22

    mov dl, 80

    mov bh, 17h

    int 10h

    mov ch, 1

    mov cl, 0

    mov dh, 3

    mov dl, 80

    mov bh, 31h

    int 10h

    mov ch, 0

    mov cl, 0

    mov dh, 0

    mov dl, 80

    mov bh, 71h

    int 10h

    mov ch, 6

    mov cl, 10

    mov dh, 6

    mov dl, 32

    mov bh, 31h

    int 10h

    mov ch, 6

    mov cl, 7

    mov dh, 6

    mov dl, 9

    mov bh, 71h

    int 10h

    mov ch, 6

    mov cl, 41

    mov dh, 6

    mov dl, 67

    mov bh, 31h

    int 10h

    mov ch, 6

    mov cl, 68

    mov dh, 6

    mov dl, 70

    mov bh, 71h

    int 10h

    mov ch, 10

    mov cl, 10

    mov dh, 10

    mov dl, 32

    mov bh, 31h

    int 10h

    mov ch, 10

    mov cl, 7

    mov dh, 10

    mov dl, 9

    mov bh, 71h

    int 10h

    mov ch, 10

    mov cl, 41

    mov dh, 10

    mov dl, 67

    mov bh, 31h

    int 10h

    mov ch, 10

    mov cl, 68

    mov dh, 10

    mov dl, 70

    mov bh, 71h

    int 10h

    mov ch, 14

    mov cl, 10

    mov dh, 14

    mov dl, 32

    mov bh, 31h

    int 10h

    mov ch, 14

    mov cl, 7

    mov dh, 14

    mov dl, 9

    mov bh, 71h

    int 10h

    mov ch, 14

    mov cl, 41

    mov dh, 14

    mov dl, 67

    mov bh, 31h

    int 10h

    mov ch, 14

    mov cl, 68

    mov dh, 14

    mov dl, 70

    mov bh, 71h

    int 10h

    mov ch, 18

    mov cl, 10

    mov dh, 18

    mov dl, 32

    mov bh, 31h

    int 10h

    mov ch, 18

    mov cl, 7

    mov dh, 18

    mov dl, 9

    mov bh, 71h

    int 10h

    mov ch, 18

    mov cl, 41

    mov dh, 18

    mov dl, 67

    mov bh, 31h

    int 10h

    mov ch, 18

    mov cl, 68

    mov dh, 18

    mov dl, 70

    mov bh, 71h

    int 10h

    mov ax, @data

    mov ds, ax

    mov ah, 09h

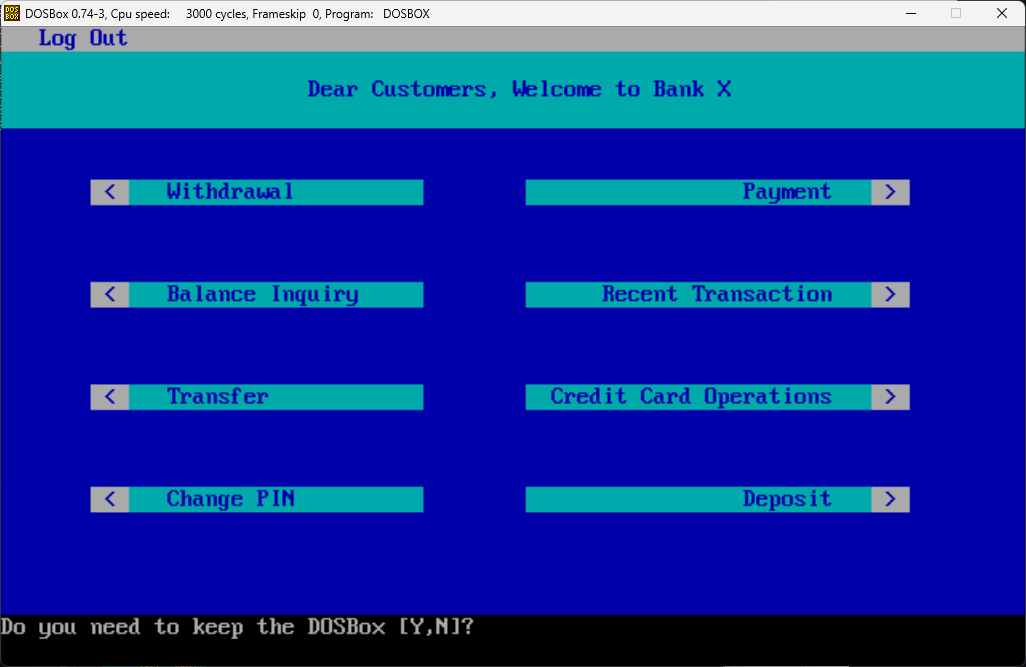
    lea dx, text

    int 21h

int 27h

END START

* + RUN SCREENSHOT



1. EXER12.ASM
   * CODE

*; Filename: EXER12.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBAR 20, 2024*

*; Description: This assembly language program will display multiple string variables on a single line.*

.model small

.stack 200h

.data

    String1 db 'Line 1', '$'

    String2 db 'Line 2', '$'

    String3 db 'Line 3', '$'

.code

ProgramStart:

    mov ax,@data

    mov ds,ax

    mov ah,9

    mov dx, offset String1

    int 21h

    mov dx, offset String2

    int 21h

    mov dx, offset String3

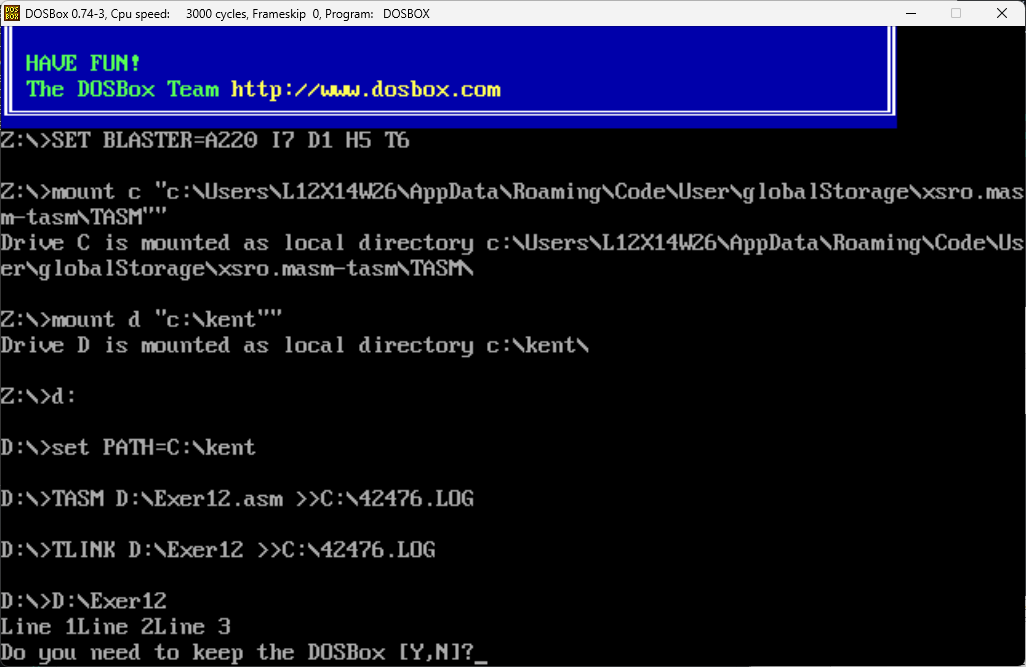
    int 21h

    mov ah,4Ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER13.ASM
   * CODE

*; Filename: EXER13.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBAR 20, 2024*

*; Description: This assembly language program will display multiple string variables on separate lines.*

.model small

.stack 200h

.data

    String1 db 'Line 1', 0dh,0ah,'$'

    String2 db 'Line 2', 0dh,0ah,'$'

    String3 db 'Line 3', 0dh,0ah,'$'

*; carriage return (ASCII code 0Dh) positions the cursor to the left side of the current line of characters*

*; line feed (ASCII code 0Ah) moves the cursor down one line on the output device.*

.code

ProgramStart:

    mov ax,@data

    mov ds,ax

    mov ah,9

    mov dx, offset String1

    int 21h

    mov dx, offset String2

    int 21h

    mov dx, offset String3

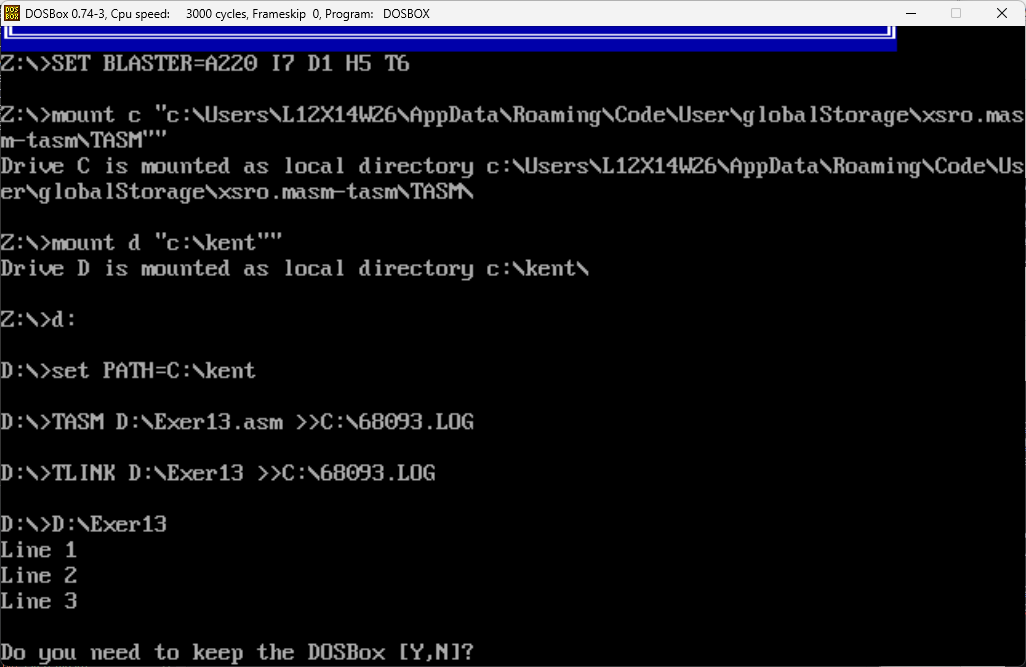
    int 21h

    mov ah,4Ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER14.ASM
   * CODE

*; Filename: EXER14.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: This assembly language program will get character input and*

*; display back character input.*

.model small

.stack 200h

.data

    inputChar db ?

    inputMsg db 'Enter a character: $'

    outputMsg db 'You entered: $'

.code

Main:

    mov ax,@data

    mov ds,ax

    mov dx, offset inputMsg

    mov ah, 09h

    int 21h

    mov ah, 01h

    int 21h

    mov inputChar, al

    mov dx,offset outputMsg

    mov ah, 09h

    int 21h

    mov dl, inputChar

    mov ah,02h

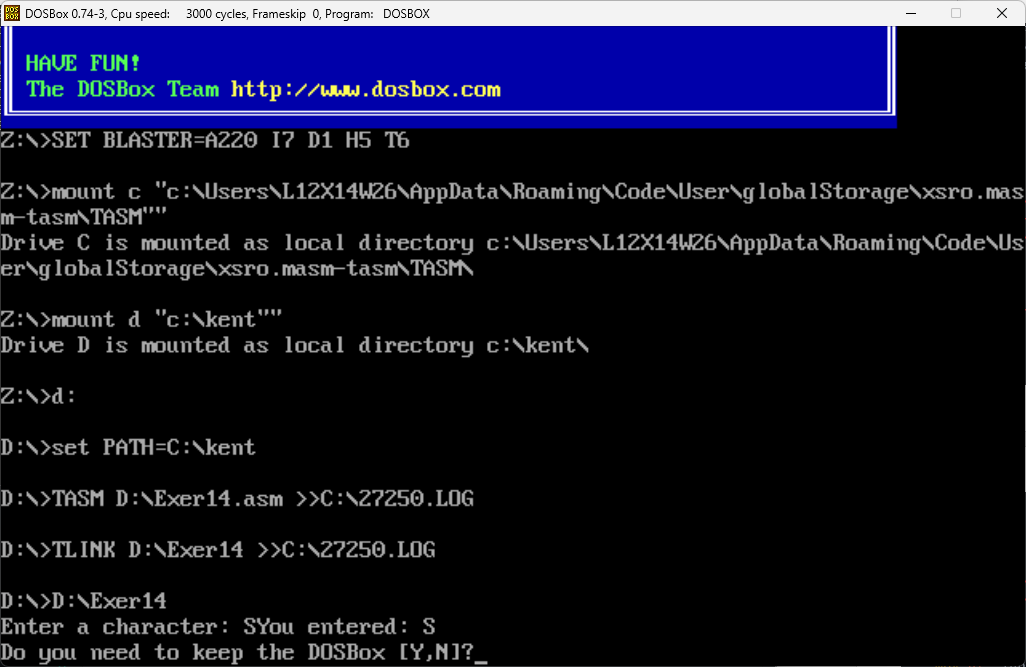
    int 21h

    mov ax,4c00h

    int 21h

end Main

* + RUN SCREENSHOT



1. EXER15.ASM
   * CODE

*; Filename: EXER15.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: This assembly language program will continuously get*

*; character input and display back input until Enter key is pressed.*

.model small

.stack 200h

.data

.code

ProgramStart:

    EchoLoop:

        mov ah,1 *; dos keyboard input*

        int 21h *; get the next key*

        cmp al,13 *; was the key Enter Key?*

        js EchoDone *; yes, so we're done echoing*

        mov dl, al *; put the character intp DL*

        mov ah,2 *; dos display output function*

        int 21h *; display the character*

        jmp EchoLoop *; echi next character*

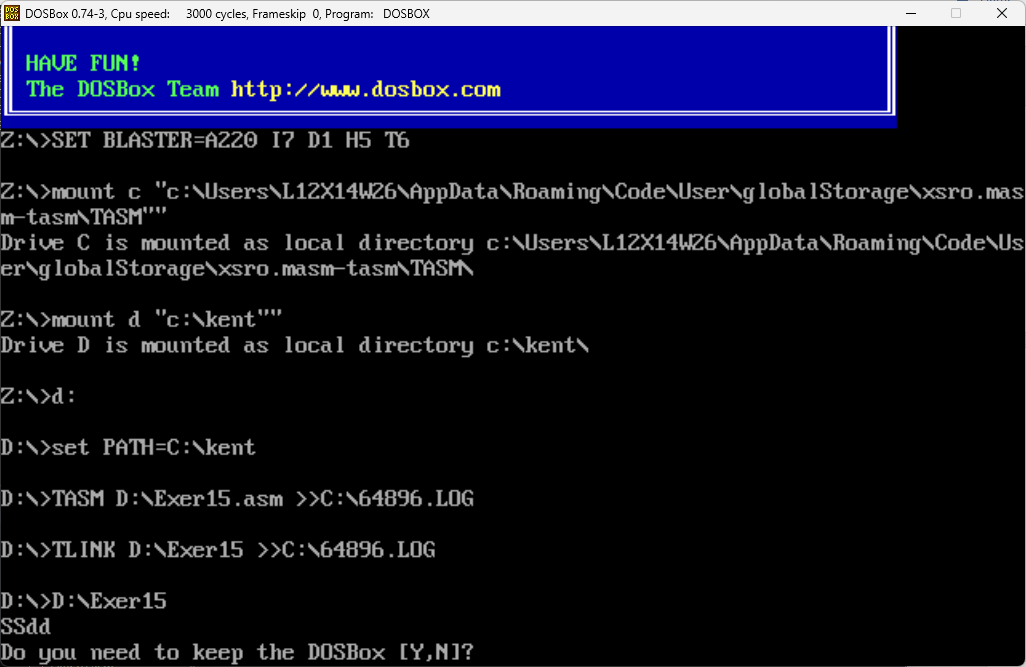
    EchoDone:

        mov ah,4ch *;dos terminate program*

        int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER16.ASM
   * CODE

*; Filename: EXER16.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: This assembly language program will get character input and*

