Date: __/__/__

MON TUE WED THU FRI SAT SUN

L-05

input 1

5 20

80 51

79

11

40

```
search value (Num, value) {
//Go linearly through all entries
// compare value with each element
for (iter = 0, iter < Num.len, iter++)
{
    if (Num[iter] == value) {
        return iter
    }
}
return -1    } C2
}
```

$C_1 \cdot n + C_2$

iter Num[iter]

① 70 , 20

51

How to convert relation to function

→ Consider only Worst case
→ Average case
→ Best case

Worst case: value absent / value present at End

flight

industrial plant

$f(n) \in \Theta(g(n))$

f maps problem size to resource (time) consumptions

$c_1 \quad c_2 \quad n_0$

$0 \le c_1 g(n) \le f(n) \le c_2 g(n) \quad \forall n \ge n_0$

$f(n) = n^2 + 2n + 5$

$g(n) = n^2$

$g(n) = n$

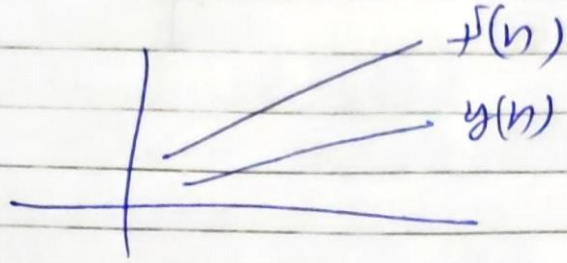
$f(n) = n^2 + 2n + 5 \quad g(n) = n^2 \quad c_1 = 1, c_2 = 8, n_0 = 1$

n	f(n)	g(n)	$c_1 g(n)$	$c_2 g(n)$
0	5	0	0	0
1	8	1	1	8
2	13	4	4	32
3	20	9	9	72
4	29	16	16	128
5	40	25	25	200

$f(n) \in \Theta(g(n)) \quad c_1 = 1 \quad c_2 = 8 \quad n_0 = 1$

$f(n) \in O(g(n))$ c n_0

$0 \leq f(n) \leq c \cdot g(n) \quad \forall n \geq n_0$



$f(n) = n^2 + 5n + 2 \quad g(n) = n^2 \quad f(n) \in O(g(n))$?

$c \quad n_0$

$f(n) = 5n + 2 \quad g(n) = n^2 \quad c = 2 \quad n_0 = 3$

n	$5n+2$	n^2	$2 \cdot n^2$	$0.5n^2$
0	2	0	0	0
1	7	1	2	0.5
2	12	4	8	2
3	17	9	18	4.5
4	22	16	32	8
5	27	25	50	12.5
6	32	36		18

$c = 0.5$

L-6

Θ asymptotic =

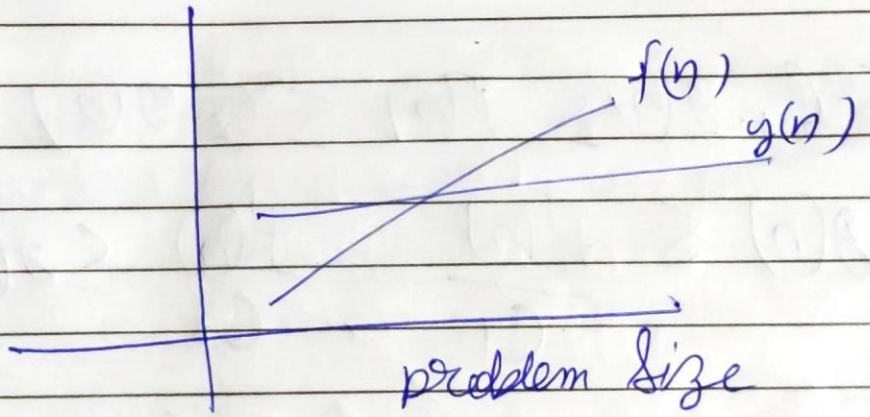
O asymptotic upper bound ≤

Ω asymptotic ≥

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

c, n_0

$0 \leq c \cdot g(n) \leq f(n) \quad \forall n \geq n_0$



$f(n) = n^2 + 5n + 2, \quad g(n) = n^2 \quad f(n) \in \Omega(g(n))$ if $c = 1, n_0 = 0$

$f(n) \geq c\,g(n) \qquad c = 500$

Transitivity.

