```
   s.str = "Hello"
   Subroutine(s)
End
```

As this example shows, you define a structure type using the Structure keyword in a procedure file. You define a local structure variable using the STRUCT keyword in a user-defined function. And you pass a structure from one user-defined function to another as a pass-by-reference parameter, as indicated by the use of & in the subroutine parameter declaration.

Defining Structures

Structures are defined in a procedure file with the following syntax:

```
Structure structureName
   memType memName [arraySize] [, memName [arraySize]]
   …
EndStructure
```

Structure member types (*memType*) can be any of the following Igor objects: Variable, String, WAVE, NVAR, SVAR, DFREF, FUNCREF, or STRUCT.

Igor structures also support additional member types, as given in the next table, for compatibility with C programming structures and disk files.

Igor Member Type	C Equivalent	Byte Size	Note
char	signed char	1	
uchar	unsigned char	1	
int16	16-bit int	2	
uint16	unsigned 16-bit int	2	
int32	32-bit int	4	
uint32	unsigned 32-bit int	4	
int64	64-bit int	8	Igor7 or later
uint64	unsigned 64-bit int	8	Igor7 or later
float	float	4	
double	double	8	

The Variable and double types are identical although Variable can be also specified as complex using the /C flag.

The optional *arraySize* must be specified using an expression involving literal numbers or locally-defined constants enclosed in brackets. The value must be between 1 and 400 for all but STRUCT where the upper limit is 100. The upper limit is a consequence of how memory is allocated on the stack frame.

Structures are two-byte aligned. This means that if an odd number of bytes has been allocated and then a nonchar field is defined, an extra byte of padding is inserted in front of the new field. This is mainly of concern only when reading and writing structures from and to disk files.

The Structure keyword can be preceded with the **Static** keyword (see page V-906) to make the definition apply only to the current procedure window. Without the Static designation, structures defined in one procedure window may be used in any other.

Structure Initialization

When a user-defined function that declares a structure is called, the structure fields are automatically initialized. Numeric fields are initialized to zero. String, WAVE, NVAR, SVAR, DFREF, and FUNCREF fields are initialized to null.