*; and determine if input is 'y' or 'Y' to display good morning message*

*; else display good afternoon message.*

.model small

.stack 200h

.data

    TimePrompt db 'Is it after 12 noon (Y/N)?$'

    GoodMorningMessage label byte

    db 13,10,'Good morning, world!',13,10,'$'

    GoodAfternoonMessage label byte

    db 13,10,'Good afternoon, world!',13,10,'$'

.code

ProgramStart:

    mov ax, @data

    mov ds, ax

    mov dx,offset TimePrompt

    mov ah,9

    int 21h

    mov ah,1

    int 21h

    cmp al, 'y'

    jz IsAfternoon

    cmp al, 'Y'

    jnz IsMorning

IsAfternoon:

    mov dx, offset GoodAfternoonMessage

    jmp DisplayGreeting

IsMorning:

    mov dx, offset GoodMorningMessage

DisplayGreeting:

    mov ah,9

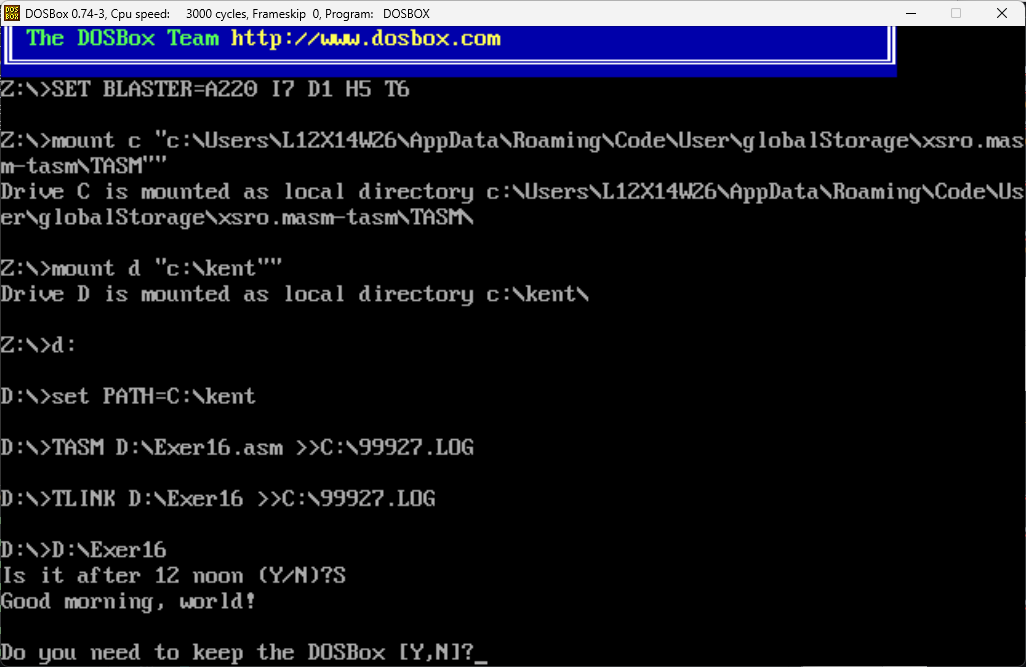
    int 21h

    mov ah,4ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER17.ASM
   * CODE

*; Filename: EXER17.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: This assembly language program will get string input and*

*; display back string.*

.model small

.stack 100h

.data

    MAXIMUM\_STRING\_LENGTH EQU 1000

    StringInput DB MAXIMUM\_STRING\_LENGTH DUP (?)

    InputPrompt DB 'Enter text: $'

.code

ProgramStart:

    mov ax, @data

    mov ds, ax

    mov dx, offset InputPrompt

    mov ah,9

    int 21h

    mov ah, 3fh

    mov bx, 0

    mov cx, MAXIMUM\_STRING\_LENGTH

    mov dx,offset StringInput

    int 21h

    and ax, ax

    js Done

    mov cx,ax

    push cx

    mov bx, offset StringInput

    pop cx

    mov ah,40h

    mov bx, 1

    mov dx,offset StringInput

    int 21h

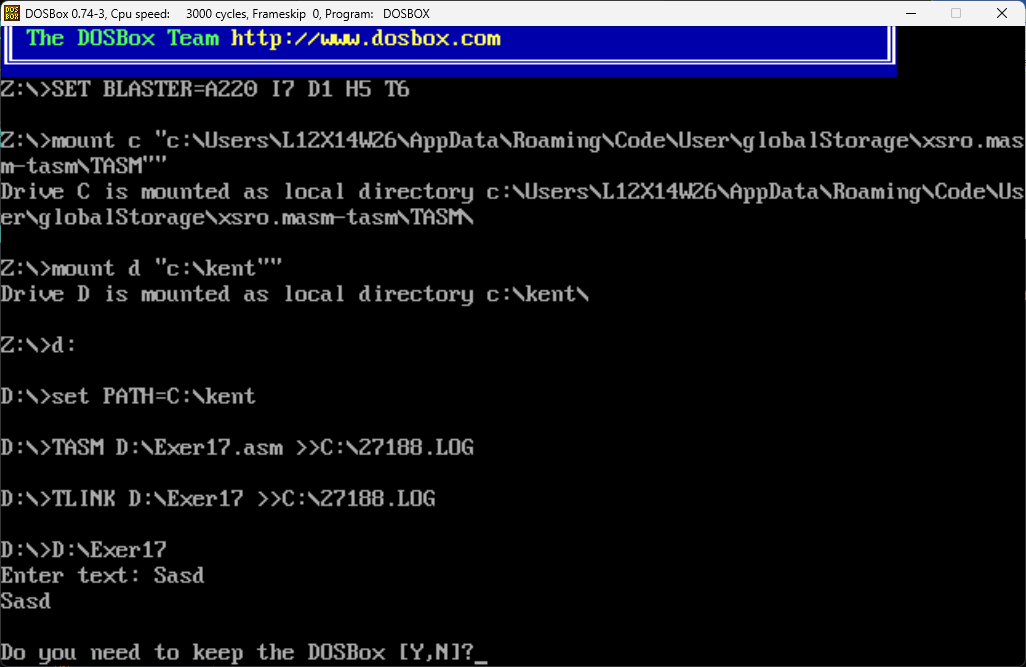
Done:

    mov ah, 4ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER18.ASM
   * CODE

*; Filename: EXER18.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: This assembly language program will get string input and*

*; display the reverse of the string.*

.model small

.stack 100h

.data

    MAXIMUM\_STRING\_LENGTH equ 1000

    StringToReverse db MAXIMUM\_STRING\_LENGTH DUP (?)

    ReverseString db MAXIMUM\_STRING\_LENGTH DUP (?)

.code

ProgramStart:

    mov ax, @data

    mov ds,ax

    mov ah, 3fh

    mov bx, 0

    mov cx, MAXIMUM\_STRING\_LENGTH

    mov dx, offset StringToReverse

    int 21h

    and ax,ax

    jz Done

    mov cx,ax

    push cx

    mov bx, offset StringToReverse

    mov si, offset ReverseString

    add si, cx

    dec si

ReverseLoop:

    mov al,[bx]

    mov [si],al

    inc bx

    dec si

    loop ReverseLoop

    pop cx

    mov ah,40h

    mov bx,1

    mov dx,offset ReverseString

    int 21h

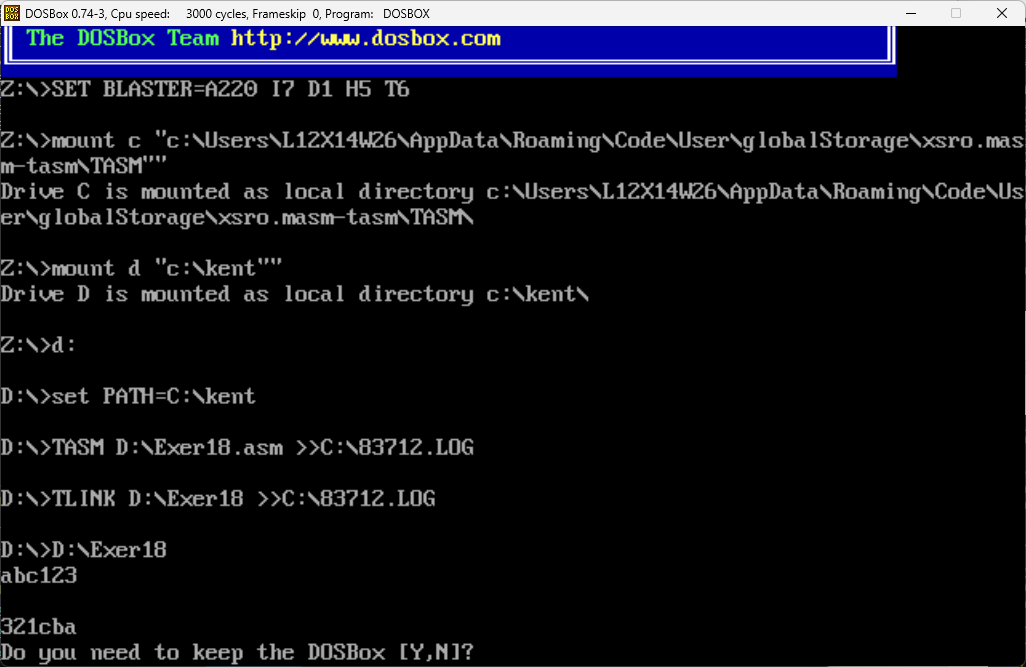
Done:

    mov ah,4ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER19.ASM
   * CODE

*; Filename: EXER19.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: Create a program that asks the user to input 3 characters. Then display each character on its own line.*

.model small

.stack 100h

.data

    prompt1 db 'Enter first character: $'

    prompt2 db 'Enter second character: $'

    prompt3 db 'Enter third character: $'

    newline db 13, 10, '$'

    output1 db 'The first character is ', '$'

    output2 db 'The second character is ', '$'

    output3 db 'The third character is ', '$'

    firstChar db ?

    secondChar db ?

    thirdChar db ?

.code

ProgramStart:

    mov ax, @data

    mov ds, ax

    mov ah, 09h

    lea dx, prompt1

    int 21h

    mov ah, 01h

    int 21h

    mov firstChar, al

    mov ah, 09h

    lea dx, newline

    int 21h

    mov ah, 09h

    lea dx, prompt2

    int 21h

    mov ah, 01h

    int 21h

    mov secondChar, al

    mov ah, 09h

    lea dx, newline

    int 21h

    mov ah, 09h

    lea dx, prompt3

    int 21h

    mov ah, 01h

    int 21h

    mov thirdChar, al

    mov ah, 09h

    lea dx, newline

    int 21h

    mov ah, 09h

    lea dx, output1

    int 21h

    mov dl, firstChar

    mov ah, 02h

    int 21h

    mov ah, 09h

    lea dx, newline

    int 21h

    mov ah, 09h

    lea dx, output2

    int 21h

    mov dl, secondChar

    mov ah, 02h

    int 21h

    mov ah, 09h

    lea dx, newline

    int 21h

    mov ah, 09h

    lea dx, output3

    int 21h

    mov dl, thirdChar

    mov ah, 02h

    int 21h

    mov ah, 09h

    lea dx, newline

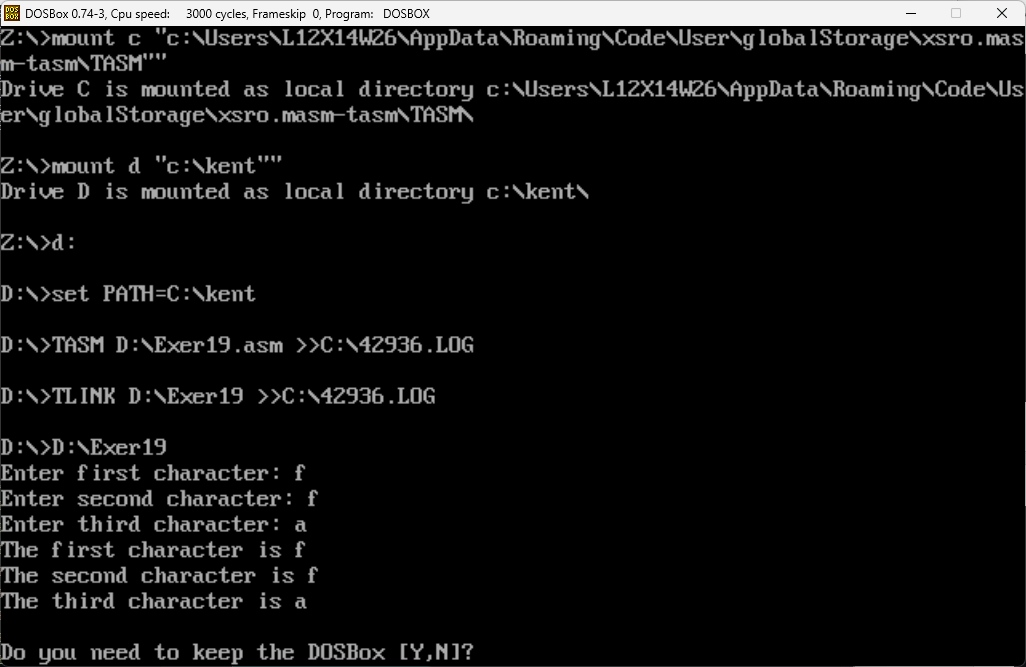
    int 21h

    mov ah, 4ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER20.ASM
   * CODE

*; Filename: EXER20.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: Create a program that asks user to input a character.*

*; Determine if character is letter A and display message “Yes, you have entered*

*; letter A.” If not, display “No, you have not entered letter A. You*

*; entered character \_.”*

.model small

.stack 200h

.data

    prompt db 'Enter a character: $'

    outputYes db 'Yes, you have entered letter A.$'

    outputNo db 'No, you have not entered letter A. You entered character: $'

    newline db 13, 10, '$'

    inputChar db ?

