Ex. 28.2-3

By Strassen's algorithm, if the multiplications are k times, we can get the time in

$T(n) = k\,T(n/3) + \Theta(n^2)$

$T(n) = \Theta(n^{\log_3 k}) = o(n^{\lg 7})$

$\log_3 k < \log_2 7$

$\log k < (\log 7 * \log 3) / \log 2$

$k < 21.849862$

the largest k is 21.

Time analysis： $\Theta(n^{\log_3 k}) = \Theta(n^{2.771244})$

Prob. 31-1

assume $c = \gcd(a, b)$, $a = c*m$, $b = c*n$ and $\gcd(m, n) = 1$

a.

if a , b are both even, c is also even.

let $c = 2*r$

$a = 2*r*m$

$b = 2*r*n$

$\gcd(a/2, b/2) = \gcd(r*m, r*n)$

since $\gcd(m, n) = 1$, $\gcd(r*m, r*n) = \gcd(r, r) = r$

so $\gcd(a, b) = 2 * \gcd(a/2, b/2)$

b.

if a is odd and b is even, c will be odd

since b is even and c is odd

$b = c*n = c*(2*n')$

$\gcd(m, n) = \gcd(m, 2*n') = 1$

$\rightarrow \gcd(m, n') = 1$

so $\gcd(a, b/2) = \gcd(c*m, c*n') = c$

c.

if a and b are both odd

$gcd((a-b)/2, b) = gcd((m-n)*c/2, c*m) = c * gcd((m-n)/2, m)$

assume $gcd(m - n, m) = k > 1$

$m – n = k * p$

$m = k * q$

$n = k * (q - p)$

contribute that $gcd(m, n) = 1$

so $gcd(m-n, m) = 1$, $gcd((m-n)/2, m) = 1$

$gcd((a-b)/2, b) = c$

d.

```
gcd(a, b){
  if (b = 0)
      return a
  if (a is even)
      if (b is even)
          return 2*gcd(a/2, b/2)
      else if (a/2 > b)
          return gcd(a/2, b)
      else
          return gcd(b, a/2)
  else if (b is even)
      return gcd(a, b/2)
  else if ((a-b)/2 > b)
      return gcd((a-b)/2, b)
  else
      return gcd(b, (a-b)/2)
}
```

In the last two case：

$(a - b)/2 > b \rightarrow (a - b)/2 < a/2$

$(a - b)/2 \leq b \rightarrow a \leq 3b$

if $a > 2*b$, the first part is reduced more than half

else, the second part is reduced more than half

Since each recursive need at most 3 parity test => $O(1)$

And in the worst case, each turn will be reduced a or b in half.

So $T = O(\log a) + O(\log b) = O(\log a)$