


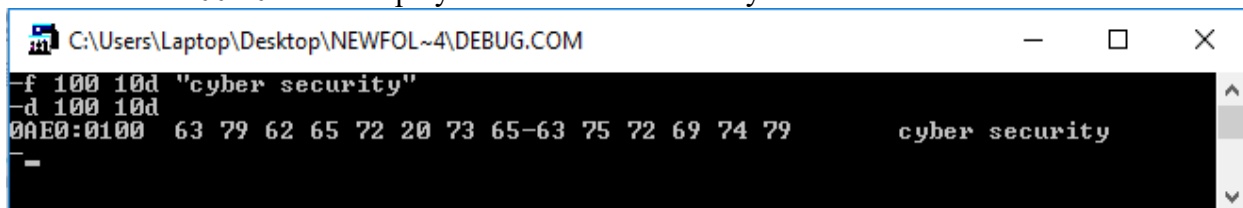
Fill : F range list

This command can be used to *clear* large areas of Memory as well as *filling* smaller areas with a continuously repeating phrase or single byte.

Examples:


Fill the locations of memory with a string “cyber security” starting at location 100h.

- F 100 10d “cyber security” 
- D 100 10d To display the contents of memory



```
C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-f 100 10d "cyber security"
-d 100 10d
00E0:0100  63 79 62 65 72 20 73 65-63 75 72 69 74 79      cyber security
-
```

Fill (1) Kbyte of the memory locations starting at address 300h with the value AB.

- F 300 6ff ab 

```

C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-f 300 6ff ab
-d 300 6ff
0AE0:0300 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0310 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0320 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0330 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0340 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0350 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0360 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0370 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0380 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0390 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03A0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03B0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03C0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03D0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03E0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:03F0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0400 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0410 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0420 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0430 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0440 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0450 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0460 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0470 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0480 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0490 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04A0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04B0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04C0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04D0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04E0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:04F0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0500 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0510 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....

.....
.....
.....
0AE0:0660 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0670 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0680 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:0690 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06A0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06B0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06C0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06D0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06E0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....
0AE0:06F0 AB AB AB AB AB AB AB AB AB-AB AB AB AB AB AB AB .....

```

Exercise:

Write a debugger command to fill a block of memory 128 bytes with the a sequence of values 10,20,30,10,20,30,10,20,30..... , Note the start address of block is 100h

Compare: C range address

The command is used to compares two blocks of memory. If there are no differences, then DEBUG simply displays another prompt (-).

Here's an example of what happens when there *are* differences:

- C 100 103 104



```
C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-f 100 17f 10 20 30
-c 100 103 104
0AE0:0100 10 20 0AE0:0104
0AE0:0101 20 30 0AE0:0105
0AE0:0102 30 10 0AE0:0106
0AE0:0103 10 20 0AE0:0107
-c 100 103 106
-
```

Search: S range list

Searches within a range of addresses for a pattern of one or more byte values given in a list. The list can be comprised of numbers or character strings enclosed by matching single or double quote marks.

```
C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-f 300 303 "BIOS"
-s 100 400 "BIOS"
0AE0:0300
-
```

```
C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-e fe00:100 "BIOS"
-e fe00:200 "BIOS"
-s fe00:0000 ffff "BIOS"
FE00:0100
FE00:0200
-d fe00:100 250
FE00:0100 42 49 4F 53 49 6E 73 69-67 6E 69 61 20 49 6E 63 BIOSInsignia Inc
FE00:0110 15 3E 5F 20 00 00 00 00-00 00 00 00 00 00 00 00 .>_ .....
FE00:0120 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0130 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0140 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0150 FF FF FF FF 48 46 58 52-45 44 49 52 00 2C 00 00 ....HF&REDIR...
FE00:0160 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0170 CD 72 C4 C4 FE 0D 0A 00-00 00 00 00 00 00 00 00 ..r.....
FE00:0180 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0190 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01A0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01B0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01C0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01D0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01E0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:01F0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0200 42 49 4F 53 00 00 00 00-00 00 00 00 00 00 00 00 BIOS.....
FE00:0210 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0220 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0230 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0240 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
FE00:0250 00
-
```