.code

ProgramStart:

    mov ax, @data

    mov ds, ax

    mov ah, 09h

    lea dx, prompt

    int 21h

    mov ah, 01h

    int 21h

    mov inputChar, al

    mov ah, 09h

    lea dx, newline

    int 21h

    cmp al, 'A'

    je isCharA

isNotCharA:

    mov ah, 09h

    lea dx, outputNo

    int 21h

    mov dl, inputChar

    mov ah, 02h

    int 21h

    jmp endProgram

isCharA:

    mov ah, 09h

    lea dx, outputYes

    int 21h

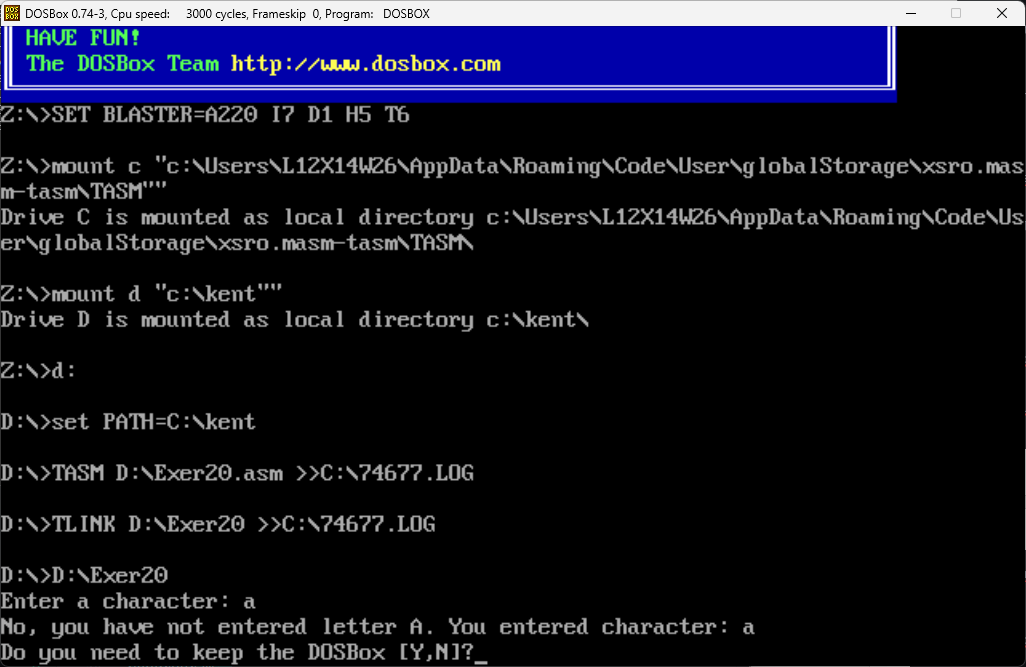
endProgram:

    mov ah, 4ch

    int 21h

end ProgramStart

* + RUN SCREENSHOT



1. EXER21.ASM
   * CODE

*; Filename: EXER21.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: Create a program that asks for your first name, middle name, and*

*; family name. Then display “Hello, FIRST NAME MIDDLE NAME FAMILY NAME!”.*

.MODEL small

.STACK 200h

.DATA

    regMsg db 'REGISTRATION FORM$', 0dh, 0ah

    firstNamePrompt db 0Dh, 0Ah,'Enter your first name: $'

    middleNamePrompt db 0Dh, 0Ah, 'Enter your middle name: $'

    familyNamePrompt db 0Dh, 0Ah, 'Enter your family name: $'

    greetingMessage db 0Dh, 0Ah, 'Hello, $'

    space db ' $'

    firstName db 20, 20 dup(?)

    middleName db 20, 20 dup(?)

    familyName db 20, 20 dup(?)

    exclamation db '!$'

.CODE

Main:

    mov ax, @data

    mov ds, ax

    mov dx, OFFSET regMsg

    mov ah, 09h

    int 21h

    mov dx, OFFSET firstNamePrompt

    mov ah, 09h

    int 21h

    lea dx, firstName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET middleNamePrompt

    mov ah, 09h

    int 21h

    lea dx, middleName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET familyNamePrompt

    mov ah, 09h

    int 21h

    lea dx, familyName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET greetingMessage

    mov ah, 09h

    int 21h

    lea si, firstName + 2

    mov cl, [firstName+1]

DisplayFirstName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayFirstName

    mov dl, ' '

    mov ah, 02h

    int 21h

    lea si, middleName + 2

    mov cl, [middleName+1]

DisplayMiddleName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayMiddleName

    mov dl, ' '

    mov ah, 02h

    int 21h

    lea si, familyName + 2

    mov cl, [familyName+1]

DisplayFamilyName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayFamilyName

    mov dl, '!'

    mov ah, 02h

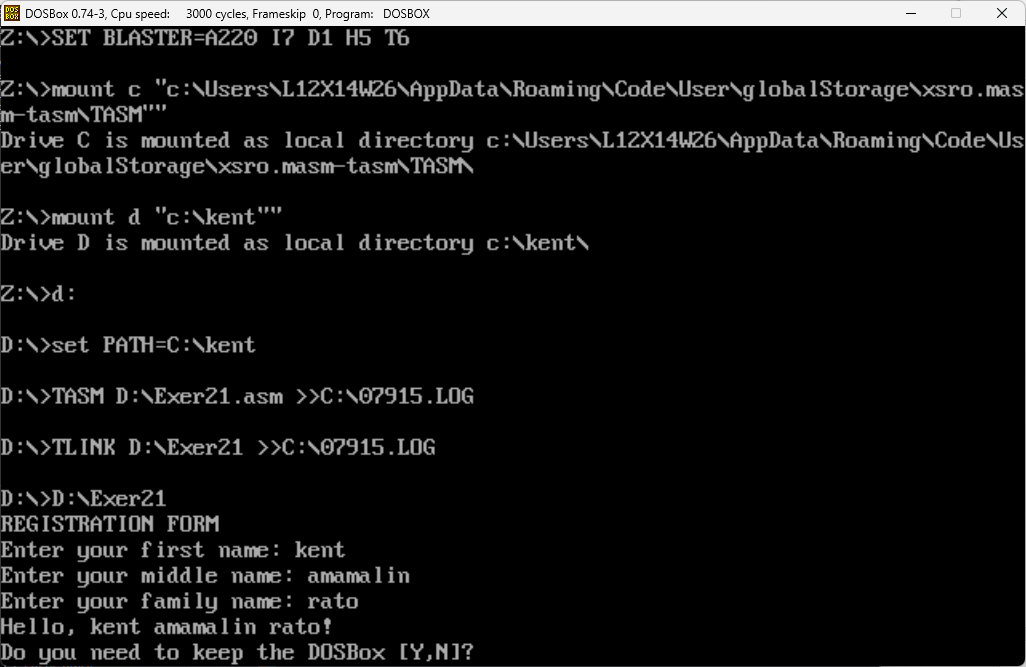
    int 21h

    mov ax, 4C00h

    int 21h

END Main

* + RUN SCREENSHOT



1. EXER3.ASM
   * CODE

*; Filename: EXER22.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: Create a College Enrollment Form. Get user input. Display a summary*

*; of the inputs. Include the necessary documentation as comments in*

*; your code.*

.MODEL small

.STACK 200h

.DATA

    header db 'Cebu Institute of Technology – University', 0dh, 0ah, 'STUDENT ENROLLMENT FORM', 0dh, 0ah, '$'

    studentIdPrompt db 0Dh, 0Ah,'Student ID Number: $'

    firstNamePrompt db 0Dh, 0Ah,'Enter your first name: $'

    middleNamePrompt db 0Dh, 0Ah, 'Enter your middle name: $'

    familyNamePrompt db 0Dh, 0Ah, 'Enter your family name: $'

    regMsg db 'REGISTRATION FORM$', 0dh, 0ah

    greetingMessage db 0Dh, 0Ah, 'Hello, $'

    space db ' $'

    firstName db 20, 20 dup(?)

    middleName db 20, 20 dup(?)

    familyName db 20, 20 dup(?)

    exclamation db '!$'

.CODE

Main:

    mov ax, @data

    mov ds, ax

    mov dx, OFFSET header

    mov ah, 09h

    int 21h

    mov dx, OFFSET firstNamePrompt

    mov ah, 09h

    int 21h

    lea dx, firstName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET middleNamePrompt

    mov ah, 09h

    int 21h

    lea dx, middleName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET familyNamePrompt

    mov ah, 09h

    int 21h

    lea dx, familyName

    mov ah, 0Ah

    int 21h

    mov dx, OFFSET greetingMessage

    mov ah, 09h

    int 21h

    lea si, firstName + 2

    mov cl, [firstName+1]

DisplayFirstName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayFirstName

    mov dl, ' '

    mov ah, 02h

    int 21h

    lea si, middleName + 2

    mov cl, [middleName+1]

DisplayMiddleName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayMiddleName

    mov dl, ' '

    mov ah, 02h

    int 21h

    lea si, familyName + 2

    mov cl, [familyName+1]

DisplayFamilyName:

    mov dl, [si]

    mov ah, 02h

    int 21h

    inc si

    loop DisplayFamilyName

    mov dl, '!'

    mov ah, 02h

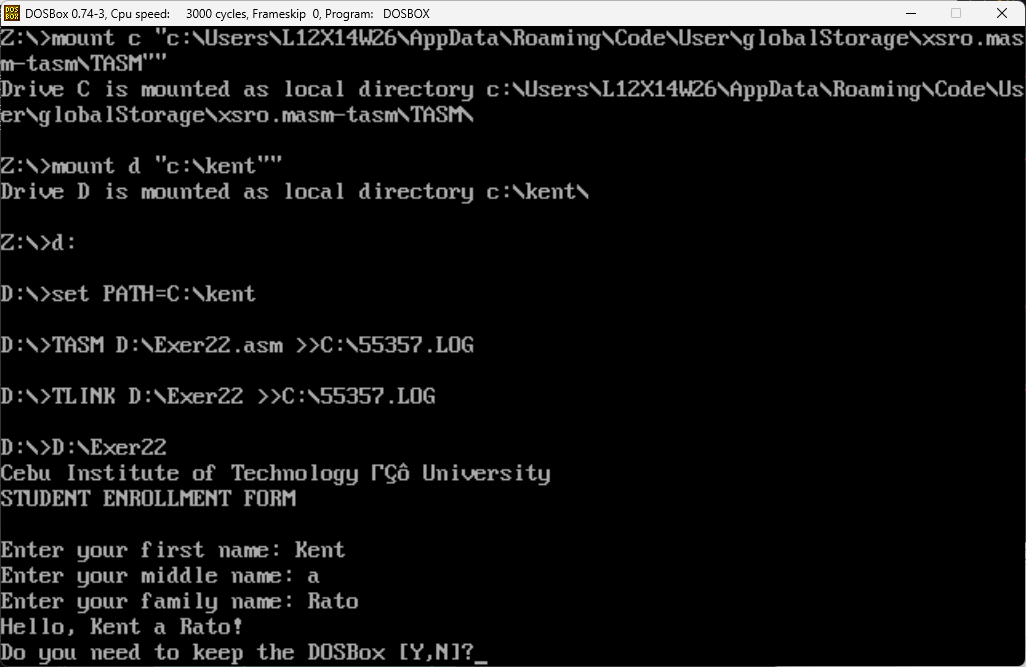
    int 21h

    mov ax, 4C00h

    int 21h

END Main

* + RUN SCREENSHOT



1. EXER23.ASM
   * CODE

*; Filename: EXER23.ASM*

*; Programmer Name: KENT A. RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description:*

.MODEL SMALL

.STACK 100h

.DATA

    menu db 'Cebu Institute of Technology - University',0ah,'$'

    menu1 db 'Automated Teller Machine',0ah,0ah, '$'

    menu2 db 'Main Menu',0ah,'$'

    menu3 db '1 Balance Inquiry',0ah,'$'

    menu4 db '2 Withdrawal',0ah,'$'

    menu5 db '3 Deposit',0ah,'$'

    menu6 db '4 Transfer',0ah,'$'

    menu7 db '5 Bills Payment',0ah,'$'

    menu8 db '6 Change Pin',0ah,'$'

    menu9 db '7 Exit',0ah,'$'

    menu10 db 'Enter number of your choice: $'

    thanks db 0ah,'Thank you for banking with us!',0ah,'$'

    thanks2 db 'Copyright 2024',0ah,'$'

    thanks3 db 'Programmer: KENT A. RATO',0ah,'$'

    msg1 db 0ah,0ah,'You have chosen 1 Balance Inquiry',0ah,'$'

    msg2 db 0ah,0ah,'You have chosen 2 Withdrawal',0ah,'$'

    msg3 db 0ah,0ah,'You have chosen 3 Deposit',0ah,'$'

    msg4 db 0ah,0ah,'You have chosen 4 Transfer',0ah,'$'

    msg5 db 0ah,0ah,'You have chosen 5 Bills Payment',0ah,'$'

    msg6 db 0ah,0ah,'You have chosen 6 Change Pin',0ah,'$'

    msg7 db 0ah,0ah,'You have chosen 7 Exit',0ah,'$'

.CODE

MAIN PROC

*; Initialize data segment*

    MOV AX, @DATA

    MOV DS, AX

*; Display menus*

    LEA DX, menu

    MOV AH, 09h

    INT 21h

    LEA DX, menu1

    MOV AH, 09h

    INT 21h

    LEA DX, menu2

    MOV AH, 09h

    INT 21h

    LEA DX, menu3

    MOV AH, 09h

    INT 21h

    LEA DX, menu4

    MOV AH, 09h

    INT 21h

    LEA DX, menu5

    MOV AH, 09h

    INT 21h

    LEA DX, menu6

    MOV AH, 09h

    INT 21h

    LEA DX, menu7

    MOV AH, 09h

    INT 21h

    LEA DX, menu8

    MOV AH, 09h

    INT 21h

    LEA DX, menu9

    MOV AH, 09h

    INT 21h

    LEA DX, menu10

    MOV AH, 09h

    INT 21h

*; Get user input (single character)*

    MOV AH, 01h

    INT 21h

*; Compare and jump to appropriate section*

    CMP AL, '1'

    JE Is1

    CMP AL, '2'

    JE Is2

    CMP AL, '3'

    JE Is3

    CMP AL, '4'

    JE Is4

    CMP AL, '5'

    JE Is5

    CMP AL, '6'

    JE Is6

    CMP AL, '7'

    JE Is7

*; If no valid choice is made, exit program*

Is1:

    MOV DX, OFFSET msg1

    JMP DisplayGreeting

Is2:

    MOV DX, OFFSET msg2

    JMP DisplayGreeting

Is3:

    MOV DX, OFFSET msg3

    JMP DisplayGreeting

Is4:

    MOV DX, OFFSET msg4

    JMP DisplayGreeting

Is5:

    MOV DX, OFFSET msg5

    JMP DisplayGreeting

Is6:

    MOV DX, OFFSET msg6

    JMP DisplayGreeting

Is7:

    MOV DX, OFFSET msg7

    JMP DisplayGreeting

DisplayGreeting:

