To import a library you must use the preprocessor statement:

|  |
| --- |
| INCLUDE Irvine32.inc |

Boilerplate:

|  |
| --- |
| ifndef X64  .686p ;Pentium Pro class of CPUs full instruction set  .XMM ;makes math calc faster, additional registers for vector ops  .model flat, C ; flat memory addressing, function conv agrees with C  endif |

Structure of an assembly language program

|  |
| --- |
| ; Program description  ; Author  ; Creation date  ; Revisions  ; Date  .data  ; (insert variables here)  .code  main PROC  ; (insert executable instructions here)  exit ; ends the program  main ENDP  ; (insert additional procedures here)  END main |

To write a comment, precede it with a semi-colon. When writing multi-line comments you can use the COMMENT keyword followed by either an exclamation mark or an ampersand, then the commend, then another exclamation mark or ampersand.

|  |
| --- |
| COMMENT !  this is a comment written in the assembly language  !  COMMENT &  using an ampersand instead of an exclamation mark also works  & |

The .data section includes information about variable assignment during the runtime of the program:

|  |
| --- |
| .data |

Defining variables:

Within the data section you can define variables using the following syntax:

[variable name] [data type] ?

You can also assign values using the following syntax:

[variable name] [data type] [data]

Data types:

* BYTE
  + This data type is used to store values in 8-bits, it can be used to store characters or strings if used properly.
  + To let the computer know when you want the string to end you must use a break condition, this can be done by adding a ‘, 0’ after the assigned value.

|  |
| --- |
| ExampleString BYTE "This is an example string", 0 |

* + This data-type can also be used for storing 8-bit integers.
* SBYTE

|  |
| --- |
| Num1 BYTE ? ; this is an 8-bit integer  Num2 SBYTE ? ; this is an 8-bit signed-integer  NumD BYTE 18 ; this integer value is written in decimal  NumH BYTE 12h ; this integer value is written in hexadecimal  NumB BYTE 10010b ; this integer value is written in binary |

* WORD (word) [2-bytes]
  + Stores a 16-bit integer value in memory.
* DWORD (double word) [4-bytes]
  + Stores a 32-bit integer value in memory.
* SDWORD (signed double word) [4-bytes]
  + Stores a 32-bit integer value in memory, the first bit is used as a sign bit (two’s compliment)

|  |
| --- |
| Num1 WORD ? ; this is a 16-bit integer  Num2 DWORD ? ; this is a 32-bit integer  Num3 SDWORD ? ; this is a 32-bit signed-integer |

Creating arrays in assembly:

|  |
| --- |
| Array1 BYTE 10, 20, 30, 40 ; this is an array with 4 8-bit values  Array2 QWORD 100, 200, 300, 400 ; this is an array with 4 64-bit values |