27/12/2022
DS
1st class

Data Structure

Data → Organized or unorganized

Information → Collection of organized data or processed data that are meaningful.

DS → organized collection of related data

DS
- Sorting
- Searching
- Array
  - Stack (LIFO)
  - Queue (FIFO)
- Linked list
- Graph
  - BFS
  - DFS

* Complexity

```
int a = 5;      10
int b = 10;     5

a * b = 50
```

~~a+b~~

```
a = a+b   = 15
b = a-b   = 5
a = a-b   = 15-5
          = 10
```

15 10 15-10

```
a = a * b;
b = a/b;
a = a/b;
```

31/12/2022
DS
2nd Class

Book

Data Structures
- Seymour Lipschutz

Database → Organized collection of related data.

Primary key → Unique
↳ NOT NULL (Undefined value)

Candidate key →

~~Zero~~
Zero - Defined value

Super key →

Basic operation in DS

i) Traversing

ii) searching → Linear search $c(n) = n/2$ [better for small amount of data
Binary search $c(n) = \log_2 n$

iii) Inserting

iv) Deleting

v) Sorting

vi) Merging

Algorithm

~~Binary~~

BINARY (DATA, LB, UB, ITEM, LOC)

1. Set BEG = LB, END = UB and MID = INT((BIG + END)/2)
2. Repeat step 3 and 4 while BEG ≤ END
   and DATA[MID] ≠ ITEM
3. IF ITEM < DATA[MID]
   then Set END = MID - 1

   ELSE
   Set BEG = MID + 1
4. Set
   MID = INT ((BEG + EN)/2)
5. IF DATA[MID] = ITEM

   ~~Set LOC = MI~~
   Set LOC = MID

   Else
   Set LOC = NULL
6. Exit

Data

(11), 22, 30, 33, 40, 44, 55, 60, 66, 77, 80, 88, (99)

ITEM = 40

BEG = 1
END = 13
MID = (1+13)/2
= 14/2
= 7

(11), 22, 30, 33, 40, 44, (55), 60, 66, 77, 80, 88, (99)

40 < 55

BEG = 1
END = MID - 1 = 7 - 1 = 6
MID = (1+6)/2 = 7/2 = 3

(11), 22, (30), 33, 40, 44, (55) ~~60, 66~~

40 > 30

BEG = MID + 1 = 3 + 1 = 4
END = 6
MID = INT((4+6)/2) = INT(10/2) = 5

* No. of comparison in binary search
* Limitations in binary search.

03/01/2022
DS
3rd Class

Array

(i) Traversal
(ii) Search
(iii) Inserting
(iv) Deletion
(v) Sorting
(vi) Merging

Array
Record
Pointer

DS Distribution
- Linear
- Non-Linear

4:1 ~~Find~~

AAA(5:50)
BBB(-5:10)
CCC(18) // 18:1 = 18

(i) Find the number of elements in each array

Length = UB-LB+1

Length(AAA) = 50-5+1 = 46

Length (BBB) = 10+5+1 = 16

Length (CCC) = 18-1+1 = 18

(b) Suppose, AAA(5:50)

Base(AAA) = 300

W = 4 memory per cells for AAA

A (AAA[15]) = 300 + 4(15-5) = 300 + 40 = 340

A (AAA[35]) = 300 + 4(35-5) = 300 + 120 = 520

~~A(AAA[55]) = 300 + 4(55-5) = 300 +~~

A(AAA[55]) → No location as upper bound crossed.

Insertion

LA → Linear Array
N → The no. of element
k → position
ITEM → element to insert

INSERT(LA, N, K, ITEM)

1. SET J = N
2. Repeat step 3 and 4 while J ≥ K
3. Set LA[J+1] = LA[J]
4. Set J = J-1
5. Set LA[K] = ITEM
6. Set N = N+1
7. Exit

Deletion

DELETE (LA, N, K, ITEM)