    MOV AH, 09h       *; DOS print string function*

    INT 21h           *; display the greeting*

*; Display thank you messages*

    LEA DX, thanks

    MOV AH, 09h

    INT 21h

    LEA DX, thanks2

    MOV AH, 09h

    INT 21h

    LEA DX, thanks3

    MOV AH, 09h

    INT 21h

*; Exit program*

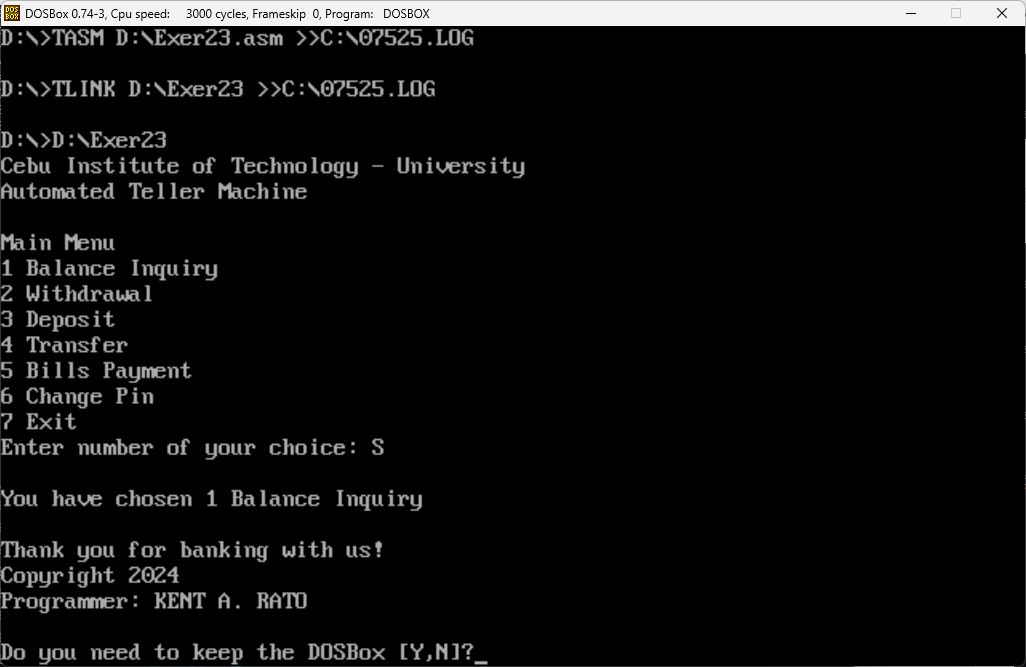
    MOV AH, 4Ch       *; DOS terminate program function*

    INT 21h

MAIN ENDP

END MAIN

* + RUN SCREENSHOT



1. EXER24.ASM
   * CODE

*; Filename: EXER24.ASM*

*; Programmer Name: KENT RATO*

*; Date: SEPTEMBER 20, 2024*

*; Description: Edit your Laboratory Prelim Hands-on Exam. Before the form displays,*

*; ask the user to input all the needed data. Afterwards, display the*

*; form with all the data entered by the user.*

.model small

.stack 100h

.data

menu    db '                                                                         ', 13, 10

        db '                   Cebu Institute of Technology - University               ', 13, 10

        db '                       VEHICLE STICKER APPLICATION FORM                   ', 13, 10

        db '                        Please fill out the form below.                    ', 13, 10, 10

        db '   Personnel Type:                     V    Vehicle Sticker Type:           V ', 13, 10, 10

        db '   Name of Applicant/Driver:                ID Number:                      ', 13, 10, 10

        db '   Mobile Number:                           Address:                        ', 13, 10, 10

        db '   Vehicle Make(s)/Brand:                   Plate Number:                   ', 13, 10, 10

        db '   Vehicle Color:                      V    Vehicle Type:                   V ', 13, 10, 10, 10

        db '                                    SUBMIT                      ', 13, 10, 10

        db '                        Copyright 2024 Kent A. Rato      ', 13, 10, 10, 10

        db '                                  Thank You!            ', 13, 10, '$'

personnelType db 9, 0, 9 dup(0)

applicantName db 12, 0, 12 dup(0)

mobileNumber db 12, 0, 12 dup(0)

carMake db 12, 0, 12 dup(0)

carColor db 9, 0, 9 dup(0)

stickerType db 9, 0, 9 dup(0)

idNum db 12, 0, 12 dup(0)

address db 12, 0, 12 dup(0)

plateNumber db 12, 0, 12 dup(0)

carType db 9, 0, 9 dup(0)

.code

printForm proc

    xor al, al

*; Grey BG*

    mov ah, 06h

    mov ch, 1 *; row start*

    mov cl, 2 *; col start*

    mov dh, 19  *; row end*

    mov dl, 78  *; col end*

    mov bh, 70h *; grey bg with black text*

    int 10h

*; header red bg*

    mov ah, 06h

    mov ch, 1

    mov cl, 3

    mov dh, 3

    mov dl, 77

    mov bh, 4fh *; red bg with white text*

    int 10h

*; header yellow blink line*

    mov ah, 06h

    mov ch, 3

    mov cl, 3

    mov dh, 3

    mov dl, 77

    mov bh, 0ceh *; red bg with yellow blinking text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 5

    mov cl, 29

    mov dh, 5

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 5

    mov cl, 66

    mov dh, 5

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 7

    mov cl, 29

    mov dh, 7

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 7

    mov cl, 66

    mov dh, 7

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 9

    mov cl, 29

    mov dh, 9

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 9

    mov cl, 66

    mov dh, 9

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 9

    mov cl, 29

    mov dh, 9

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 9

    mov cl, 66

    mov dh, 9

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 11

    mov cl, 29

    mov dh, 11

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 11

    mov cl, 66

    mov dh, 11

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg left*

    mov ah, 06h

    mov ch, 13

    mov cl, 29

    mov dh, 13

    mov dl, 40

    mov bh, 0fh *; black bg with white text*

    int 10h

*; black bg right*

    mov ah, 06h

    mov ch, 13

    mov cl, 66

    mov dh, 13

    mov dl, 77

    mov bh, 0fh *; black bg with white text*

    int 10h

*; Upper Left V*

    mov ah, 06h

    mov ch, 5

    mov cl, 38

    mov dh, 5

    mov dl, 40

    mov bh, 4fh *; red bg with white text*

    int 10h

*; Upper Right V*

    mov ah, 06h

    mov ch, 5

    mov cl, 75

    mov dh, 5

    mov dl, 77

    mov bh, 4fh *; red bg with white text*

    int 10h

*; Lower Right V*

    mov ah, 06h

    mov ch, 13

    mov cl, 38

    mov dh, 13

    mov dl, 40

    mov bh, 4fh *; red bg with white text*

    int 10h

*; Lower Left V*

    mov ah, 06h

    mov ch, 13

    mov cl, 75

    mov dh, 13

    mov dl, 77

    mov bh, 4fh *; red bg with white text*

    int 10h

*; Red BG Submit Button*

    mov ah, 06h

    mov ch, 16

    mov cl, 35

    mov dh, 16

    mov dl, 42

    mov bh, 4eh *; red bg with yellow text*

    int 10h

*; Blinking Yellow thankyou*

    mov ah, 06h

    mov ch, 21

    mov cl, 0

    mov dh, 21

    mov dl, 78

    mov bh, 8eh

    int 10h

*; print*

    mov ah, 09h

    mov dx, offset menu

    int 21h

    ret

printForm endp

main proc

    mov ax, @data *; db setup*

    mov ds, ax

    mov ah, 00h

    mov al, 03h   *; display setup*

    int 10h

    call printForm

*; personnelType*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 5         *; Row (0-based)*

    mov dl, 029          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, personnelType

    mov ah, 0Ah

    int 21h

*; applicantName*

    mov ah, 02h

    mov bh, 00h

    mov dh, 7

    mov dl, 029

    int 10h

    lea dx, applicantName

    mov ah, 0Ah

    int 21h

*; mobileNumber*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 9          *; Row (0-based)*

    mov dl, 029          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, mobileNumber

    mov ah, 0Ah

    int 21h

*; carMake*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 11          *; Row (0-based)*

    mov dl, 029          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, carMake

    mov ah, 0Ah

    int 21h

*; carColor*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 13          *; Row (0-based)*

    mov dl, 029          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, carColor

    mov ah, 0Ah

    int 21h

*; stickerType*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 5          *; Row (0-based)*

    mov dl, 066          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, stickerType

    mov ah, 0Ah

    int 21h

*; idNum*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 7          *; Row (0-based)*

    mov dl, 066          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, idNum

    mov ah, 0Ah

    int 21h

*; address*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 9         *; Row (0-based)*

    mov dl, 066          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, address

    mov ah, 0Ah

    int 21h

*; plateNumber*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 11         *; Row (0-based)*

    mov dl, 066          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, plateNumber

    mov ah, 0Ah

    int 21h

*; carType*

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 13          *; Row (0-based)*

    mov dl, 066          *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

    lea dx, carType

    mov ah, 0Ah

    int 21h

    mov ah, 02h          *; Function to set cursor position*

    mov bh, 00h          *; Page number (0 for standard screen)*

    mov dh, 20           *; Row (0-based)*

    mov dl, 80           *; Column (0-based)*

    int 10h              *; Call BIOS interrupt*

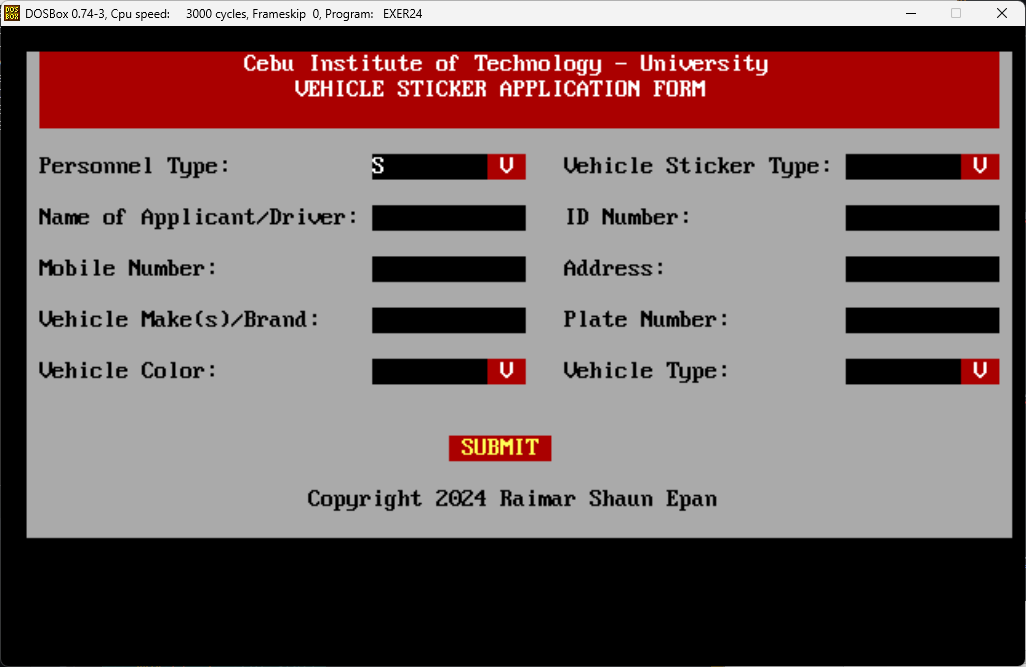
    mov ax, 4C00h *; return 0*

    int 21h

    main endp

end main

* + RUN SCREENSHOT



1. EXER25.ASM
   * CODE

*; Filename: EXER25.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description: This assembly language program will get 3 string inputs and*

*;              display back the 3 strings on separate lines.*

.model small

.stack 100h

.data

inputString1 db 50 DUP('$')  *; Reserve 50 bytes for input string*

inputString2 db 50 DUP('$')

inputString3 db 50 DUP('$')

promptString1 db 'Enter first string: $'

promptString2 db 'Enter second string: $'

promptString3 db 'Enter third string: $'

outputString1 db 'You entered first string: $'

outputString2 db 'You entered second string: $'

outputString3 db 'You entered third string: $'

.code

MAIN PROC

*; Initialize data segment*

    MOV AX, @DATA

    MOV DS, AX

*; Prompt the user for input first string*

    LEA DX, promptString1

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Read input first string*

    LEA DX, inputString1

    MOV AH, 0Ah            *; DOS function to read a string*

    INT 21h

*; Display new line*

    MOV AH, 02h

    MOV DL, 0Ah            *; 0Ah is new line*

    INT 21h

*; Prompt the user for input second string*

    LEA DX, promptString2

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Read input second string*

    LEA DX, inputString2

    MOV AH, 0Ah            *; DOS function to read a string*

    INT 21h

*; Display new line*

    MOV AH, 02h

    MOV DL, 0Ah            *; 0Ah is new line*

    INT 21h

*; Prompt the user for input third string*

    LEA DX, promptString3

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Read input third string*

    LEA DX, inputString3

    MOV AH, 0Ah            *; DOS function to read a string*

    INT 21h

*; Display new line*

    MOV AH, 02h

    MOV DL, 0Ah            *; 0Ah is new line*

    INT 21h

*; Display the output message for first string*

    LEA DX, outputString1

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Display the entered first string*

    LEA DX, inputString1 + 2  *; Skip the first two bytes (length and max length)*

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Display new line*

    MOV AH, 02h

    MOV DL, 0Ah            *; 0Ah is new line*

    INT 21h

*; Display the output message for second string*

    LEA DX, outputString2

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Display the entered second string*

    LEA DX, inputString2 + 2  *; Skip the first two bytes (length and max length)*

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Display new line*

    MOV AH, 02h

    MOV DL, 0Ah            *; 0Ah is new line*

    INT 21h

*; Display the output message for third string*

    LEA DX, outputString3

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Display the entered third string*

    LEA DX, inputString3 + 2  *; Skip the first two bytes (length and max length)*

    MOV AH, 09h            *; DOS function to display a string*

    INT 21h

*; Exit program*

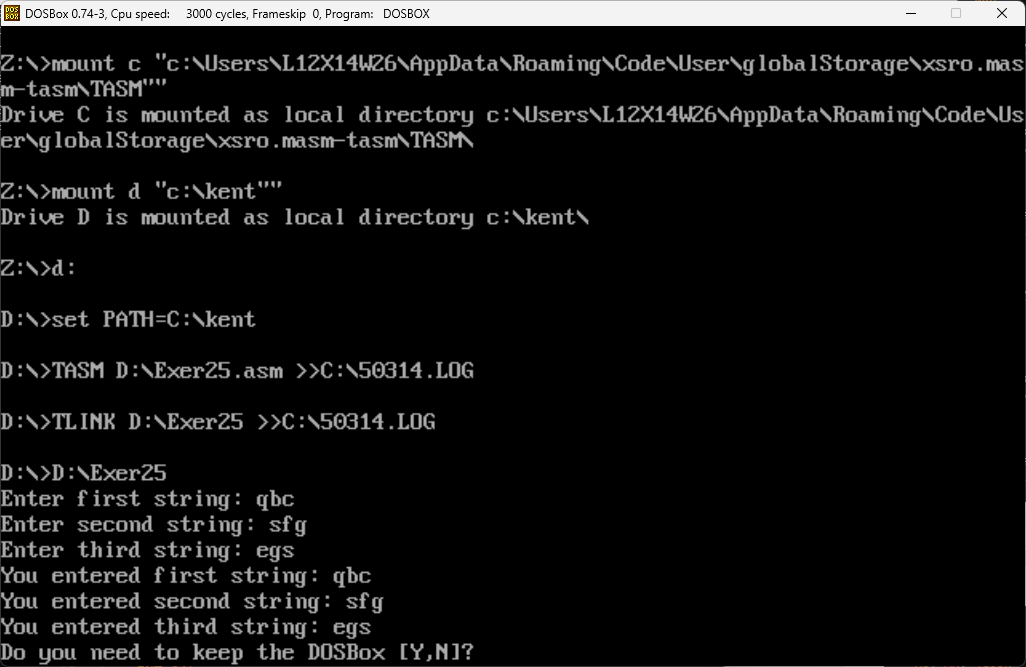
    MOV AX, 4C00h          *; DOS function to terminate program*

