14.2

a. To restore the frame pointer of the caller program.

b. To continue the caller program with the return address.

c. To return the result of the callee program to the caller program.

14.4

a. The caller function.

b. The callee function.

c. The callee function.

d. The callee function.

14.15

xEFF6	16 (x1 in f)
xEFF7	main’s frame pointer
xEFF8	x3103 (return address)
xEFF9	0 (return value)
xEFFA	4 (x in f)
xEFFB	5 (y in f)
xEFFC	6 (z in f)
xEFFD	6 (c in main)
xEFFE	5 (b in main)
xEFFF	4 (a in main)

17.2

No, because the recursive function can be called anywhere and hence the return address is not always the same.

17.5

a.

(1) The result is 0.

(2) The result is 2.

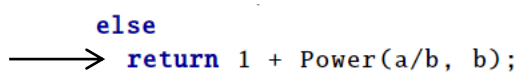
(3) The result is 0.

b. If a>0 and b>0, the function Power computes $\lfloor \log_b a \rfloor$.

c.

R6 ⟶	frame pointer
	return address to the caller function Power
	0 (return value)
	1
	7
	frame pointer
	return address
	saved for return value
	11
	7

(The address should refer to **return 1 + Power(a/b, b);**)



17.7

a. 2048. Because the activation record of SevenUp occupies 8 bytes of memory and the stack can occupy up to 16 Kbytes of memory, from which we can have the largest value should be 2048. In fact, the value should be a little bit smaller because of the activation record of main.

b. 512. Because the activation record of SevenUp occupies 8 bytes of memory and the stack can occupy up to 4 Kbytes of memory, from which we can have the largest value should be 512. In fact, the value should be a little bit smaller because of the activation record of main.