1. Set ITEM = LA[K]
2. Repeat for J = K to N-1 set LA[J] = LA[J+1]
3. Set N = N-1
4. Exit

Bubble Sort ***

No. of comparison = No. of elements - 1

Ex-4.7

(A) 32, 51, 27, 85, 66, 23, 17, 57

$$F(n) = (n-1) + (n-2) + (n-3) + \dots + 1$$

$$= \frac{(n-1)(n-1+1)}{2}$$

$$= \frac{n(n-1)}{2}$$

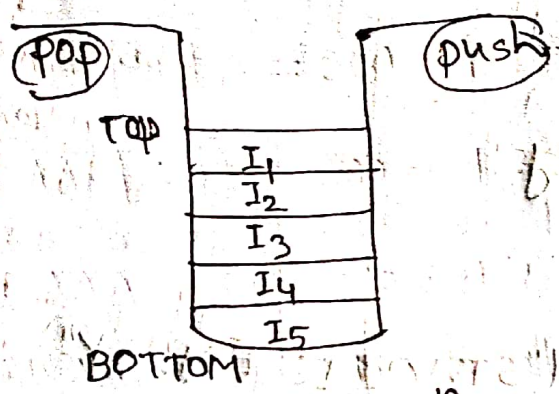
$$= \frac{n^2}{2} - \frac{n}{2}$$

$O(n^2)$

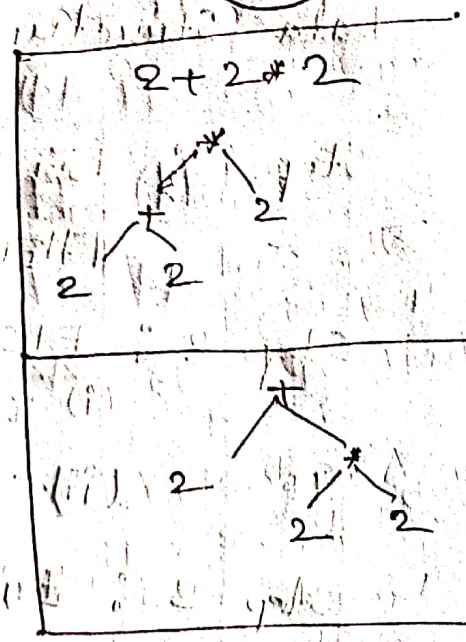
07/01/2023
DS
4th Class

Stacks

push, pop, status → present stat of stacks



Infix
Postfix
Prefix



Insertion

ALGORITHM PUSH_A(ITEM)

Input :

Output :

Data structure :

Step 1 : IF TOP ⩾ SIZE
THEN PRINT "STACK IS FULL"

Step 2 : ELSE
(i) TOP = TOP+1
(ii) A[TOP] = ITEM

Step 3 : ENDIF

Step 4 : Exit

Deletion

ALGORITHM POP(A)

Input : A stack with n elements

Output :

Data Structure : Array

Step 1 : IF TOP < 1
THEN PRINT ("STACK IS EMPTY")

Step 2 : ELSE
(i) ITEM = A[TOP]
(ii) TOP = TOP - 1

Step 3 : Exit

Example

Input

* ((A+B) * (C/D) - (E ^ (F * G)))

(A + B) ^ C - (D * E) / F

For exam
Arithmatic expression
Infix to postfix

Infix to postfix & prefix

Infix

((A + ((B ^ C) - D)) * (E - (A / C)))

Postfix

ABC^D-+ EAC/-*

* /
+ -
<, <=, >, >=
AND
OR, XOR

bridge player

10/01/2023
DS
5th class

INSERTION SORT

77, 33, 44, 11, 88, 22, 66, 55

Pass	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
K=1	77							
K=2	77	33						
	33	77						
K=3	33	77	44					
	33	44	77					
K=4	33	44	77	11				
	11	33	44	77				
K=5	11	33	44	77	88			
	11	33	44	77	88			
K=6	11	33	44	77	88	22		
	11	22	33	44	77	88		
K=7	11	22	33	44	77	88	66	
	11	22	33	44	66	77	88	
K=8	11	22	33	44	66	77	88	55
	11	22	33	44	55	66	77	88

Sorting

- Bubble Sort*
- Insertion Sort
- Selection Sort*
- Merge Sort
- Heap Sort
- Radix Sort
- Quick Sort

Insertion Sort Complexity

Worst Case

$O(n^2)$

$\frac{n(n+1)}{2}$

$1+2+3+\ldots+(n-1)$

$= \frac{n(n-1)}{2}$

$= \frac{n^2}{2} - \frac{n}{2}$

Average Case $O(n^2)$

$f(n) = \frac{1}{2} + \frac{2}{2} + \ldots + \frac{n-1}{2}$

$= \frac{n(n-1)}{2 \times 2}$

$= \frac{n^2}{4} - \frac{n}{4}$

Code

```
a = [5,8,6,1,7,9]
sizeof(a) = 6
for(i=0; i<6; i++)
{
    x = a[i];
    j = i-1;
while(j>=0 && a[j]>x)
{
    a[j+1] = a[j];
    j--;
}
a[j+1] = x
```