    INT 21h

MAIN ENDP

END MAIN

* + RUN SCREENSHOT



1. EXER26.ASM
   * CODE

*; Filename: EXER26.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description: Make a program to input two integers, add the two integers,*

*;              and display the sum of the two integers.*

.model small

.stack 100h

.data

input1 db 'Enter first integer: $'

input2 db 13,10,'Enter second integer: $'

in1 dw ?

in2 dw ?

sum dw ?

sum1 db 20 DUP ('$')

res db 13,10,'Sum: $'

.code

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0        *; Counter for digits*

    mov bx, 10       *; Base for decimal conversion*

loop1:

    xor dx, dx       *; Clear DX for division*

    div bx            *; Divide AX by 10*

    push dx          *; Push remainder onto stack*

    inc cx           *; Increment digit count*

    cmp ax, 0        *; Check if quotient is zero*

    jne loop1

loop2:

    pop dx           *; Pop remainder from stack*

    add dl, '0'      *; Convert to ASCII character*

    mov ah, 02h      *; Function to print a character*

    int 21h

    dec cx           *; Decrement digit count*

    cmp cx, 0        *; Check if all digits are printed*

    jne loop2

    pop dx

    pop cx

    pop bx

    pop ax

    ret

start:

    mov ax, @data

    mov ds, ax

*; Input first integer*

    lea dx, input1

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in1, ax

*; Input second integer*

    lea dx, input2

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in2, ax

*; Add the two integers*

    mov ax, in1

    add ax, in2

    mov sum, ax

*; Display the result*

    lea dx, res

    mov ah, 9

    int 21h

    mov ax, sum

    call itoa

    mov dx, offset sum1

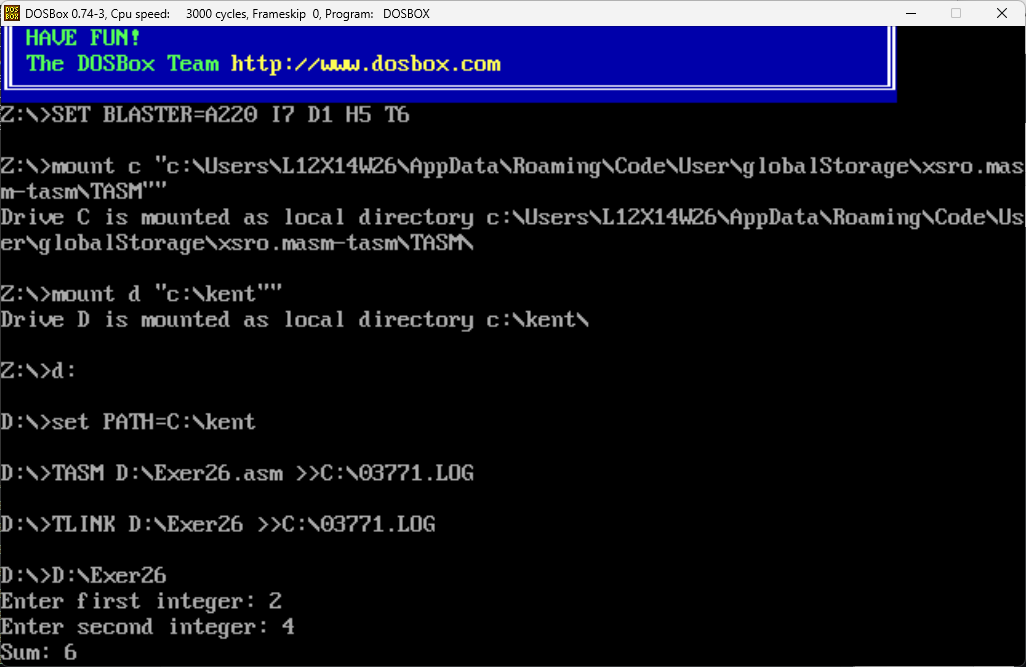
    mov ah, 9

    int 21h

    int 20h           *; Exit program*

end start

* + RUN SCREENSHOT



1. EXER27.ASM
   * CODE

*; Filename: EXER27.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description: Make a program to input two integers, subtract the second*

*;              integer from the first integer, and display the difference.*

.model small

.stack 100h

.data

input1 db 'Enter first integer (subtrahend): $'

input2 db 13,10,'Enter second integer (minuend): $'

in1 dw ?

in2 dw ?

difference dw ?

output db 13,10,'Difference: $'

.code

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0        *; Counter for digits*

    mov bx, 10       *; Base for decimal conversion*

loop1:

    xor dx, dx       *; Clear DX for division*

    div bx            *; Divide AX by 10*

    push dx          *; Push remainder onto stack*

    inc cx           *; Increment digit count*

    cmp ax, 0        *; Check if quotient is zero*

    jne loop1

loop2:

    pop dx           *; Pop remainder from stack*

    add dl, '0'      *; Convert to ASCII character*

    mov ah, 02h      *; Function to print a character*

    int 21h

    dec cx           *; Decrement digit count*

    cmp cx, 0        *; Check if all digits are printed*

    jne loop2

    pop dx

    pop cx

    pop bx

    pop ax

    ret

start:

    mov ax, @data

    mov ds, ax

*; Input first integer (subtrahend)*

    lea dx, input1

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'      *; Convert ASCII to integer*

    mov ah, 0

    mov in1, ax

*; Input second integer (minuend)*

    lea dx, input2

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'      *; Convert ASCII to integer*

    mov ah, 0

    mov in2, ax

*; Subtract the two integers*

    mov ax, in1      *; Load the first integer*

    sub ax, in2      *; Subtract the second integer*

    mov difference, ax

*; Display the result*

    lea dx, output

    mov ah, 9

    int 21h

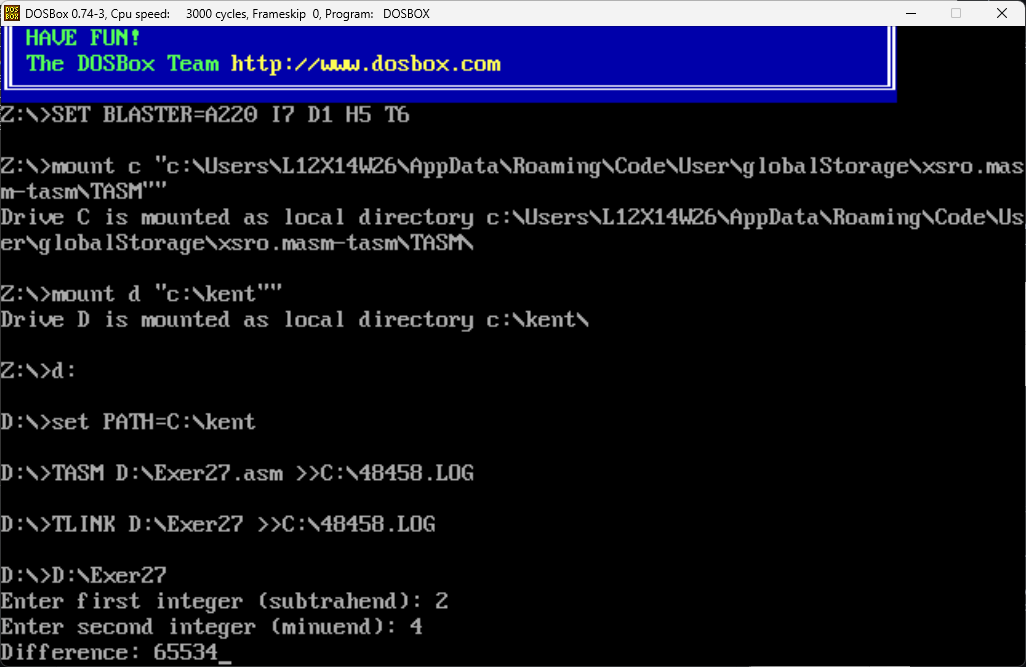
    mov ax, difference

    call itoa

    int 20h          *; Exit program*

end start

* + RUN SCREENSHOT



1. EXER28.ASM
   * CODE

*; Filename: EXER28.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description : Make a program to input two integers, multiply the two*

*; integers, and display the product of the two integers.*

.model small

.stack 100h

.data

input1 db 'Enter first integer (multiplicand): $'

input2 db 13,10,'Enter second integer (multiplier): $'

in1 dw ?

in2 dw ?

sum dw ?

sum1 db 20 DUP ('$')

res db 13,10,'Difference: $'

.code

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0          *; Counter for digits*

    mov bx, 10         *; Base for decimal conversion*

loop1:

    xor dx, dx        *; Clear DX for division*

    div bx             *; Divide AX by 10*

    push dx           *; Push remainder onto stack*

    inc cx             *; Increment digit count*

    cmp ax, 0         *; Check if quotient is zero*

    jne loop1

loop2:

    pop dx             *; Pop remainder from stack*

    add dl, '0'        *; Convert to ASCII character*

    mov ah, 02h       *; Function to print a character*

    int 21h

    dec cx             *; Decrement digit count*

    cmp cx, 0         *; Check if all digits are printed*

    jne loop2

    pop dx

    pop cx

    pop bx

    pop ax

    ret

start:

    mov ax, @data

    mov ds, ax

    lea dx, input1

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in1, ax

    lea dx, input2

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in2, ax

    mov ax, in1

    mov dx, in2

    mul dx

    mov sum, ax

    lea dx, res

    mov ah, 9

    int 21h

    mov ax, sum

    call itoa

    mov dx, offset sum1

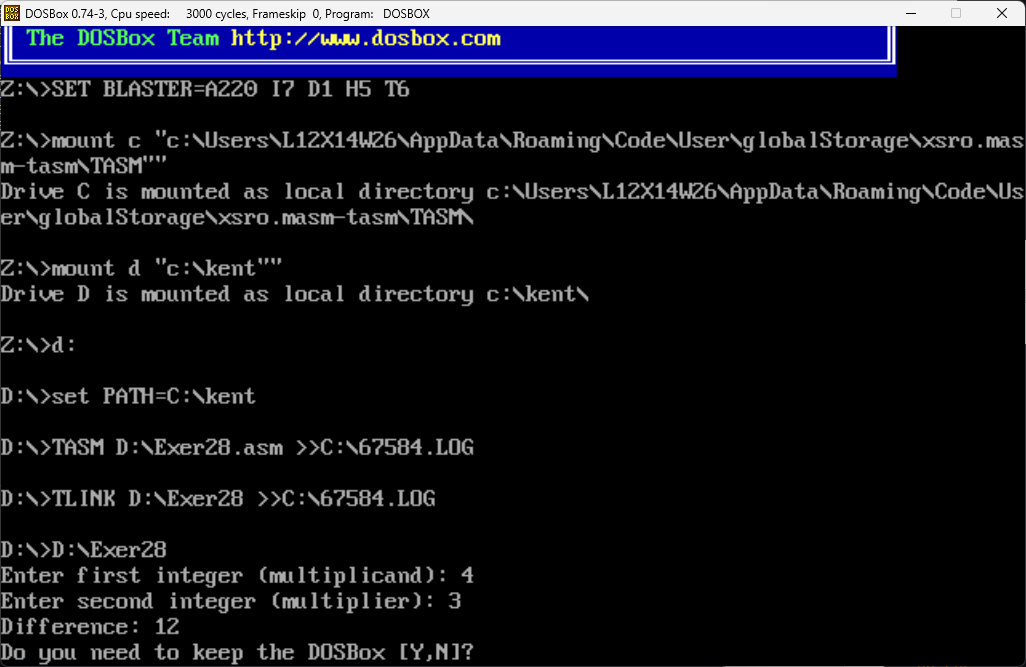
    mov ah, 9

    int 21h

    int 27h

end start

* + RUN SCREENSHOT



1. EXER29.ASM
   * CODE

*; Filename: EXER29.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description : Make a program to input two integers, divide the first by*

*; the second integer, and display the quotient of the two integers.*

.model small

.stack 100h

.data

input1 db 'Enter first integer (dividend): $'

input2 db 13,10,'Enter second integer (divisor): $'

in1 dw ?

in2 dw ?

sum dw ?

sum1 db 20 DUP ('$')

res db 13,10,'Quotient: $'

.code

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0          *; Counter for digits*

    mov bx, 10         *; Base for decimal conversion*

loop1:

    xor dx, dx        *; Clear DX for division*

    div bx             *; Divide AX by 10*

    push dx           *; Push remainder onto stack*

    inc cx             *; Increment digit count*

    cmp ax, 0         *; Check if quotient is zero*

    jne loop1

loop2:

    pop dx             *; Pop remainder from stack*

    add dl, '0'        *; Convert to ASCII character*

    mov ah, 02h       *; Function to print a character*

    int 21h

    dec cx             *; Decrement digit count*

    cmp cx, 0         *; Check if all digits are printed*

    jne loop2

    pop dx

    pop cx

    pop bx

    pop ax

    ret

start:

    mov ax, @data

    mov ds, ax

    lea dx, input1

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in1, ax

    lea dx, input2

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in2, ax

    mov ax, in1

    xor dx, dx

    mov bx, in2

    div bx

    mov sum, ax

    lea dx, res

    mov ah, 9

    int 21h

    mov ax, sum

    call itoa

    mov dx, offset sum1

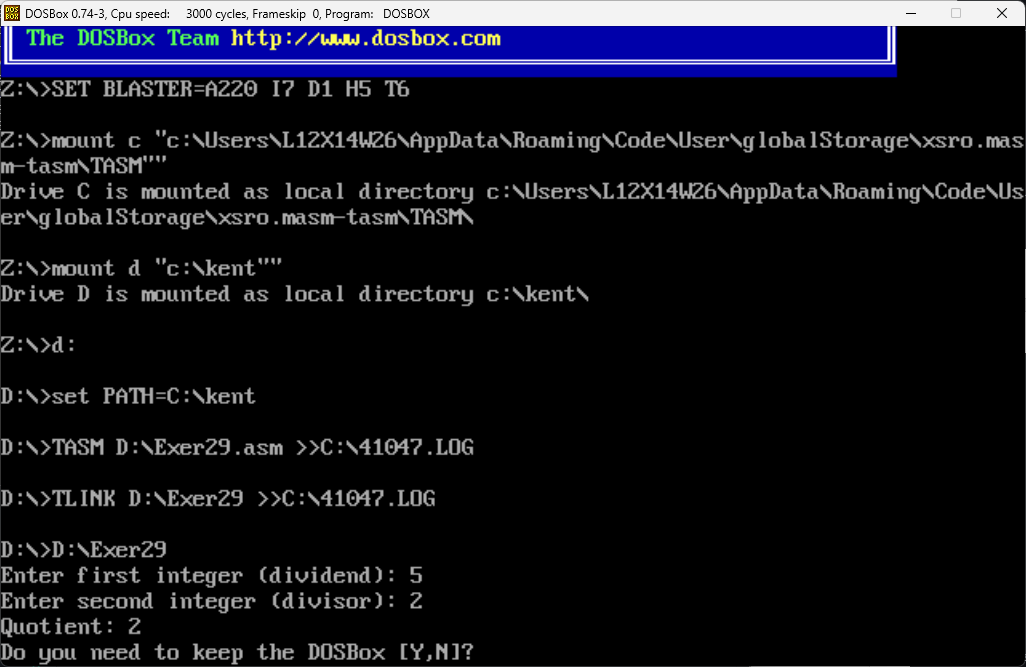
    mov ah, 9

    int 21h

    int 27h

end start

* + RUN SCREENSHOT



1. EXER30.ASM
   * CODE

*; Filename: EXER30.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description : Make a program to input two integers. Compare the two*

