Complicated declarations in C

Most of the times declarations are simple to read, but it is hard to read some declarations which involve pointer to functions. For example, consider the following declaration from "signal.h".

```
void (*bsd_signal(int, void (*)(int)))(int);
```

Let us see the steps to read complicated declarations.

1) Convert C declaration to postfix format and read from right to left.
2) To convert expression to postfix, start from innermost parenthesis, If innermost parenthesis is not present then start from declarations name and go right first. When first ending parenthesis encounters then go left. Once whole parenthesis is parsed then come out from parenthesis.
3) Continue until complete declaration has been parsed.
Let us start with simple example. Below examples are from "K & R" book.

```
1)  int (*fp) ();
```

Let us convert above expression to postfix format. For the above example, there is no innermost parenthesis, that's why, we will print declaration name i.e. "fp". Next step is, go to right side of expression, but there is nothing on right side of "fp" to parse, that's why go to left side. On left side we found "*", now print "*" and come out of parenthesis. We will get postfix expression as below.

```
fp  *  ()  int
```

Now read postfix expression from left to right. e.g. fp is pointer to function returning int

Let us see some more examples.

```
2) int (*daytab)[13]
```

Postfix : daytab * [13] int
Meaning : daytab is pointer to array of 13 integers.

```
3) void (*f[10]) (int, int)
```

Postfix : f[10] * (int, int) void
Meaning : f is an array of 10 pointer to function(which takes 2 arguments of type int) returning void

```
4) char (*(*x())[]) ()
```

Postfix : x () * [] * () char

Meaning : x is a function returning pointer to array of pointers to function returnging char

```
5) char (*(*x[3])())[5]
```

Postfix : x[3] * () * [5] char

Meaning : x is an array of 3 pointers to function returning pointer to array of 5 char's

```
6) int *(*(*arr[5])()) ()
```

Postfix : arr[5] * () * () * int

Meaning : arr is an array of 5 pointers to functions returning pointer to function returning pointer to integer

```
7) void (*bsd_signal(int sig, void (*func)(int)))(int);
```

Postfix : bsd_signal(int sig, void(*func)(int)) * (int) void

Meaning : bsd_signal is a function that takes integer & a pointer to a function(that takes integer as argument and returns void) and returns pointer to a function(that take integer as argument and returns void)