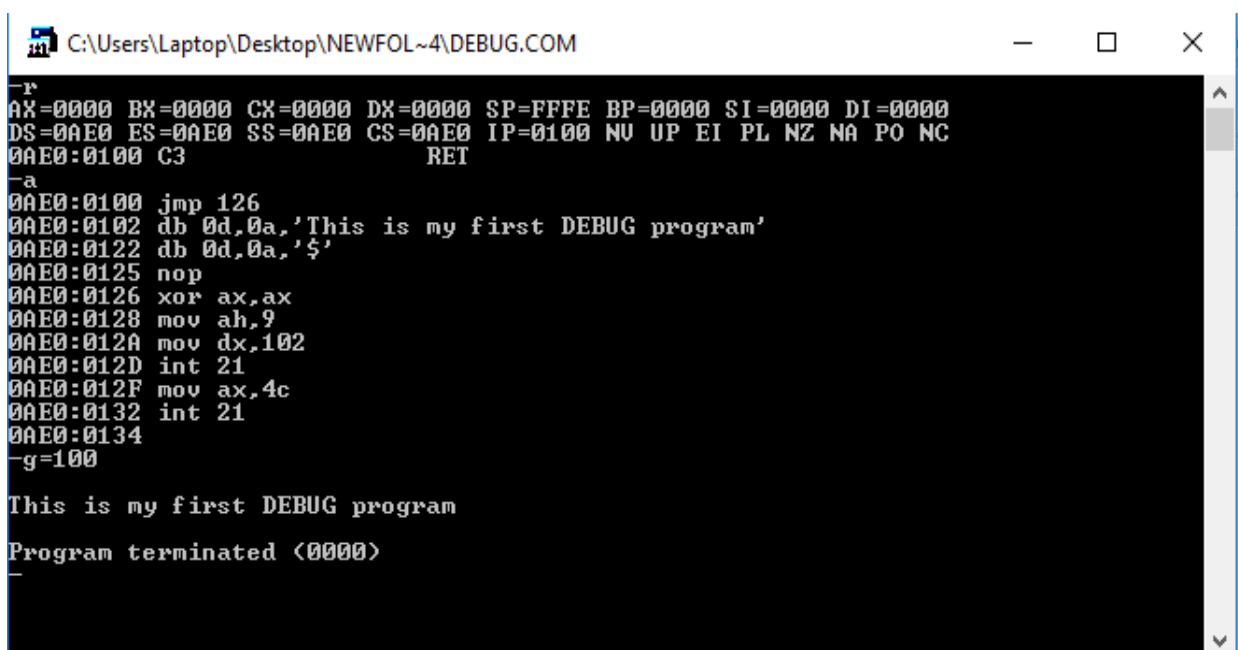
Assemble: A [address]

Creates machine executable code in memory beginning at CS:0100 (or the specified address) from the 8086/8088 (and 8087) Assembly Language instructions which are entered. Although no Macro instructions nor labels are recognized, you can use the *pseudo-instructions* 'DB' and 'DW' (so you can use the DB Opcode to enter ASCII data like this: DB 'This is a string',0D,0A).

The 'A' command remembers the last location where any data was assembled, so successive 'A' commands (when no address is specified) will always begin at the next address in the chain of assembled instructions. This aspect of the command is similar to the Dump command which remembers the location of its last dump (if no new address is specified).

The assembly process will stop *after* you ENTER an empty line.

Example (you enter the characters in bold type):



```
C:\Users\Laptop\Desktop\NEWFOL~4\DEBUG.COM
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=0AE0 ES=0AE0 SS=0AE0 CS=0AE0 IP=0100 NV UP EI PL NZ NA PO NC
0AE0:0100 C3 RET
-a
0AE0:0100 jmp 126
0AE0:0102 db 0d,0a,'This is my first DEBUG program'
0AE0:0122 db 0d,0a,'$'
0AE0:0125 nop
0AE0:0126 xor ax,ax
0AE0:0128 mov ah,9
0AE0:012A mov dx,102
0AE0:012D int 21
0AE0:012F mov ax,4c
0AE0:0132 int 21
0AE0:0134
-g=100

This is my first DEBUG program
Program terminated <0000>
```

Go: G [=address] [addresses]

Go is used to run a program and set *breakpoints* in the program's code. As we saw in an Example for the ENTER command, the '=address' option is used to tell DEBUG a starting location. If you use 'g' all by itself, execution will begin at whatever location is pointed to by the CS:IP registers. Optional *breakpoints* (meaning the program will HALT before executing the code at any of these locations) of up to any ten addresses may be set by simply listing them on the command line.

Requirements: Breakpoints can only be set at an address containing the first byte of a valid **8088/8086 Opcode**. So don't be surprised if picking some arbitrary address never halts the program; especially if you're trying to DEBUG a program containing opcodes DEBUG can't understand (that's anything 'requiring' a CPU above an 8088/8086)!