Data Definitions

A data definition statement in assembly language is used to reserve memory for a variable and to specify its data type. The general syntax for a data definition statement is as follows:

[label] directive value

Where:
- **label** is an optional name for the variable.
- **directive** is the data type of the variable.
- **value** is the initial value of the variable.

For example, the following data definition statement reserves 4 bytes of memory for a variable named count and initializes it to the value 12345:

```
count DWORD 12345
int count = 12345;
```

So, **label is variable, directive is the datatype, and value is the value.**

The following are some other examples of data definition statements in assembly language:

```
message DB "Hello, world!"
age BYTE 25
salary SDWORD 100000
```

The first statement reserves 13 bytes of memory for a variable named message and initializes it to the string "Hello, world!".

The second statement reserves 1 byte of memory for a variable named age and initializes it to the value 25. The third statement reserves 4 bytes of memory for a variable named salary and initializes it to the value 100000.

It is important to note that the data type of a variable must be specified in the data definition statement. This is because the assembler needs to know how much memory to reserve for the variable and how to interpret its value.

Data definition statements in assembly language are similar to creating variables in C programming. eg

```
count DWORD 12345
```

is equivalent to the following variable declaration in C programming:

```
int count = 12345;
```

- **BYTE (8 bits)**: Short form - **B**
- **SBYTE (8 bits, signed)**: Short form - **SB**
- **WORD (16 bits)**: Short form - **W**
- **SWORD (16 bits, signed)**: Short form - **SW**
- **DWORD (32 bits)**: Short form - **D**
- **SDWORD (32 bits, signed)**: Short form - **SD**
- **FWORD (48 bits)**: Short form - **FW**
- **QWORD (64 bits)**: Short form - **Q**
- **TBYTE (80 bits)**: Short form - **T**
- **REAL4 (32-bit floating-point)**: Short form - **F**
- **REAL8 (64-bit floating-point)**: Short form - **FF**
- **REAL10 (80-bit floating-point)**: Short form - **FT**

For example, the following data definition statement is equivalent to the one above:

```
message DB "Hello, world!"

message DB "Hello, world!", 0
```

The second statement explicitly specifies a **null terminator (0)** at the end of the string. This is not required, but it is good practice to do so.

Table 3-3 Legacy Data Directives.

Directive	Usage
DB	8-bit integer
DW	16-bit integer
DD	32-bit integer or real
DQ	64-bit integer or real
DT	define 80-bit (10-byte) integer