Ex 1.8:

```
float rsum(float list[], int n)
{
  if(n) return rsum(list, n-1) + list[n-1];
  return 0;
}
```

- This means that the compiler must save the parameters, the local variables, and the return address for each recursive call.
- The space needed for one recursive call is the number of bytes reqd. for the two parameters and the return address.
- Assume that an integer and a pointer each require 4 bytes.

Type	Name	Number of bytes
parameter: array pointer	list[]	4
parameter: integer	n	4
return address: (used internally)		4
Total per recursive call		12

- If the array has n = MAX_SIZE numbers, the total variable space needed for the recursive version is $S_{rsum}(MAX\_SIZE) = 12 * MAX\_SIZE$
- The recursive version has a far greater overhead.

Exercises 1.5.1

1) Determine the space complexity of the iterative and recursive factorial functions created in Exercise 7, Section 1.3

Soln: The iterative_factorial function(int n) has only fixed space requirements. No structured variable exists. $\therefore S_{iterative\_factorial}(n) = 0$

The only variable is the input n which takes up a fixed space of 4 bytes.

- Now we analyze the long recursive_factorial(int n) function.
  : Assume that the return address has 4 bytes.