*; integers. Display which one is smaller and bigger of the two integers.*

.model small

.stack 100h

.data

input1 db 'Enter first integer: $'

input2 db 13,10,'Enter second integer: $'

in1 dw ?

in2 dw ?

big db 20 DUP ('$')

smol db 20 DUP ('$')

res db 13,10,'Bigger: $'

res1 db 13,10,'Smaller: $'

.code

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0          *; Counter for digits*

    mov bx, 10         *; Base for decimal conversion*

itoa\_loop1:

    xor dx, dx        *; Clear DX before dividing*

    div bx             *; AX / BX, result in AX, remainder in DX*

    push dx           *; Push remainder onto stack*

    inc cx             *; Count the number of digits*

    cmp ax, 0         *; Check if quotient is zero*

    jne itoa\_loop1    *; Repeat if not*

itoa\_loop2:

    pop dx             *; Pop the last remainder*

    add dl, '0'        *; Convert to ASCII*

    mov ah, 02h       *; Print character function*

    int 21h           *; Interrupt to print character*

    dec cx             *; Decrease the digit counter*

    cmp cx, 0         *; Check if finished printing all digits*

    jne itoa\_loop2    *; Continue if not*

    pop dx

    pop cx

    pop bx

    pop ax

    ret

start:

    mov ax, @data

    mov ds, ax

    lea dx, input1

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in1, ax

    lea dx, input2

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov in2, ax

    mov ax, in1

    mov bx, in2

    cmp ax, bx

    jg isbigger       *; Jump to isbigger if in1 > in2*

    jl islower        *; Jump to islower if in1 < in2*

isbigger:

    lea dx, res

    mov ah, 9

    int 21h

    mov ax, in1

    call itoa

    lea dx, res1

    mov ah, 9

    int 21h

    mov ax, in2

    call itoa

    jmp done

islower:

    lea dx, res

    mov ah, 9

    int 21h

    mov ax, in2

    call itoa

    lea dx, res1

    mov ah, 9

    int 21h

    mov ax, in1

    call itoa

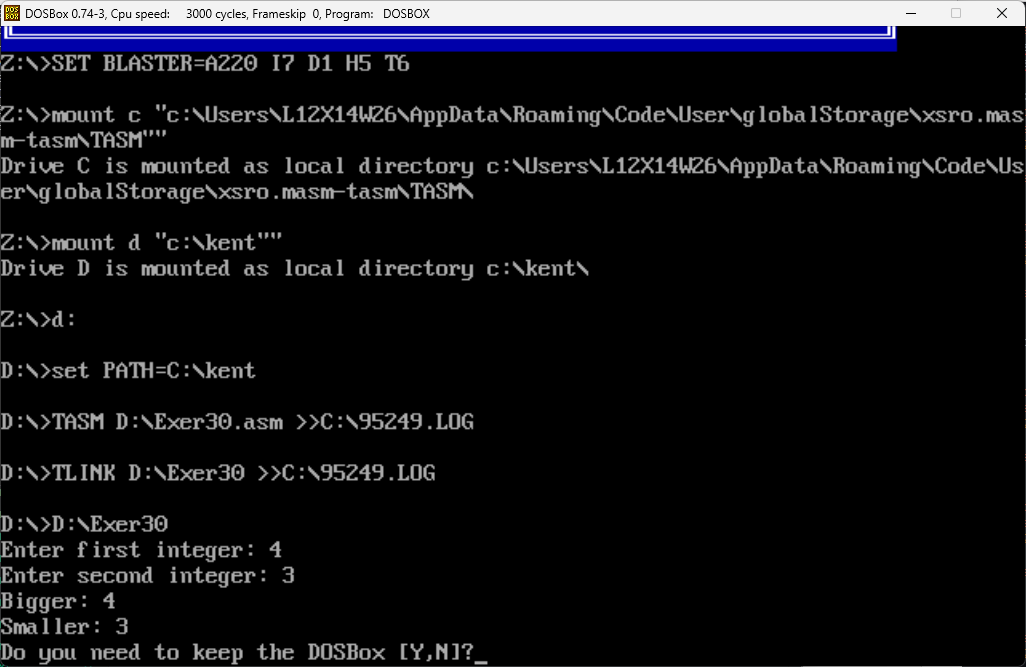
done:

    mov ah, 4Ch

    int 21h

end start

* + RUN SCREENSHOT



1. EXER31.ASM
   * CODE

*; Filename: EXER31.ASM*

*; Programmer Name: Kent A. Rato*

*; Date: September 28, 2024*

*; Description : Create a program to input an integer. Using a loop, display*

*; from 1 to the integer value entered by the user.*

.model small

.stack 100h

.data

num dw ?

prompt db 'Enter an integer: $'

.code

start:

    mov ax, @data

    mov ds, ax

    mov cx, 1           *; Initialize counter*

    lea dx, prompt

    mov ah, 9

    int 21h

    mov ah, 01h

    int 21h

    sub al, '0'

    mov ah, 0

    mov num, ax

    mov bx, num         *; Load input number into BX*

    cmp cx, bx

    jl conti            *; If CX < BX, continue*

conti:

    mov ah, 02h

    mov dl, 0ah         *; Print newline*

    int 21h

    mov ax, cx

    call itoa           *; Convert number to string and print*

    inc cx              *; Increment counter*

    cmp cx, bx

    jle conti           *; Loop if CX <= BX*

    mov ah, 4Ch

    int 21h

itoa:

    push ax

    push bx

    push cx

    push dx

    mov cx, 0          *; Counter for digits*

    mov bx, 10         *; Base for decimal conversion*

itoa\_loop1:

    xor dx, dx         *; Clear DX before dividing*

    div bx              *; AX / BX, result in AX, remainder in DX*

    push dx            *; Push remainder onto stack*

    inc cx             *; Count the number of digits*

    cmp ax, 0         *; Check if quotient is zero*

    jne itoa\_loop1     *; Repeat if not*

itoa\_loop2:

    pop dx              *; Pop the last remainder*

    add dl, '0'         *; Convert to ASCII*

    mov ah, 02h        *; Print character function*

    int 21h            *; Interrupt to print character*

    dec cx             *; Decrease the digit counter*

    cmp cx, 0          *; Check if finished printing all digits*

    jne itoa\_loop2     *; Continue if not*

    pop dx

    pop cx

    pop bx

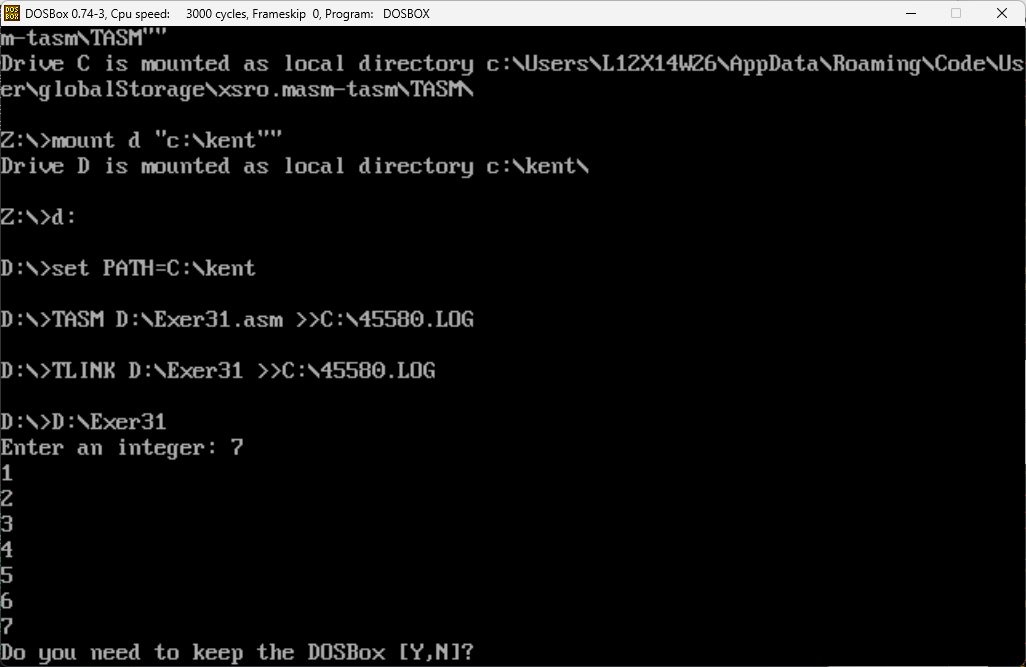
    pop ax

    ret

int 27h               *; Terminate program*

end start

* + RUN SCREENSHOT



1. EXER3.ASM
   * CODE

*; Filename: EXER32.ASM*

*; Programmer Name: KENT A. RATO*

*; Date: SEPTEMBER 29, 2024*

*; Description: Create and call a function, sub-program, or procedure for the main menu and for each math operation.*

.model small

.stack 500h

.data

    intro       db 'THE CALCULATOR',13,10

                DB 'Created by: KENT A. RATO',13,10

                db 'Date: September 29, 2024', 13,10,10

                db 'Main Menu',13,10,'$'

    a           db 'a - Addition',13,10,'$'

    s           db 's - Subtraction',13,10,'$'

    m           db 'm - Multiplication',13,10,'$'

    d           db 'd - Division',13,10,'$'

    e           db 'e - Exit',13,10,'$'

    choice      db 'Enter your choice: $'

    aPrompt     db 'Addition$'

    aPrompt1    db 13,10,'Enter first addend: $'

    aPrompt2    db 13,10,'Enter second addend: $'

    aDisplay1   db 13,10,'First addend is: $'

    aDisplay2   db 13,10,'Second addend is: $'

    aDisplay3   db 13,10,'Sum: $'

    sPrompt     db 'Subtraction$'

    sPrompt1    db 13,10,'Enter minuend: $'

    sPrompt2    db 13,10,'Enter subtrahend: $'

    sDisplay1   db 13,10,'Minuend is: $'

    sDisplay2   db 13,10,'Subtrahend is: $'

    sDisplay3   db 13,10,'Difference: $'

    mPrompt     db 'Multiplication$'

    mPrompt1    db 13,10,'Enter multiplicand: $'

    mPrompt2    db 13,10,'Enter multiplier: $'

    mDisplay1   db 13,10,'Multiplicand is: $'

    mDisplay2   db 13,10,'Multiplier is: $'

    mDisplay3   db 13,10,'Product: $'

    dPrompt     db 'Division$'

    dPrompt1    db 13,10,'Enter dividend: $'

    dPrompt2    db 13,10,'Enter divisor: $'

    dDisplay1   db 13,10,'Dividend is: $'

    dDisplay2   db 13,10,'Divisor is: $'

    dDisplay3   db 13,10,'Quotient: $'

    eDisplay    db 'Exit Program$'

    invalid     db 'INVALID CHOICE!$'

    ending      db 13,10,'Press Enter to continue.$'

    negSign     db '-$'

    divZero     db 'Error: Division by zero is not allowed.$', 13, 10

    input1   dw ?

    input2   dw ?

    sum      dw ?

    diff     dw ?

    prod     dw ?

    quo      dw ?

.code

print proc

    mov ah, 09h

    int 21h

    ret

print endp

getNum PROC

*; Read a single digit from keyboard and store in AX*

    mov ah, 01h

    int 21h

    sub al, '0'          *; Convert from ASCII to integer*

    mov ah, 0

    ret

getNum ENDP

getChar PROC

    mov ah,01h

    int 21h

    ret

getChar ENDP

converter proc

    push ax

    push bx

    push cx

    push dx

    mov cx, 0            *; Counter for digits*

    mov bx, 10           *; Base for decimal conversion*

    converter\_loop1:

        xor dx, dx           *; Clear DX before dividing*

        div bx               *; AX / BX, result in AX, remainder in DX*

        push dx              *; Push remainder onto stack*

        inc cx               *; Count the number of digits*

        cmp ax, 0            *; Check if quotient is zero*

        jne converter\_loop1       *; Repeat if not*

    converter\_loop2:

        pop dx               *; Pop the last remainder*

        add dl, '0'          *; Convert to ASCII*

        mov ah, 02h          *; Print character function*

        int 21h              *; Interrupt to print character*

        dec cx               *; Decrease the digit counter*

        cmp cx, 0            *; Check if finished printing all digits*

        jne converter\_loop2       *; Continue if not*

    pop dx

    pop cx

    pop bx

    pop ax

    ret

converter endp

newLine PROC

    mov ah, 02h

    mov dl, 13

    int 21h

    mov ah, 02h

    mov dl, 10

    int 21h

    ret

newLine ENDP

addition proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,20h *;green bg and black text*

    mov cx,8

    int 10h

    lea dx, aPrompt

    call print

    lea dx, aPrompt1

    call print

    call getNum

    mov input1, ax

    lea dx, aPrompt2

    call print

    call getNum

    mov input2, ax

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    mov dx, input1

    add dx, input2

    mov sum, dx

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    lea dx, aDisplay1

    call print

    mov ax, input1

    call converter

    lea dx, aDisplay2

    call print

    mov ax, input2

    call converter

    lea dx, aDisplay3

    call print

    mov ax, sum

    call converter

    call newLine

    ret

addition endp

subtraction proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,30h *;cyan bg and black text*

    mov cx,11

    int 10h

    lea dx, sPrompt

    call print

    lea dx, sPrompt1

    call print

    call getNum

    mov input1, ax

    lea dx, sPrompt2

    call print

    call getNum

    mov input2, ax

    lea dx, sDisplay1

    call print

    mov ax, input1

    call converter

    lea dx, sDisplay2

    call print

    mov ax, input2

    call converter

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    mov ax, input1

    sub ax, input2

    mov diff, ax

    cmp ax, 0

    lea dx, sDisplay3

    call print

    jge display\_result

    neg ax

    mov diff, ax

    lea dx, negSign

    call print

    display\_result:

        mov ax, diff

        call converter

    call newLine

    ret

subtraction endp

multiplication proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,50h *;purple bg and black text*

    mov cx,14

    int 10h

    lea dx, mPrompt

    call print

    lea dx, mPrompt1

    call print

    call getNum

    mov input1, ax

    lea dx, mPrompt2

    call print

    call getNum

    mov input2, ax

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    mov ax, input1

    mov dx, input2

    mul dx

    mov prod, ax

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    lea dx, mDisplay1

    call print

    mov ax, input1

    call converter

    lea dx, mDisplay2

    call print

    mov ax, input2

    call converter

    lea dx, mDisplay3

    call print

    mov ax, prod

    call converter

    call newLine

    ret

multiplication endp

checkAndHandleZero PROC

    cmp bx, 0

    jne continueDivision

    lea dx, divZero

    call print

    continueDivision:

        ret

checkAndHandleZero ENDP

division proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,60h *;orange bg and black text*

    mov cx,8

    int 10h

    lea dx, dPrompt

    call print

    lea dx, dPrompt1

    call print

    call getNum

    mov input1, ax

    lea dx, dPrompt2

    call print

    call getNum

    mov input2, ax

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    mov ax, input1

    mov bx, input2

    call checkAndHandleZero

    xor dx, dx

    div bx

    mov quo, ax

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

    lea dx, dDisplay1

    call print

    mov ax, input1

    call converter

    lea dx, dDisplay2

    call print

    mov ax, input2

    call converter

    lea dx, dDisplay3

    call print

    mov ax, quo

    call converter

    call newLine

    ret

division endp

exitProg proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,10h *;blue bg and black text*

    mov cx,12

    int 10h

    lea dx, eDisplay

    call print

    call newLine

    call terminate

exitProg endp

invalidChoice proc

    call newLine

    call newLine

    mov ah,09h

    mov bl,0CEh *;red bg and blinking yellow text*

    mov cx,15

    int 10h

    lea dx, invalid

    mov ah, 09h

    int 21h

    call newLine

    ret

invalidChoice endp

endingDisplay proc

    lea dx, ending

    call print

    call getChar

    ret

endingDisplay endp

terminate proc

    lea dx, ending

    call print

    call getChar

    mov ax, 4C00h

    int 21h

terminate endp

start:

    mov  ax, @data

    mov  ds, ax

    startLoop:

        mov ax, 3

        int 10h

        lea dx, intro

        call print

        mov ah,09h

        mov bl,20h *;green bg and black text*

        mov cx,12

        int 10h

        lea dx, a

        call print

        mov ah,09h

        mov bl,30h *;cyan bg and black text*

        mov cx,15

        int 10h

        lea dx, s

        call print

        mov ah,09h

        mov bl,50h *;purple bg and black text*

        mov cx,18

        int 10h

        lea dx, m

        call print

        mov ah,09h

        mov bl,60h *;orange bg and black text*

        mov cx,12

        int 10h

        lea dx, d

        call print

        mov ah,09h

        mov bl,10h *;blue bg and black text*

        mov cx,8

        int 10h

        lea dx, e

        call print

        call newLine

        lea dx, choice

        call print

        mov ah, 01h

        int 21h

        cmp al,'a'

        je doAdd

        cmp al,'s'

        je doSub

        cmp al,'m'

        je doMult

        cmp al,'d'

        je doDiv

        cmp al,'e'

        je doExit

        jne doInvalid

        doAdd:

            call addition

            call endingDisplay

            jmp startLoop

        doSub:

            call subtraction

            call endingDisplay

            jmp startLoop

        doMult:

            call multiplication

            call endingDisplay

            jmp startLoop

        doDiv:

            call division

            call endingDisplay

            jmp startLoop

        doExit:

            call exitProg

        doInvalid:

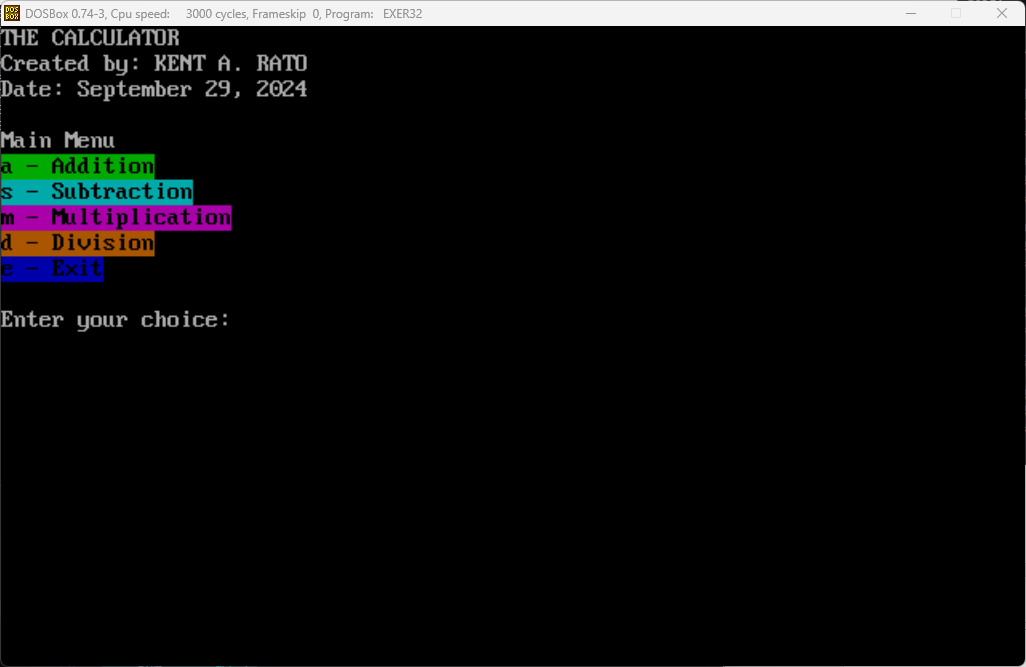
            call invalidChoice

            call endingDisplay

            jmp startLoop

end start

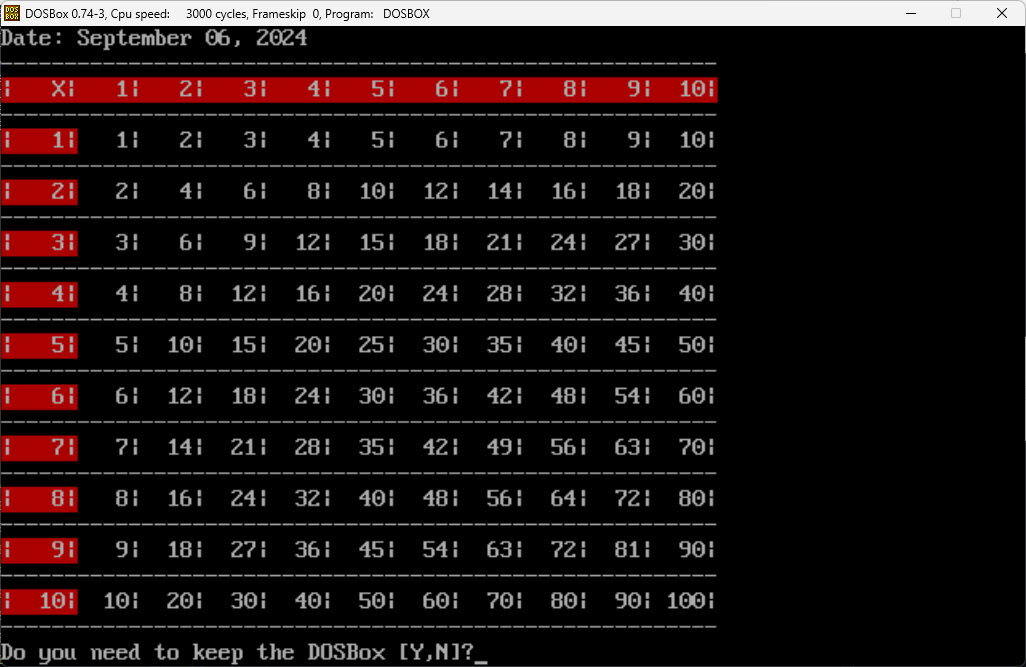
* + RUN SCREENSHOT



B. Hands-on Exams

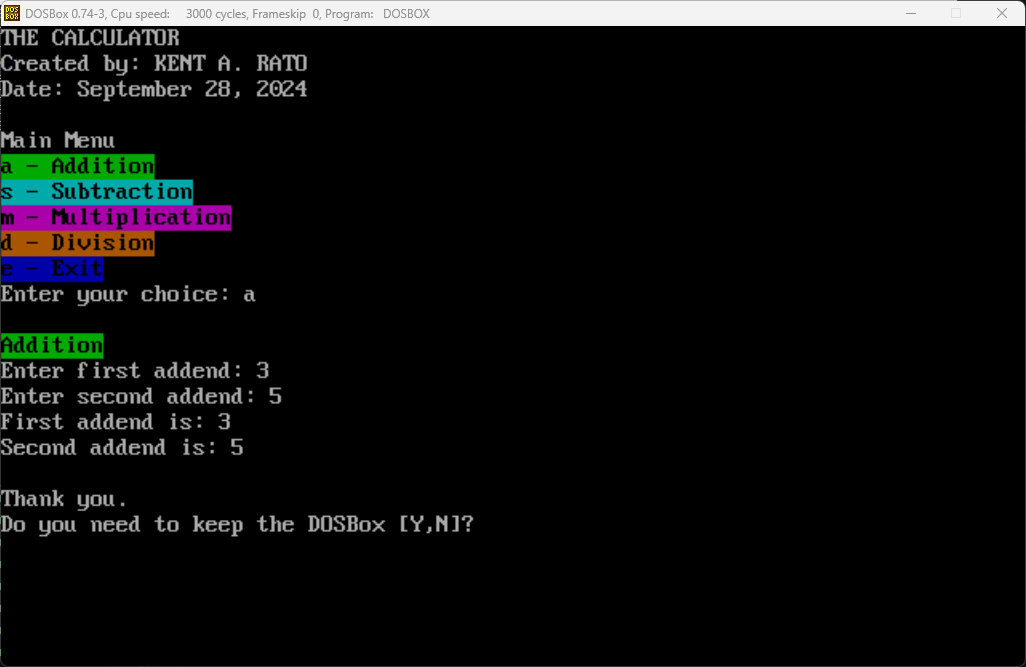
1. EXAM1.ASM

* CODE
* .model small
* .stack 100h
* .data
* titleName db 'MULTIPLICATION TABLE$'
* createdBy db 'Created by: KENT RATO$'
* date      db 'Date: September 06, 2024$'
* line      db '--------------------------------------------------------$'
* row1      db '|   X|   1|   2|   3|   4|   5|   6|   7|   8|   9|  10|$'
* row2      db '|   1|   1|   2|   3|   4|   5|   6|   7|   8|   9|  10|$'
* row3      db '|   2|   2|   4|   6|   8|  10|  12|  14|  16|  18|  20|$'
* row4      db '|   3|   3|   6|   9|  12|  15|  18|  21|  24|  27|  30|$'
* row5      db '|   4|   4|   8|  12|  16|  20|  24|  28|  32|  36|  40|$'
* row6      db '|   5|   5|  10|  15|  20|  25|  30|  35|  40|  45|  50|$'
* row7      db '|   6|   6|  12|  18|  24|  30|  36|  42|  48|  54|  60|$'
* row8      db '|   7|   7|  14|  21|  28|  35|  42|  49|  56|  63|  70|$'
* row9      db '|   8|   8|  16|  24|  32|  40|  48|  56|  64|  72|  80|$'
* row10     db '|   9|   9|  18|  27|  36|  45|  54|  63|  72|  81|  90|$'
* row11     db '|  10|  10|  20|  30|  40|  50|  60|  70|  80|  90| 100|$'
* .code
* main proc
* *;set data*
* mov ax, @data
* mov ds, ax
* *;print information*
* mov ah, 09h
* lea dx, titleName
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, createdBy
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, date
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;line1 start line*
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row1*
* mov ah, 09h
* mov bl, 47h
* mov cx, 56
* int 10h
* mov ah, 09h
* lea dx, row1
* int 21h
* *;line2*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row2*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row2
* int 21h
* *;line3*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row3*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row3
* int 21h
* *;line4*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row4*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row4
* int 21h
* *;line5*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row5*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row5
* int 21h
* *;line6*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row6*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row6
* int 21h
* *;line7*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row7*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row7
* int 21h
* *;line8*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row8*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row8
* int 21h
* *;line9*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row9*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row9
* int 21h
* *;line10*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row10*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row10
* int 21h
* *;line11*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* *;row11*
* mov ah, 09h
* mov bl, 47h
* mov cx, 6
* int 10h
* mov ah, 09h
* lea dx, row11
* int 21h
* *;line12 end line*
* mov ah, 02h
* mov cl, 0Ah
* mov dl, cl
* int 21h
* mov ah, 09h
* lea dx, line
* int 21h
* *;exit program*
* mov ah, 4Ch
* int 21h
* main endp
* end main
* RUN SCREENSHOT