Selection Sort

77, 33, 44, 11, 88, 22, 66, 55

Pass	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
K=1	77	33	44	11	88	22	66	55
Loc=4	11	33	44	77	88	22	66	55
K=2	11	33	44	77	88	22	66	55
Loc=6	11	22	44	77	88	33	66	55
K=3	11	22	44	77	88	33	66	55
Loc=6	11	22	33	77	88	44	66	55
K=4	11	22	33	77	88	44	66	55
Loc=6	11	22	33	44	88	77	66	55
K=5	11	22	33	44	88	77	66	55
Loc=8	11	22	33	44	55	77	66	88
K=6	11	22	33	44	55	77	66	88
Loc=7	11	22	33	44	55	66	77	88
K=7	11	22	33	44	55	66	77	88
Loc=	11	22	33	44	55	66	77	88

Average Case and Worst Case Complexity

$$F(n) = (n-1)+(n-2)+\ldots +3+2+1$$

$$= 1+2+3+\ldots +n-1$$

$$= \frac{n(n-1)}{2}$$

$$= \frac{n^2}{2} - \frac{n}{2}$$

code

```
for(i=0; i<n; i++){
    for(j=i+1; j<n; j++)
```

14/01/2023
DS
6th class

String Pattern Matching Algorithm

* Bio-Informatics
* DNA sequencing

Formalize String Matching Problem

* A text → T[1...n]
* A pattern → P[1,m]
* m <= n
* $\Sigma = \{a, b, \ldots, z\}$, $\Sigma = \{0, 1\}$

NAIVE

```
NAIVE-STRING-MATCHER(T, P)

n = T.length
m = P.length
for s = 0 to n-m
    if P[1...m] == T[s+1 ... s+m]
        print "Pattern occurs with shift" s
```

Complexity

$O(m(n-m+1)) \sim O(n^2)$

KMP Algorithm

* Knuth – Morris – Pratt Algorithm
* Improves the worst case time complexity to $O(n)$

KMP Algorithm Preprocessing

* text = $T[1 \ldots n]$
* Pattern = $P[1 \ldots m]$
* LPS = $LPS[1 \ldots m]$

pattern[]

A	B	X	A	B

LPS[]

0	0	0	1	2

LPS[i] → Length of maximum matching prefix (suffix) of pattern

KMP

Text[]

A	B	X	A	B	A	B	X	A	B

pattern[]

A	B	X	A	B

LPS[]

0	0	0	1	2

17/01/2023
DS
7th class

Prefix function, π

```
m ← length[P]   // 'P' pattern to be matched
π[1] ← 0
k ← 0

for q ← 2 to m
    do while k>0 and P[k+1] != P[q]
        do k ← π[k]
    If P[k+1] = P[q]
        then k ← k+1
    π[q] ← k

Return π
```

0	1	2	3	4	5	6
a	b	a	b	a	c	a
0	0	1	2	3	0	1

21/01/2023
DS
8th class

(29)

Pattern

a	b	a	b	a	c	a
0	0	1	2	3	0	1

Text

b a c b a b a b a b a c a a b

Prefix function (P)

1. m ← length[P]
2. π[1] ← 0
3. k ← 0
4. For q ← 2 to m
5. do while) k > 0 and P[k+1] != P[q]
6. inner loop do k ← π[k]
7. If P[k+1] = P[q]
8. then k ← k+1
9. π[q] ← k
10. return π

Outer loop

P = ababaca

π → for storing value of proper prefix value

1-base indexing

	1	2	3	4	5	6	7
P →	a	b	a	b	a	c	a
	0	0	1	2	3	0	1

KMP MATCHER (T, P)

```
n = length[T]
m = length[P]
π ← Compute_Prefix_Function(P)
q ← 0
For i ← 1 to n
    do while q > 0 and P[q+1] ≠ T[i]
        do q ← π[q]
    If P[q+1] = T[i]
        then q ← q+1
    If q = m
        then print → pattern found
```

Merge Sort

22/01/2023
DS
Synopsis

1. 66, 33, 40, 22, 55, 88, 60, 11, 80, 20, 50, 44, 77, 30
2. 33, 66, 22, 40, 55, 88, 11, 60, 22, 20, 44, 50, 30, 77
3. 22, 33, 40, 66, 11, 55, 60, 88, 20, 44, 50, 80, 30, 77
4. 11, 22, 33, 40, 55, 60, 66, 88, 20, 30, 44, 50, 77, 80
5. 11, 20, 22, 30, 33, 40, 44, 50, 55, 60, 66, 77, 80, 88

