**Debugger Commands:**

From the DOS prompt type

```
DEBUG <filename>
```

where `<filename>` is the name of the machine language file.

Program enters into an environment waiting for a command. The prompt for a command is a single dash:

```
-
```

**Commands:**

- **-R** gives you a display of the registers and their contents (in hexadecimal)

- **-Rxy** where `xy` is a register name.
  1. First prints out the current contents of register `xy` in hexadecimal.
  2. Prints a `:` , waiting for you to enter a new hexadecimal number. If you enter a hexadecimal number and hit return, the register is modified to contain the new hexadecimal number. The previous data in the register is lost.
  3. If you do not enter a hexadecimal number when you see the `:` prompt, but instead just hit the `enter` key, the contents of the register are not changed.

Examples:

- **-RAX** reads/modifies the AX register
- **-RSI** reads/modifies the SI register.
- **-RDS** reads/modifies the DS register.

- **-T** (Trace) Steps the program forward 1 step, executing the command that begins at address `CS:IP`. The contents of the registers are modified according to what the instruction tells the computer to do, and the final states of the register just after the execution of the instruction are displayed.

- **-T x** (Trace x steps). Steps the program forward x steps, starting execution at the command that begins at address `CS:IP`. The contents of the registers are modified according to what the instructions tell the computer to do, and the final states of the register just after the execution of each instruction are displayed.

- **-G** (go) resumes execution of the program starting at the address `CS:IP`. Execution continues with each successive instruction, and in general will not stop at any point.

- **-G xxxx** (go to line xxxx). Resumes execution of the program beginning at `CS:IP` and continuing until just before the execution of line `CS:xxxx`. Line `CS:xxxx` is not executed. The contents of all registers are displayed just once, at final state of the registers, after IP is set to xxxx.
-a``name a file FFFFFF YYY``. This names the current file to be FFFFFF YYY. This only gives a name; it does not write the data.

-w Writes the current computer machine code, starting at CS:100+BX, and writing CX bytes, to a DOS filename FFFFFF YYY. Note that the name FFFFFF can only have 7 letters at most, and the suffix YYY can only have 3 letters as most.

-u unassemble computer program starting at present line. This back-converts the machine language code to mnemonics like ADD CX, BX and DIV AX.

-u yyyy unassemble computer program starting at line yyyy. This back-converts the machine language code to mnemonics like ADD CX, BX and DIV AX.

-a assemble computer program starting at present line. The computer waits for you to enter mnemonics like MOV AX,2 and SUB CX,BX. The computer converts the mnemonics from the text form to the actual machine language instructions for the computer processor. If you hit an enter without entering a new mnemonic, the program assumes you are finished and exits the assembler.

-a xxxx assemble computer program starting at line xxxx. The computer waits for you to enter mnemonics like MOV AX,2 and SUB CX,BX. The computer converts the mnemonics from the text form to the actual machine language instructions for the computer processor. If you hit an enter without entering a new mnemonic, the program assumes you are finished and exits the assembler.

-d display the contents of memory locations in segment DS, beginning at present line.

-d xxxx display the contents of memory locations beginning at DS:xxxx