1. EXAM2.ASM

* CODE
* *;Filename: EXAM2.ASM*
* *;CS243 Lab Hands-on Exam No. 2*
* *;First Semester SY 2024-2025*
* *;Student Name: KENT A. RATo*
* *;Date Finished: September 28, 2024*
* .model small
* .stack 500h
* .data
* intro       db 'THE CALCULATOR',13,10
* DB 'Created by: KENT A. RATO',13,10
* db 'Date: September 28, 2024', 13,10,10
* db 'Main Menu',13,10,'$'
* a           db 'a - Addition',13,10,'$'
* s           db 's - Subtraction',13,10,'$'
* m           db 'm - Multiplication',13,10,'$'
* d           db 'd - Division',13,10,'$'
* e           db 'e - Exit',13,10,'$'
* choice      db 'Enter your choice: $'
* aPrompt     db 'Addition$'
* aPrompt1    db 13,10,'Enter first addend: $'
* aPrompt2    db 13,10,'Enter second addend: $'
* aDisplay1   db 13,10,'First addend is: $'
* aDisplay2   db 13,10,'Second addend is: $'
* sPrompt     db 'Subtraction$'
* sPrompt1    db 13,10,'Enter minuend: $'
* sPrompt2    db 13,10,'Enter subtrahend: $'
* sDisplay1   db 13,10,'Minuend is: $'
* sDisplay2   db 13,10,'Subtrahend is: $'
* mPrompt     db 'Multiplication$'
* mPrompt1    db 13,10,'Enter multiplicand: $'
* mPrompt2    db 13,10,'Enter multiplier: $'
* mDisplay1   db 13,10,'Multiplicand is: $'
* mDisplay2   db 13,10,'Multiplier is: $'
* dPrompt     db 'Division$'
* dPrompt1    db 13,10,'Enter dividend: $'
* dPrompt2    db 13,10,'Enter divisor: $'
* dDisplay1   db 13,10,'Dividend is: $'
* dDisplay2   db 13,10,'Divisor is: $'
* eDisplay    db 'Exit Program$'
* invalid     db 'INVALID CHOICE!$'
* thanks      db 13,10,'Thank you.$'
* input1      db 10, 10 dup(?)
* input2      db 10, 10 dup(?)
* .code
* print proc
* mov ah, 09h
* int 21h
* ret
* print endp
* scan PROC
* mov  ah, 0AH
* int  21h
* ret
* scan ENDP
* newLine PROC
* mov ah, 02h
* mov dl, 10
* int 21h
* ret
* newLine ENDP
* displaychar PROC
* mov  dl, [si]
* mov  ah, 02h
* int  21h
* inc  si
* ret
* displaychar ENDP
* start:
* mov  ax, @data
* mov  ds, ax
* mov ax, 3
* int 10h
* lea dx, intro
* call print
* mov ah,09h
* mov bl,20h *;green bg and black text*
* mov cx,12
* int 10h
* lea dx, a
* call print
* mov ah,09h
* mov bl,30h *;cyan bg and black text*
* mov cx,15
* int 10h
* lea dx, s
* call print
* mov ah,09h
* mov bl,50h *;purple bg and black text*
* mov cx,18
* int 10h
* lea dx, m
* call print
* mov ah,09h
* mov bl,60h *;orange bg and black text*
* mov cx,12
* int 10h
* lea dx, d
* call print
* mov ah,09h
* mov bl,10h *;blue bg and black text*
* mov cx,8
* int 10h
* lea dx, e
* call print
* lea dx, choice
* call print
* mov ah, 01h
* int 21h
* cmp al, 'a'
* je case\_a
* jmp check\_s
* case\_a:
* call newLine
* call newLine
* mov ah,09h
* mov bl,20h *;green bg and black text*
* mov cx,8
* int 10h
* lea dx, aPrompt
* call print
* lea dx, aPrompt1
* call print
* lea dx, input1
* call scan
* lea dx, aPrompt2
* call print
* lea dx, input2
* call scan
* lea dx, aDisplay1
* call print
* lea  si, input1 + 2
* mov  cl, [input1 + 1]
* loop1:
* call displaychar
* loop loop1
* lea dx, aDisplay2
* call print
* lea  si, input2 + 2
* mov  cl, [input2 + 1]
* loop2:
* call displaychar
* loop loop2
* call newLine
* jmp thank\_you
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* check\_s:
* cmp al, 's'
* je case\_s
* jmp check\_m
* case\_s:
* call newLine
* call newLine
* mov ah,09h
* mov bl,30h *;cyan bg and black text*
* mov cx,11
* int 10h
* lea dx, sPrompt
* call print
* lea dx, sPrompt1
* call print
* lea dx, input1
* call scan
* lea dx, sPrompt2
* call print
* lea dx, input2
* call scan
* lea dx, sDisplay1
* call print
* lea  si, input1 + 2
* mov  cl, [input1 + 1]
* loop3:
* call displaychar
* loop loop3
* lea dx, sDisplay2
* call print
* lea  si, input2 + 2
* mov  cl, [input2 + 1]
* loop4:
* call displaychar
* loop loop4
* call newLine
* jmp thank\_you
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* check\_m:
* cmp al, 'm'
* je case\_m
* jmp check\_d
* case\_m:
* call newLine
* call newLine
* mov ah,09h
* mov bl,50h *;purple bg and black text*
* mov cx,14
* int 10h
* lea dx, mPrompt
* call print
* lea dx, mPrompt1
* call print
* lea dx, input1
* call scan
* lea dx, mPrompt2
* call print
* lea dx, input2
* call scan
* lea dx, mDisplay1
* call print
* lea  si, input1 + 2
* mov  cl, [input1 + 1]
* loop5:
* call displaychar
* loop loop5
* lea dx, mDisplay2
* call print
* lea  si, input2 + 2
* mov  cl, [input2 + 1]
* loop6:
* call displaychar
* loop loop6
* call newLine
* jmp thank\_you
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* check\_d:
* cmp al, 'd'
* je case\_d
* jmp check\_e
* case\_d:
* call newLine
* call newLine
* mov ah,09h
* mov bl,60h *;orange bg and black text*
* mov cx,8
* int 10h
* lea dx, dPrompt
* call print
* lea dx, dPrompt1
* call print
* lea dx, input1
* call scan
* lea dx, dPrompt2
* call print
* lea dx, input2
* call scan
* lea dx, dDisplay1
* call print
* lea  si, input1 + 2
* mov  cl, [input1 + 1]
* loop7:
* call displaychar
* loop loop7
* lea dx, dDisplay2
* call print
* lea  si, input2 + 2
* mov  cl, [input2 + 1]
* loop8:
* call displaychar
* loop loop8
* call newLine
* jmp thank\_you
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* check\_e:
* cmp al, 'e'
* je case\_e
* jmp invalidChoice
* case\_e:
* call newLine
* call newLine
* mov ah,09h
* mov bl,10h *;blue bg and black text*
* mov cx,12
* int 10h
* lea dx, eDisplay
* call print
* call newLine
* jmp thank\_you
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* invalidChoice:
* call newLine
* call newLine
* mov ah,09h
* mov bl,0CEh *;red bg and blinking yellow text*
* mov cx,15
* int 10h
* lea dx, invalid
* mov ah, 09h
* int 21h
* call newLine
* *;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*
* thank\_you:
* lea dx, thanks
* call print
* mov ax, 4C00h
* int 21h
* end start
* RUN SCREENSHOT



1. EXAM3.ASM

* CODE

*;Filename: EXAM3.ASM*

*;CS243 Lab Hands-on Exam No. 3*

*;First Semester SY 2024-2025*

*;Student Name: KENT A. RATO*

*;Date Finished: October 04, 2024*

.model small

.stack 100h

.data

    header       db "VOTER'S REGISTRATION FORM"

                 db 0ah,"Created by: name"

                 db 0ah,"Date: 10 - 03 - 2024"

                 db 0ah,0ah,"Please enter the following data: ",0ah,'$'

    text1      db 0ah,'First Name: $'

    text2      db 0ah,'Middle Name: $'

    text3      db 0ah,'Family Name: $'

    text4      db 0ah,'Gender: $'

    text5      db 0ah,'Birthday'

                 db 0ah,09h,'Month: $'

    text6      db 0ah,09h,'Day: $'

    text7      db 0ah,09h,'Year: $'

    text8      db 0ah,'Address'

                 db 0ah,09h,'House Number: $'

    text9      db 0ah,09h,'Street: $'

    text10     db 0ah,09h,'Barangay: $'

    text11     db 0ah,09h,'City: $'

    text12     db 0ah,09h,'Province: $'

    text13     db 0ah,'Educational Background'

                 db 0ah,09h,'Elementary: '

                 db 0ah,09h,09h,'Name of School: $'

    text14     db 0ah,09h,09h,'Year Graduated: $'

    text15     db 0ah,09h,'Junior High School: '

                 db 0ah,09h,09h,'Name of School: $'

    text16     db 0ah,09h,09h,'Year Graduated: $'

    text17     db 0ah,09h,'Senior High School: '

                 db 0ah,09h,09h,'Name of School: $'

    text18     db 0ah,09h,09h,'Year Graduated: $'

    text19     db 0ah,09h,'College: '

                 db 0ah,09h,09h,'Name of School: $'

    text20     db 0ah,09h,09h,'Course: $'

    text21     db 0ah,09h,09h,'Year Graduated: $'

    input1       db 10, ?, 10 dup(' ')

    input2       db 10, ?, 10 dup(' ')

    input3       db 10, ?, 10 dup(' ')

    input4       db 10, ?, 10 dup(' ')

    input5       db 10, ?, 10 dup(' ')

    input6       db 10, ?, 10 dup(' ')

    input7       db 10, ?, 10 dup(' ')

    input8       db 10, ?, 10 dup(' ')

    input9       db 20, ?, 20 dup(' ')

    input10      db 10, ?, 10 dup(' ')

    input11      db 10, ?, 10 dup(' ')

    input12      db 10, ?, 10 dup(' ')

    input13      db 50, ?, 50 dup(' ')

    input14      db 10, ?, 10 dup(' ')

    input15      db 50, ?, 50 dup(' ')

    input16      db 10, ?, 10 dup(' ')

    input17      db 50, ?, 50 dup(' ')

    input18      db 10, ?, 10 dup(' ')

    input19      db 50, ?, 50 dup(' ')

    input20      db 10, ?, 10 dup(' ')

    input21      db 15, ?, 15 dup(' ')

    outputHeader db 0ah,0AH, 'SUMMARY OF INFORMATION'

                 DB 0AH, 'Please verify if all entries are correct. $'

    output1      db 0ah,'First Name: $'

    output2      db 0ah,'Middle Name: $'

    output3      db 0ah,'Family Name: $'

    output4      db 0ah,'Gender: $'

    output5      db 0ah,'Birthday'

                 db 0ah,09h,'Month: $'

    output6      db 0ah,09h,'Day: $'

    output7      db 0ah,09h,'Year: $'

    output8      db 0ah,'Address'

                 db 0ah,09h,'House Number: $'

    output9      db 0ah,09h,'Street: $'

    output10     db 0ah,09h,'Barangay: $'

    output11     db 0ah,09h,'City: $'

    output12     db 0ah,09h,'Province: $'

    output13     db 0ah,'Educational Background'

                 db 0ah,09h,'Elementary: '

                 db 0ah,09h,09h,'Name of School: $'

    output14     db 0ah,09h,09h,'Year Graduated: $'

    output15     db 0ah,09h,'Junior High School: '

                 db 0ah,09h,09h,'Name of School: $'

    output16     db 0ah,09h,09h,'Year Graduated: $'

    output17     db 0ah,09h,'Senior High School: '

                 db 0ah,09h,09h,'Name of School: $'

    output18     db 0ah,09h,09h,'Year Graduated: $'

    output19     db 0ah,09h,'College: '

                 db 0ah,09h,09h,'Name of School: $'

    output20     db 0ah,09h,09h,'Course: $'

    output21     db 0ah,09h,09h,'Year Graduated: $'

    thanks       db 'Vote Wisely!$'

    footer       db 'Thank you for registering.$'

.code

newLine PROC

             mov  ah, 02h

             mov  dl, 13

             int  21h

             mov  ah, 02h

             mov  dl, 10

             int  21h

             ret

newLine ENDP

thankYou PROC

             call newLine

             call newLine

             mov  ah,09h

             mov  bl,0CEh

             mov  cx,12

             int  10h

             lea  dx, thanks

             mov  ah, 09h

             int  21h

             call newLine

             ret

thankYou ENDP

    start:

             mov  ax, @data

             mov  ds, ax

             mov  ax, 3

             int  10h

             mov  ax, 1112h

             xor  bl, bl

             int  10h

             mov  ah, 06h

             xor  al, al

             xor  cx, cx

             mov  ah, 06h

             mov  ch, 0

             mov  cl, 0

             mov  dh, 0

             mov  dl, 24

             mov  bh, 01Fh

             int  10h

             mov  ah, 06h

             mov  ch, 1

             mov  cl, 0

             mov  dh, 1

             mov  dl, 34

             mov  bh, 02Fh

             int  10h

             mov  ah, 06h

             mov  ch, 2

             mov  cl, 0

             mov  dh, 2

             mov  dl, 20

             mov  bh, 03Fh

             int  10h

             lea  dx, header

             mov  ah, 9

             int  21h

             lea  dx, text1

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input1

             int  21h

             lea  dx, text2

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input2

             int  21h

             lea  dx, text3

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input3

             int  21h

             lea  dx, text4

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input4

             int  21h

             lea  dx, text5

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input5

             int  21h

             lea  dx, text6

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input6

             int  21h

             lea  dx, text7

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input7

             int  21h

             lea  dx, text8

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input8

             int  21h

             lea  dx, text9

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input9

             int  21h

             lea  dx, text10

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input10

             int  21h

             lea  dx, text11

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input11

             int  21h

             lea  dx, text12

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input12

             int  21h

             lea  dx, text13

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input13

             int  21h

             lea  dx, text14

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input14

             int  21h

             lea  dx, text15

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input15

             int  21h

             lea  dx, text16

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input16

             int  21h

             lea  dx, text17

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input17

             int  21h

             lea  dx, text18

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input18

             int  21h

             lea  dx, text19

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input19

             int  21h

             lea  dx, text20

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input20

             int  21h

             lea  dx, text21

             mov  ah, 9

             int  21h

             mov  ah, 0Ah

             lea  dx, input21

             int  21h

             lea  dx, outputHeader

             mov  ah, 9

             int  21h

             lea  dx, output1

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input1[1]

             mov  input1[bx+2], '$'

             lea  dx, input1 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output2

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input2[1]

             mov  input2[bx+2], '$'

             lea  dx, input2 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output3

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input3[1]

             mov  input3[bx+2], '$'

             lea  dx, input3 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output4

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input4[1]

             mov  input4[bx+2], '$'

             lea  dx, input4 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output5

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input5[1]

             mov  input5[bx+2], '$'

             lea  dx, input5 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output6

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input6[1]

             mov  input6[bx+2], '$'

             lea  dx, input6 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output7

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input7[1]

             mov  input7[bx+2], '$'

             lea  dx, input7 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output8

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input8[1]

             mov  input8[bx+2], '$'

             lea  dx, input8 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output9

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input9[1]

             mov  input9[bx+2], '$'

             lea  dx, input9 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output10

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input10[1]

             mov  input10[bx+2], '$'

             lea  dx, input10 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output11

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input11[1]

             mov  input11[bx+2], '$'

             lea  dx, input11 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output12

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input12[1]

             mov  input12[bx+2], '$'

             lea  dx, input12 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output13

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input13[1]

             mov  input13[bx+2], '$'

             lea  dx, input13 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output14

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input14[1]

             mov  input14[bx+2], '$'

             lea  dx, input14 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output15

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input15[1]

             mov  input15[bx+2], '$'

             lea  dx, input15 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output16

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input16[1]

             mov  input16[bx+2], '$'

             lea  dx, input16 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output17

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input17[1]

             mov  input17[bx+2], '$'

             lea  dx, input17 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output18

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input18[1]

             mov  input18[bx+2], '$'

             lea  dx, input18 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output19

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input19[1]

             mov  input19[bx+2], '$'

             lea  dx, input19 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output20

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input20[1]

             mov  input20[bx+2], '$'

             lea  dx, input20 + 2

             mov  ah, 09h

             int  21h

             lea  dx, output21

             mov  ah, 9

             int  21h

             xor  bx, bx

             mov  bl, input21[1]

             mov  input21[bx+2], '$'

             lea  dx, input21 + 2

             mov  ah, 09h

             int  21h

             mov  ah, 06h

             xor  al, al

             xor  cx, cx

             call thankYou

             lea  dx, footer

             mov  ah, 9

             int  21h

             mov  ah, 4Ch

             int  21h

end start

* RUN SCREENSHOT

