R-1.7

$1/n$

2^{100}

$\log \log n$

sqrt($\log n$)

$\log^2 n$

$n^{0.01}$

sqrt(n), $3\,n^{0.5}$

$2^{\log n}$, $5\,n$

$n \log_4 n$, $6\,n \log n$

$2\,n \log^2 n$

$4\,n^{3/2}$

$4^{\log n}$

$n^2 \log n$

n^3

2^n

4^n

2^{2n}

R-1.14

$1 + 2 + 3 + 4 + \ldots. + 2n = (2n(2n + 1)) / 2 = O(n^2)$

R-1.15

$1 + 2 + 3 + 4 + \ldots. + n^2 - 1 + n^2 = (n^2(n^2 + 1)) / 2 = O(n^4)$

R-1.17

$O(max\{f(n), g(n)\}) = O(f(n) + g(n))$

$f(n) + g(n) <= k * max(f(n), g(n))$

$f(n) <= max(f(n), g(n))$ for $n>=1$

$g(n) <= max(f(n), g(n))$ for $n>=1$

Combining both, we get

$f(n) + g(n) <= 2 * max(f(n), g(n))$ for $n>= 1$

$f(n) + g(n) = O(max(f(n), g(n)))$

Also

$f(n)+g(n) >= max\{f(n),g(n)\}$

Therefore

$O(max\{f(n), g(n)\}) = O(f(n) + g(n))$

R-1.26

$d(n)$ is $O(f(n)) \Rightarrow d(n) <= k1 * f(n)$

$e(n)$ is $O(g(n)) \Rightarrow e(n) <= k2 * g(n)$

Combining both, we get

$d(n) * e(n) = k1 * f(n) * k2 * g(n)$

$d(n) * e(n) = k * f(n) * g(n)$ where $k = k1 * k2$ (a constant)

hence $d(n) * e(n) = O(f(n) * g(n))$

C-1.1

```
start[0...n]     //Initialize with 0
end[0...n]
M_0 = 0

for t ⇐ 1 to n do
    if (M_{t-1} + A[t] > 0)
        start[t] = start[t - 1]
    else
        start[t] = t
    M_t ⇐ max(0, M_t + A[t])

m = 0
k = 0
for t ⇐ 1 to n do
    if m < M_t
        m = M_t
        k = t
        end[t] = t
return (m, start[k], end[k])
```

C-1.2

```
m = 0
M = 0
for t ⇐ 1 to n do
    M ⇐ max(0, M + A[t])
    m ⇐ max(m, M)
return M
```

C-1.4

0 0 0 1	ones bit changes n times
0 0 1 0	twos bit changes n / 2 times
0 0 1 1	fours bit changes n / 4 times
0 1 0 0	eights bit changes n / 8 times
0 1 0 1	….
0 1 1 0	
0 1 1 1	
1 0 0 0	Total time = n + n / 2 + n / 8 + …= Σ n / 2^i where i = 0 to a

= n Σ 2^{-i} (where Σ 2^{-i} is an infinite geometric series with sum ~ 2)

Total time < 2n =O(n)

C-1.12
$\log_b f(n)$ is $O(\log f(n))$

$\log_b$**x =** $\log_b c * \log_c x = \log_c x / \log_c b$

Consider b = 10
$\log_{10} n = \log_2 n / \log_2 10 = (1 / \log_2 10) * \log_2 n$

Since $(1 / \log_2 10)$ is a constant

$\log_{10} n = c (\log_2 n) = O(\log_2 n)$

Hence
$\log_b f(n)$ is $O(\log f(n))$

C-1.13

n = # of elements

If n is even ⇒ split all elements into pairs, find max and min of each pair. Put all maximums into one group and all minimums into another. This will take n / 2 comparisons.
Find maximum from max group and minimum from min group. Since each group has n / 2 elements, this will result in n / 2 + n / 2 comparisons

So total comparisons will be n / 2 + n / 2 + n / 2 = 3n/2

If n is odd ⇒ Leave the last elements and apply the same technique to find maximum and minimum. In the end compare the last element with both maximum and minimum and update them accordingly.
Total comparisons will be = 3n/2 - 2