Sorted list

Complexity NLogN

*** Application of stack → Quick Sort

Divide and Conquer →

- (44), 33, 11, 55, 70, 90, 40, 60, 99, 22, 88, 60

  44 < 60 ←
  44 < 88
  44 > 22

22, 33, 11, 55, 70, 90, 40, 60, 99, (44), 88, 60

→ 22 < 44
  33 < 44
  11 < 44
  55 > 44

22, 33, 11, (44), 70, 90, 40, 60, 99, 55, 88, 60

31/01/2023
DS
10th Class

Quick Sort

LOC

44, 33, 11, 55, 77, 90, 40, 60, 99, 22, 88, 55

BEG END

44 < 66
44 < 88
44 > 22

LOC

22, 33, 11, 55, 77, 90, 40, 60, 99, 44, 88, 66

22 < 44
33 < 44
11 < 44
55 > 44

22, 33, 11, 44, 77, 90, 40, 60, 90, 55, 88, 66

44 < 66
44 < 88
44 < 55
44 < 90
44 < 60
44 > 40

22, 33, 11, 40, 77, 90, 44, 60, 90, 55, 88, 66

22 < 44
33 < 44
11 < 44
40 < 44
77 ~~<~~ < 44

22, 33, 11, 40, 44, 90, 77, 60, 90, 55, 88, 66

44 at its desired position

6.11

LOC

(D)ATASTRUCTURES

BEG

END

D<S
D<E
D<R
D<U
D<T
D>C

Interchange = 1+1+1
= 3

Comparisons = 6+3+11+3
= 23

CATASTRUDTURES

C<D
A<D
T>D

CADASTRUTTURES

D<S
D<E
D<R
D<U
D<T
D<T
D<U
D<R
D<T
D<S

D7A

CAADSTRUTTURES

(A B C D E)

Radix Sort

348, 143, 361, 423, 538, 128, 321, 543, 366

0	1	2	3	4	5	6	7	8	9
	361 321		143 423 543			366		348 538 128	

0	1	2	3	4	5	6	7	8	9
		321 423 128	538	143 543 348		361 366			

0	1	2	3	4	5	6	7	8	9
	128 143		321 348 361 366	423	538 543				

128, 143, 321, 348, 361, 366, 423, 538, 543

11, 444, 220, 888, 7, 99, 177, 45

step-1

0	1	2	3	4	5	6	7	8	9
220	011			444	045		007 177	888	99

step-2

0	1	2	3	4	5	6	7	8	9
007	011	220		444 045			177	888	99

step-3

0	1	2	3	4	5	6	7	8	9
007 011 045 099	177	220		444				888	

7, 11, 45, 99, 177, 220, 444, 888

Complexity

2.6

Average Case

			no. of exchange
(1)	2	3	0
1	3	2	0
2	3	1	1
2	1	3	1
3	1	2	1
3	2	1	2

1 → largest
3 → smallest

$\frac{0+0+1+1+1+2}{6} = \frac{5}{6}$

$^3P_3 = 3$

$\frac{3!}{2!}$

$^3P_3 \times {}^2P_2 \times {}^1P_1$

$3 \times 2 \times 1$

$= 6$

Worst Case

$C(n) = n-1$

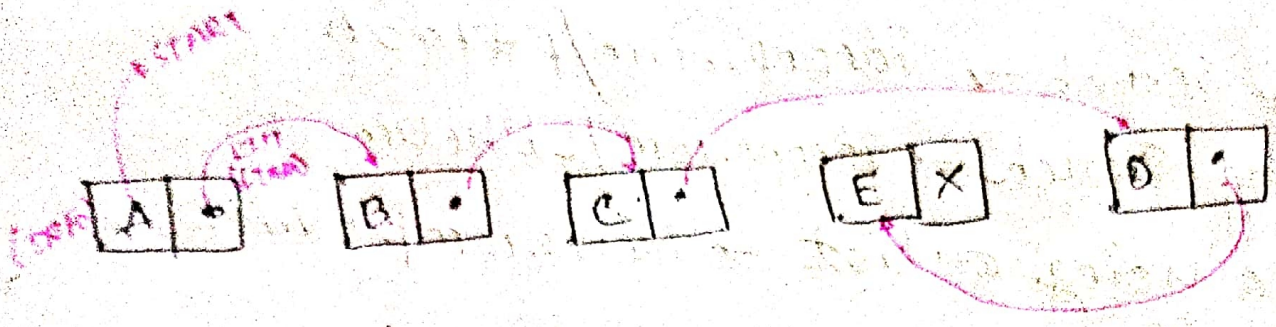
Best Case

$C(n) = 0$

07/02/2023
DS
12th Class

LINKED LIST

Insertion



* At the beginning
* At the given location

Asymptotic Notation

$f(n)$, $C(n)$

Ω
O
Θ

Mid

Sorting (Illustration; Example + Exercise)
(Algo-bubble)
Searching (Algorithm + Problem)
Pattern matching (Problem) (Naive vs KMP)
Complexity - (Problem)
Linked List (Algo + Advantages + Disadvantages)