Date ______
Page ______

Omega Notation (Ω)

$f(n) = \Omega(g(n))$ iff there exists constant c (where $c > 0$) and n_0 (where $n_0 \geq 0$) such that $\{c\,g(n) \leq f(n)\}$ for all $n \geq n_0$.

$f(n) = 2n+3 = \Omega(n)$

$c = 1$

$n \leq 2n+3$

$-3 \leq n$

$n_0 = 0$

★ In big O you can write higher growing terms for OOG and in Omega you can write lower growing terms.

If $f(n) = O(g(n))$

$g(n) = \Omega(f(n))$

(Θ) Theta Notation — Exact bound of an algo, fun....

Direct Method:

$\cancel{1}000n^2 + 100n\log n + 2n$

$= \Theta(n^2)$

$2000n + 2\log n$

~~$= \Theta(\log n)$~~ $\Theta(n)$

If $f(n) = \Theta(g(n))$

then,

$f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$

$g(n) = O(f(n))$ and $g(n) = \Omega(f(n))$

y- $f(n) = 2n^2 + n$
: $\Theta(n^2)$
$O(n^2)$
$\Omega(n^2)$

* Represents exact bound.

$\{100, 10^5, \log 2000 \ldots\} \in \Theta(1)$
$\{100n + 2n\log n, 5n+3 \ldots\} \in \Theta(n)$
$\{2n^2, \frac{n^2}{4}, 5n\log n \ldots\} \in \Theta(n^2)$

Analysis of Common loops:

1. for (int i=1; i<n; i=i×c)
{
// Some $\Theta(1)$ work
}

$c^0, c, c^2, \ldots \; c^{k-1}$
$c^{k-1} < n$
?? $k < \log_c n + 1$

Time complexity (TC) = $\Theta(\log_c n)$

2. for (int i=n; i>1; i=i/c)
{
Some $\Theta(1)$ work
}

$n/c^0, n/c, n/c^2 \ldots \; n/c^{k-1}$

$\frac{n}{c^{k-1}} > 1$

$c^{k-1} < n$

$k-1 \leq \log_c n$

$k < \log_c n + 1$

$\boxed{\Theta(\log_c n)}$

3. for (int i=2 ; i<n; i=pow(i, c))
{

some Θ(1) work

}

$2, 2^c, (2^c)^c, \ldots \ldots (((2^c)^c)^c)^{c\ldots}$

$2^{c^0} \quad 2^{c^1} \quad 2^{c^2} \ldots \ldots 2^{c^{k-1}}$

$2^{c^{k-1}} < n$

$c^{k-1} \leq \log_2 n$

$k-1 \leq \log_c \log_2 n$

$k < \log_c \log_2 n + 1$

$\boxed{\Theta(\log_2 \log_c n)}$ or $\boxed{\Theta(\log \log n)}$

Analysis on multiple loops

→ subsequent loop.

```
void fun (int n)
{
  for (i=0; i<n; i++)
  {
    some Θ(1) work
  }
  for (i=1; i<n; i=i×2)
  {
    some Θ(1) work
  }
  for (i=1; i<100; i++)
  {
    some Θ(1) work.
  }
}
```

Θ(n) + Θ(log n) + Θ(1)

Θ(log n) + Θ(1)

$\Theta(n) + \Theta(\log n) + \Theta(1)$

$\boxed{\Theta(n)}$

Nested loop:

~~void fun()~~

```
void fun(int n)
{
  for(int i=0; i<n; i++)              → Θ(n)
  {                                      *
    for(int j=1; j<n; j=j*2)          → Θ(log n)
    {
      some Θ(1) work
    }
  }
}
```

$\Theta(n) * \Theta(\log n)$

$= \boxed{\Theta(n \log n)}$

Mixed loops:

```
void fun(int n)
{
  for(i=0; i<n; i++)
  {
    for(j=1; j<n; j=j*2)
    {
      some Θ(1) work
    }
}
```

$\Theta(n \log n)$

```
for(int i=0; i<n; i++)        ] Θ(n)
{                                *          = Θ(n^2)
   for(int j=1; j<n; j++)      ] Θ(n)
   {
      // some Θ(1) work
   }
}
```

Total = ~~Θ(n logn)~~ + Θ(n²) ✓

T.C. $\boxed{\Theta(n^2)}$

<u>Different input:</u>

```
void fun(int n, int m)
{
   for(i=0; i<n; i++)
   {
      for(j=1; j<n; ~~j++~~ j=j*2)     ] Θ(n log n)
      {
         some Θ(1) work
      }
   }
   for(i=0; i<m; i++)          ] Θ(m)
   {                             * ~~Θ(mlogm)~~
      for(j=1; j<m; j++)       ] Θ(m)
      {                                  = Θ(m^2)
         some Θ(1) work
      }
   }
}
```

T.C. $\boxed{\Theta(n\log n + m^2)}$