$A : B, \; B : C \rightarrow A : C$

$f(n) \in \Theta(g(n)), \; g(n) \in \Theta(h(n)) \rightarrow f(n) \in \Theta(h(n))$

Reflexility

A:

$f(n) \in O(f(n))$

$f(n) \in O(f(n))$

$f(n) \in \Omega(f(n))$

Symmetry $A; B \leftrightarrow B; A$

$f(n) \in O(g(n)) \leftrightarrow g(n) \in O(f(n))$

$0 \leq C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n) \quad \forall n \geq n_0$

$g(n) \leq \frac{f(n)}{C_1} \qquad \frac{f(n)}{C_2} \leq g(n) \quad \forall n \geq n_0$

$\frac{1}{C_1} = C_1' \qquad \frac{1}{C_2} = C_2'$

Symmetry $f(n) \in O(g(n)) \leftrightarrow g(n) \in \Omega(f(n))$ false

counter example

Fibonacci Numbers

fib(1) = 0 & fib(2) = 1 fib(n) = fib(n-1) + fib(n-2)

FibR(n) $T(n) = T(n-1) + T(n-2)$

$T_n \approx (1.6)^n$

n	Fib(n)
1	0
2	1
3	1
4	2
5	3
6	5
7	8
8	13

```
if (n==1){
    return 0
}
if (n==2){
    return 1
}
answer = FibR(n-1) + FibR(n-2)
return (answer)
```

recursion

```
Fibs (n){
    if (n==1){
        return 0
    }
    if (n==2){
        return 1
    }
```

first case

To compute nth number start from 3rd number and then compute 4th then 5th
|
till you reach nth

```
Fibs(n) {
      if(n==1){
            return 0
      }
      if (n==2) {
            return 1        → fib(n-1)
      }                            fib(n-2)
 4    fibn1 = 1      ↙
      fibn2 = 0

      for (curr=3, curr≤n, curr=curr+1){
 C2.n       answer = fibn1 + fibn2
            fibn2 = fibn1
            fibn1 = answer
      }
      return answer  1 C3
}
```

$T(n) \in \Theta(n)$

$T(n) = C_1 + \underline{C_2 . n} + C_3$

$C_2.(n-2)$

10^{10} ins/sec → 10 sec

n	FibS	FibR
1	1	$(1.6)^1 \to 1.6$
10	10	$(1.6)^{10} \to 10^1$
100	100	$(1.6)^{100} \to 10^{20} \to 10^{\frac{1}{2}}$ sec
1000	1000	$(1.6)^{1000} \to 10^{200} \to 10^{\frac{1}{2}}$ sec
10000	10000	$(1.6)^{10000} \to 10^{2000}$
	↓	$\to 10^{1000}$ sec
	less than 1 sec.	

$T(n) \leq (1.6)^n$

↓

$2T(n-1)$

2^n

L-7

ADT
Abstract Data Types

Abstractions

Bits → Digital circuit
Files → Secondary storage
Machine code → operations by processor
Basic data types → data handled by processor

→ Ease of understanding
→ Independence from implementation
↳ Separation of concerns

Specification and implementation

⇒ Ability to build more abstraction

Abstract Data Type (ADT)

→ Data
→ Operations

int → integer

Data → 16/32/64 (bits) → wires → transistors

Operations → +, −, ×, /, ++, −−

- Access to data is only through getters
- Implementation details need not be shared
- Naturally translates to object oriented programming languages.

Array ADT

Data: <index, value>

Collection of pairs

Operations:
- CreateNew (capacity, type)
- StoreValue (index, val)
- access value (index) — core definition

Search Value (value) — might have

init Array (value)

- ✓ Store same type of data
- ✗ contiguous storage
- ✗ fixed storage
- ✓ Access element with index
- Most of the programming languages provide it
- We can have arrays of arrays
- storing element at any index
- ✓ When element at any index
- ✗ structure fix