

## Basis Path Testing

```

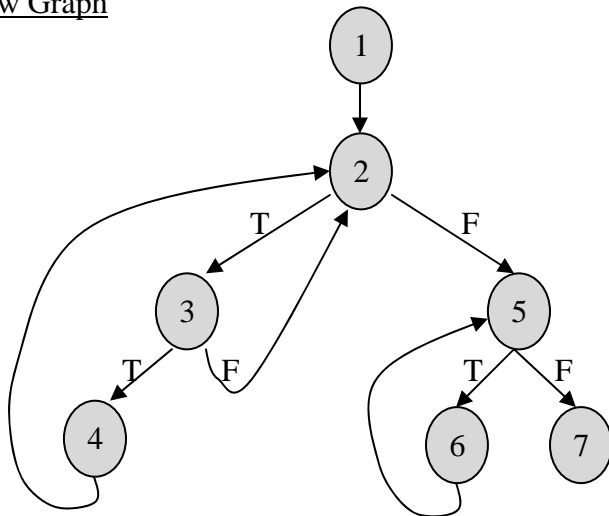
procedure delete_element (int value, int array_size, int array[])
{
1  int I;                /* loop counter */
   location = array_size + 1; /* location of value to delete */

   /* find the location of the value */
2  for I = 1 to array_size
3     if ( array[I] == value )
4         location = I;
       end if;
   end for;

   /* move each element after the target by one space */
5  for I = location to array_size
6     array[I] = array[I+1];
   end for;
7  array_size --;
}

```

### Flow Graph



Basis Paths	value	array size	array contents	result
1 2 5 7	na	0	na	PASS
1 2 5 6 7	<b>impossible path</b> - to skip node 3, size must be 0, but then 5 always goes to 7			
1 2 3 2 5 6 7	<b>impossible path</b> - "1 2 3 2" size=1 and value not in array, but then 5 would go to 7			
1 2 3 4 2 5 6 7	10	1	10	FAIL - array out of bounds

*the follow tests do exercise each decision as both True and False*

1 2 5 7	na	0	na	PASS
1 2 3 2 5 7	10	1	20	PASS
1 2 3 4 2 5 7	<b>impossible path</b> - node 4 sets "location" which forces 5 to go to 6			
1 2 3 4 2 5 6 7	10	1	10	FAIL - array out of bounds

## Boundary Value Analysis

### Input Boundaries

value:

- in the list
- not in the list

size:

- 0
- 1
- max
- max+1

array:

- all different values
- all the same values

### Output Boundaries

- array remains unchanged
- array is changed

### Test Cases

value	size	array contents
<i>in the list, all different values</i>		
20	0	10 20 30 ... 1000
20	1	10 20 30 ... 1000
20	100	10 20 30 ... 1000
20	101	10 20 30 ... 1000
<i>not in the list, all different values</i>		
-999	0	10 20 30 ... 1000
-999	1	10 20 30 ... 1000
-999	100	10 20 30 ... 1000
-999	101	10 20 30 ... 1000
<i>in the list, all the same values</i>		
10	0	10 10 10 ... 10
10	1	10 10 10 ... 10
10	100	10 10 10 ... 10
10	101	10 10 10 ... 10
<i>not in the list, all the same values</i>		
20	0	10 10 10 ... 10
20	1	10 10 10 ... 10
20	100	10 10 10 ... 10
20	101	10 10 10 ... 10