Practice God Level Questions 2

Exercise 1: Draw Text Colors

```
include Irvine32.inc

.data
    colors BYTE 2, 4, 6, 14   ; Colors: Green, Red, Yellow, White
    message BYTE "Hello, Colors!",0

.code
main PROC
    mov ecx, 4  ; Number of colors
    mov esi, 0  ; Index for colors array

loop_colors:
    mov eax, colors[esi]
    call SetTextColor
    mov edx, OFFSET message
    call WriteString
    call Crlf

    inc esi
    loop loop_colors

    call WaitMsg  ; Wait for a key press
    call Clrscr   ; Clear the screen
    call ExitProcess
main ENDP
END main
```

Exercise 2: Linking Array Items

```
.data
    start DWORD 1
    chars BYTE 'H', 'A', 'C', 'E', 'B', 'D', 'F', 'G'
    links DWORD 0, 4, 5, 6, 2, 3, 7, 0
    outputArray BYTE 8 DUP(?) ; To store the characters in order
.code
    main PROC
        mov edi, OFFSET outputArray  ; Destination for output characters
        mov esi, start  ; Start index
        mov ecx, 8     ; Number of characters to locate
        traverse_links:
            mov al, chars[esi]  ; Load character
            mov [edi], al       ; Store it in outputArray
            inc edi             ; Move to the next position in outputArray
            ; Get the next link index
            mov eax, esi
            mov ebx, 4          ; Size of DWORD (4 bytes)
            mul ebx             ; Multiply esi by 4
            mov esi, links[eax] ; Get the next link index

            loop traverse_links  ; Repeat for all characters

        ; Display the characters in outputArray
        mov edx, OFFSET outputArray
        call WriteString
        call Crlf

        call WaitMsg  ; Wait for a key press
        call ExitProcess
    main ENDP
END main
```

Exercise 3: Simple Addition (1)

```
include Irvine32.inc
.data
    prompt1 BYTE "Enter the first integer: ", 0
    prompt2 BYTE "Enter the second integer: ", 0
    resultMsg BYTE "The sum is: ", 0
    buffer1 DWORD ?
    buffer2 DWORD ?
    result DWORD ?
.code
    main PROC
        call Clrscr
        mov edx, OFFSET prompt1
        call WriteString
        call ReadInt
        mov buffer1, eax

        mov edx, OFFSET prompt2
        call WriteString
        call ReadInt
        mov buffer2, eax

        ; Add the integers
        mov eax, buffer1
        add eax, buffer2
        mov result, eax

        mov edx, OFFSET resultMsg
        call WriteString
        mov eax, result
        call WriteInt

        call WaitMsg  ; Wait for a key press
        call ExitProcess
    main ENDP
END main
```

Exercise 4: Simple Addition (2)

```
include Irvine32.inc
.data
    prompt1 BYTE "Enter the first integer: ", 0
    prompt2 BYTE "Enter the second integer: ", 0
    resultMsg BYTE "The sum is: ", 0
    buffer1 DWORD ?
    buffer2 DWORD ?
    result DWORD ?

.code
    main PROC
        mov ecx, 3  ; Repeat the process three times
        loop_repeat:
            call Clrscr

            mov edx, OFFSET prompt1
            call WriteString
            call ReadInt
            mov buffer1, eax

            mov edx, OFFSET prompt2
            call WriteString
            call ReadInt
            mov buffer2, eax

            ; Add the integers
            mov eax, buffer1
            add eax, buffer2
            mov result, eax
```

```
729
730             mov edx, OFFSET resultMsg
731             call WriteString
732             mov eax, result
733             call WriteInt
734
735             call WaitMsg  ; Wait for a key press
736
737             dec ecx
738             jnz loop_repeat  ; Repeat the process three times
739
740         call ExitProcess
741     main ENDP
742 END main
```

Exercise 5: BetterRandomRange Procedure

```
include Irvine32.inc

.data

.code
BetterRandomRange PROC
    ; Input: EBX = Lower bound (M)
    ;        EAX = Upper bound (N)
    ; Output: EAX = Random number between M and N-1

    sub eax, ebx  ; Calculate the range (N - M)
    add eax, 1    ; Include the upper bound itself
    call RandomRange
    add eax, ebx  ; Offset the result by M (lower bound)
    ret
BetterRandomRange ENDP

main PROC
    mov ecx, 50  ; Repeat 50 times
    loop_repeat:
        call Clrscr

        mov ebx, -300 ; Lower bound
        mov eax, 100  ; Upper bound
        call BetterRandomRange

        ; Display the randomly generated value
        mov edx, eax
        call WriteInt
        call Crlf
```

```

        call WaitMsg  ; Wait for a key press
        dec ecx
        jnz loop_repeat  ; Repeat the process 50 times

    call ExitProcess
main ENDP
END main
```

These solutions demonstrate the requested functionality for each exercise. Feel free to modify them, as you do your practice to be a

better programmer.