* look into div, loop, and mul procedures
* look up meaning of DUP and when data segments start at certain addresses
* MAKE SURE PROGRAMMER CALCULATOR SAYS “WORD” FOR BINARY TO SIGNED INTEGER
  + “QWORD” FOR BINARY TO UNSIGNED
* two’s compliment of an integer is reversing (inverting) the bits and adding 1
* a 0-integer sign means positive, 1 means negative
* a word on x86 systems is 16 bits, doubleword is 32 bits
* largest unsigned integer that can be stored in 24 bits is 16,777,215
* extended index registers are ESI and EDI
* control bus uses binary signals to sync actions of all devices attached to system bus
* if a clock oscillates 10 billion times per sec, a single clock cycle is 1.0 x 10 ^ -10 sec
* real-address mode allows for 1MB of memory, protected mode allows 4GB
* 3 steps of instruction execution cycle = fetch, decode, execute
* lowest 8 bits of EDX register is DL
* the assembler reads assembly language source file and produces an object file
* assembly language instructions are executed at RUNTIME
* 6 instruction cycle steps
  + fetch instruction
  + increment instruction counter
  + decode
  + if instruction requires memory access…
  + execute
  + output operand…
* MOV imm,mem / mem,mem / imm,imm are not allowed
* MOVZX is for unsigned integers, MOVSX is for signed
* when DumpMem is called, the ESI register contains the starting address of data
* DumpRegs displays the CPU flags and 32-bit registers
* EDX contains the offset of a character array when calling GetCommandTail
* WriteHex writes an unsigned 32-bit integer to standard output in hexadecimal format
* what is a single instruction that inverts bits 5 and 6 in BL without changing any other bits? xor bl,1100000b
* the following are allowed formats for the MUL and DIV instruction
  + MUL mem32
  + MUL reg
  + MUL mem16
  + MUL mem8
  + DIV mem32
  + DIV mem16
  + DIV reg
  + DIV mem8
* the sizes of the sign, exponent, and significand for a Single Precision x86 floating point num are 1, 8, 23
  + for Double Extended Precision x86 floating point num it is 1, 16, 63
* Advantages of stack params vs. register params
  + Stack params reduce code clutter
  + Stack params are compatible with high-level languages
* passing arguments to procedures on the stack is more flexible