CHEAT SHEET

**1. Clrscr**

**Description:** Clears the console window and moves the cursor to the upper-left corner.

**Example:**

include irvine32.inc

.code

main proc

call Clrscr

exit

main endp

end main

**Expected Output:** The console window will be cleared.

**2. Crlf**

**Description:** Writes a carriage return and line feed (newline).

**Example:**

include irvine32.inc

.data

msg byte "Hello, World!", 0

.code

main proc

mov edx, offset msg

call writestring

call crlf

call writestring

exit

main endp

end main

**Expected Output:**

Hello, World!

Hello, World!

**3. WriteBin**

**Description:** Writes a 32-bit unsigned integer to the console in binary format.

**Example:**

include irvine32.inc

.code

main proc

mov eax, 13

call WriteBin

exit

main endp

end main

**Expected Output:**

1101

**4. WriteChar**

**Description:** Writes a single character to the console.

**Example:**

include irvine32.inc

.code

main proc

mov al, 'A'

call WriteChar

exit

main endp

end main

**Expected Output:**

A

**5. WriteDec**

**Description:** Writes an unsigned 32-bit integer in decimal format.

**Example:**

include irvine32.inc

.code

main proc

mov eax, 12345

call WriteDec

exit

main endp

end main

**Expected Output:**

12345

**6. WriteHex**

**Description:** Writes a 32-bit integer in hexadecimal format.

**Example:**

include irvine32.inc

.code

main proc

mov eax, 255

call WriteHex

exit

main endp

end main

**Expected Output:**

FF

**7. WriteInt**

**Description:** Writes a signed 32-bit integer in decimal format.

**Example:**

include irvine32.inc

.code

main proc

mov eax, -10

call WriteInt

exit

main endp

end main

**Expected Output:**

-10

**8. WriteString**

**Description:** Writes a null-terminated string to the console.

**Example:**

include irvine32.inc

.data

msg byte "Hello, Irvine!", 0

.code

main proc

mov edx, offset msg

call writestring

exit

main endp

end main

**Expected Output:**

Hello, Irvine!

**9. ReadChar**

**Description:** Waits for a single character input and returns it in AL.

**Example:**

include irvine32.inc

.data

msg byte "Press a key: ", 0

.code

main proc

mov edx, offset msg

call writestring

call ReadChar

call WriteChar

exit

main endp

end main

**Expected Output:**

Press a key: [your input character]

**10. ReadDec**

**Description:** Reads an unsigned 32-bit integer from the keyboard.

**Example:**

include irvine32.inc

.data

msg byte "Enter a number: ", 0

.code

main proc

mov edx, offset msg

call writestring

call ReadDec

call WriteDec

exit

main endp

end main

**Expected Output:**

Enter a number: [your input number]

**11. ReadHex**

**Description:** Reads a 32-bit hexadecimal number from the keyboard.

**Example:**

include irvine32.inc

.data

msg byte "Enter a hex number: ", 0

.code

main proc

mov edx, offset msg

call writestring

call ReadHex

call WriteHex

exit

main endp

end main

**Expected Output:**

Enter a hex number: [your input in hex]

**12. ReadInt**

**Description:** Reads a signed 32-bit integer from the keyboard.

**Example:**

include irvine32.inc

.data

msg byte "Enter an integer: ", 0

.code

main proc

mov edx, offset msg

call writestring

call ReadInt

call WriteInt

exit

main endp

end main

**Expected Output:**

Enter an integer: [your input integer]

**13. ReadString**

**Description:** Reads a string from the keyboard, terminated by the Enter key.

**Example:**

include irvine32.inc

.data

buffer byte 20 dup(?)

msg byte "Enter a string: ", 0

.code

main proc

mov edx, offset msg

call writestring

mov edx, offset buffer

mov ecx, 20

call ReadString

call crlf

mov edx, offset buffer

call writestring

exit

main endp

end main

**Expected Output:**

Enter a string: [your input string]

[your input string]

