Lecture 15.1

Topics:

1. Interpretation of Complex Declarations

1. Interpretation of Complex Declarations

By using the mix of several built-in and derived data types, new programming declarations can be achieved. It is important to understand a declaration and what it is and how to use it.

1.1 Right—Left Rule

There are many ways or rules that can be used to interpret the programming declarations. One of these rules is a so-called **right-left** rule. This rule will

- Start with the identifier in the center of the declaration, and
- Read the declaration by alternatively going first to the right, and then the left until all entities are read.

Note that the read-expression needs to be balanced with the identifier at the center. One may need to fill in a place holder (i.e., a symbol **#**) to balance the read-expression.

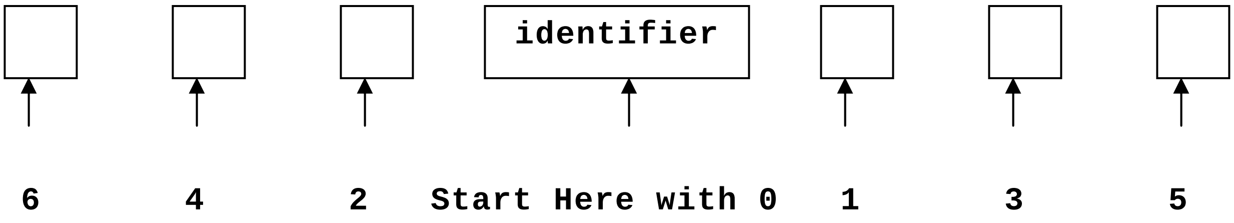

Figure 1 Right—Left Rule

1.2 Examples

In the following cases, the programming declarations are translated to read-expressions before performing the reading to interpret them.

Case #1

```
int iX;          → Declaration
int iX #         → Read-expression
 2   0  1

  iX is # an int.   → # is a place-holder and treated as blank
Or,
  iX is an int.
```

Case #2

```
int* iPtr;        → Declaration
```

```
int * iPtr # #    → Read-expression
 4  2  0   1 3
```

iPtr is # a pointer to # an int.   → # is treated as a blank

Or,

iPtr is a pointer to an int.

Case #3

```
int iAry[ 5 ];    → Declaration
int iAry [ 5 ]    → Read-expression
 2   0     1
```

iAry is a [5] array of int.

Or,

iAry is an array of 5 values of int.

Case #4

```
int iAry2[ 5 ][ 6 ];  → Declaration
int iAry2 [ 5 ][ 6 ]  → Read-expression
 2    0       1
```

iAry2 is a [5][6] array of values of int.

Or,

iAry2 is a [5][6] array of 5 rows, where each row has 6 column of values of int.

Case #5

```
int* iAryOfPtr[ 5 ];       → Declaration
int * iAryOfPtr [ 5 ] #    → Read-expression
 4  2     0       1   3
```

iAryOfPtr is an [5] array of pointers to # int.

Or,

iAryOfPtr is an array of 5 pointers to int.

Case #6

```
int ( * iPtrToArray ) [ 5 ];     → Declaration
int ( * iPtrToArray # ) [ 5 ]    → Read-expression
 4    2      0      1     3
```

iPtrToArray is a pointer to [5] array of values of int.

Or,

iPtrToArray is a pointer to an array of 5 values of int.

<u>Case #7</u>

```
int* foo( argList );       → Declaration
int * foo ( argList ) #    → Read-expression
 4  2  0       1      3
```

foo is a function that returns a pointer to # an int.

Or,

foo is a function that returns a pointer to an int.