LECTURE 13 - Program Development using TASM

Step	Input	Program	output
Edit the program	Key board	Editor	myfile.asm
Assemble the program	myfile.asm	MASM or TASM	myfile.obj
Link the program	myfile.obj	LINK or TLINK	myfile.exe

Structure of an Assembly Language Program Using TASM

```
.model small                    ; Select a memory model
.stack stack_size               ; Define the stack size
.data

                                ; Variable and array declarations
                                ; Declare variables at this level
.code
main proc

                                ; Write the program main code here
main endp
;Other Procedures

                                ; Always organize your program
                                ; into procedures
end main                        ; To mark the end of the source file
```

Simple Assembly Language Program

```
        .MODEL SMALL
        .STACK 64
        .DATA
DATA1   DB 52h
DATA2   DB 29h
SUM     DB ?
        .CODE
MAIN    PROC FAR
        MOV AX,@DATA
        MOV DS,AX
        MOV AL,DATA1
        MOV BL,DATA2
        ADD AL,BL
        MOV SUM,AL
        MOV AH,4Ch
        INT 21h
MAIN    ENDP
END MAIN
```

Title directive:

The title directive is optional and specifies the title of the program. Like a comment, it has no effect on the program.

The Model directive:

The model directive gives information on how much memory the assembler would allocate for the program. This depends on the size of the data and the size of the program or code.

LECTURE 13 - Program Development using TASM

Segment directives:

Segments are declared using **directives**. The following directives are used to specify the following segments:

Stack Segment: .stack

.stack followed by a value that indicates the size of the stack

Stack addresses are computed as offsets into this segment

Data Segments: .data

.data followed by declarations of variables or definitions of constants.

Used to set aside storage for variables.

Constants are defined within this segment in the program source.

Code Segment: . code

The code segment contains executable instructions and calls **procedures**.

Data Types and Data Definition

```
DATA1     DB 25   ; Dec
DATA2     DB 10001001b
DATA3     DB 12h
ORG 0010h
DATA4     DB "2591"
ORG 0018h
DATA5     DB ?
```

This is how data is initialized in the data segment

```
0000 19
0001 89
0002 12
0010 32 35 39 31
0018
```

Other Definitions - Examples

```
DB 6 DUP(FFh)          ;duplicate fill 6 bytes with FFh
DW 954
DW 253Fh
DW 253Fh,'HI'          ; allocates two bytes
DD 5C2A57F2h           ;allocates four bytes
DQ "HI"                ;allocates eight bytes
COUNT EQU 25
COUNTER1 DB COUNT
COUNTER2 DB COUNT
```