**14. Delay**

**Description:** Pauses the program for the specified interval (milliseconds).

**Example:**

include irvine32.inc

.code

main proc

mov eax, 2000 ; Delay for 2000 ms (2 seconds)

call Delay

exit

main endp

end main

**Expected Output:** The program waits for 2 seconds before exiting.

**15. Randomize**

**Description:** Seeds the random number generator.

**Example:**

include irvine32.inc

.code

main proc

call Randomize

exit

main endp

end main

**Expected Output:** Seeds the random number generator for future random number generation.

**16. DumpRegs**

**Description:** Displays all CPU registers (EAX, EBX, etc.) on the console.

**Example:**

include irvine32.inc

.code

main proc

call DumpRegs

exit

main endp

end main

**Expected Output:** The contents of the CPU registers are printed on the console.

**17. DumpMem**

**Description:** Writes the block of memory in hexadecimal format.

**Example:**

include irvine32.inc

.data

buffer byte "hello", 0

.code

main proc

mov esi, offset buffer

mov ecx, 5 ; Length of the string

mov ebx, 1 ; Type (1 = byte)

call DumpMem

exit

main endp

end main

**Expected Output:**

Contents of memory in hexadecimal format

**18. getDateTime**

**Description:** Gets the current date and time from the system and stores them in registers (EAX, EBX, etc.).

**Example:**

include irvine32.inc

.code

main proc

call getDateTime

; After this call:

; EAX = year

; EBX = month

; ECX = day

; EDX = hour

; ESI = minute

; EDI = second

call DumpRegs ; Display the register values to show the date and time

exit

main endp

end main

**Expected Output:** The current date and time stored in EAX, EBX, ECX, etc., will be displayed in the register dump.

**19. GetMaxXY**

**Description:** Gets the number of columns (DX) and rows (AX) in the console window buffer.

**Example:**

include irvine32.inc

.code

main proc

call GetMaxXY

; DX = columns (width)

; AX = rows (height)

call DumpRegs ; Display the register values to show the screen dimensions

exit

main endp

end main

**Expected Output:** The current console buffer size (columns and rows) will be displayed in the register dump.

**20. GetTextColor**

**Description:** Returns the active foreground and background text colors.

**Example:**

include irvine32.inc

.code

main proc

call GetTextColor

; AL = foreground color

; AH = background color

call DumpRegs ; Display register values to show the text color settings

exit

main endp

end main

**Expected Output:** The current text foreground and background color codes will be displayed in the register dump.

**21. Gotoxy**

**Description:** Moves the cursor to a specific row (DH) and column (DL) in the console window.

**Example:**

include irvine32.inc

.code

main proc

mov dh, 5 ; Move to row 5

mov dl, 10 ; Move to column 10

call Gotoxy

mov edx, offset msg

call writestring ; Print the message at the new position

exit

main endp

end main

**Expected Output:** The string will be printed starting at row 5, column 10.

**22. MsgBox**

**Description:** Displays a message box with a custom title and message.

**Example:**

include irvine32.inc

.data

msg byte "Hello, World!", 0

titles byte "Greetings", 0

.code

main proc

mov edx, offset msg

mov ebx, offset titles

call MsgBox

exit

main endp

end main

**Expected Output:** A pop-up message box with the title "Greetings" and the message "Hello, World!".

**23. MsgBoxAsk**

**Description:** Displays a Yes/No question in a message box and returns the result in EAX (6 for Yes, 7 for No).

**Example:**

include irvine32.inc

.data

question byte "Do you want to continue?", 0

titles byte "Confirmation", 0

.code

main proc

mov edx, offset question

mov ebx, offset titles

call MsgBoxAsk

cmp eax, 6 ; Check if Yes was clicked

je yes\_clicked

cmp eax, 7 ; Check if No was clicked

je no\_clicked

yes\_clicked:

mov edx, offset msg\_yes

call writestring

jmp end\_program

no\_clicked:

mov edx, offset msg\_no

call writestring

end\_program:

exit

main endp

end main

**Expected Output:** A Yes/No message box with the question "Do you want to continue?" will appear. Depending on the choice, "Yes" or "No" will be printed in the console.

**24. SetTextColor**

**Description:** Sets the foreground and background colors for text output. The color is determined by the formula EAX = Foreground + (Background \* 16).

**Example:**

include irvine32.inc

.data

msg byte "This text is colored!", 0

.code

main proc

mov eax, 4 + (2 \* 16) ; Red foreground, Green background

call SetTextColor

mov edx, offset msg

call writestring

call crlf

exit

main endp

end main

**Expected Output:** The text will be displayed with a red foreground and green background.

**25. WaitMsg**

**Description:** Displays a message and waits for the Enter key to be pressed.

**Example:**

include irvine32.inc

.data

msg byte "Press Enter to continue...", 0

.code

main proc

mov edx, offset msg

call writestring

call WaitMsg ; Wait for the user to press Enter

exit

main endp

end main

**Expected Output:** The message "Press Enter to continue..." will be displayed, and the program will wait for the user to press Enter.