UNIT–5 NOTES

SORTING AND SEARCHING

---

1. Introduction to Sorting

Sorting ka matlab data ko kisi order me arrange karna hota hai.

Usually:

- Ascending order
- Descending order

Example

Unsorted:

```
40 10 30 20
```

Sorted:

```
10 20 30 40
```

Advantages of Sorting

- Searching fast hoti hai
- Data organized hota hai
- Processing easy hoti hai

Types of Sorting Techniques

1. Bubble Sort
2. Selection Sort
3. Insertion Sort
4. Quick Sort
5. Merge Sort
6. Heap Sort
7. Shell Sort
8. Radix Sort

---

2. Bubble Sort

Bubble sort me adjacent elements compare hote hain aur swap kiye jate hain.

Largest element har pass me end tak pahunch jata hai.

Algorithm

```
1. Compare adjacent elements
2. Swap if wrong order
3. Repeat for all elements
4. Continue until sorted
```

Example

```
40 20 10 30
```

After sorting:

```
10 20 30 40
```

Advantages

- Simple implementation
- Easy understanding

Disadvantages

- Slow for large data

Time Complexity

$O(n^2)$

---

3. Selection Sort

Selection sort me smallest element select karke beginning me place karte hain.

Algorithm

```
1. Find minimum element
2. Swap with first position
3. Repeat for remaining array
```

Example

```
30 20 10
```

Sorted:

```
10 20 30
```

Advantages

- Less swapping

Disadvantages

- Slow for large arrays

Time Complexity

$O(n^2)$

---

4. Insertion Sort

Insertion sort cards arrangement jaisa hota hai.

Har element correct position par insert hota hai.

Algorithm

```
1. Take one element
2. Compare with previous
3. Insert at correct position
4. Repeat
```

Example

```
40 20 10
```

Sorted:

```
10 20 40
```

Advantages

- Efficient for small data
- Stable sorting

Disadvantages

- Slow for large data

Time Complexity

$O(n^2)$

---

5. Quick Sort

Quick sort divide and conquer technique use karta hai.

Ek pivot select hota hai aur elements divide hote hain.

Steps

```
1. Select pivot
2. Smaller elements left
3. Larger elements right
4. Recursively sort
```

Example

```
50 20 10 40
```

Pivot:

```
40
```

Sorted:

```
10 20 40 50
```

Advantages

- Very fast

- Efficient for large data

Disadvantages

- Worst case slow

Time Complexity

Best:

$O(n \log n)$

Worst:

$O(n^2)$

---

6. Merge Sort

Merge sort bhi divide and conquer use karta hai.

Array ko divide karke merge kiya jata hai.

Steps

```
1. Divide array
2. Sort subarrays
3. Merge sorted arrays
```

Example

```
40 20 10 30
```

Sorted:

```
10 20 30 40
```

Advantages

- Stable sorting
- Efficient

Disadvantages

- Extra memory required

Time Complexity

```
O(n log n)
```

---

7. Heap Sort

Heap sort heap tree use karta hai.

Max heap create karke sorting hoti hai.

Steps

```
1. Build max heap
2. Swap root with last
3. Heapify remaining elements
```

Advantages

- Efficient
- No extra memory

Disadvantages

- Complex implementation

Time Complexity

```
O(n log n)
```

---

8. Shell Sort

Shell sort insertion sort ka improved version hai.

Gap use hota hai.

Working

```
1. Choose gap
2. Compare distant elements
3. Reduce gap gradually
```

Advantages

- Faster than insertion sort

Disadvantages

- Complex gap selection

Time Complexity

Approx:

```
O(n log n)
```

---

9. Radix Sort

Radix sort digits ke basis par sorting karta hai.

Example

```
170 45 75 90
```

Digits ke according sort.

Advantages

- Fast for integers

Disadvantages

- Only specific data types

Time Complexity

`O(nk)`

Comparison of Sorting Techniques

Sorting	Time Complexity	Stable
Bubble	$O(n^2)$	Yes
Selection	$O(n^2)$	No
Insertion	$O(n^2)$	Yes
Quick	O(n log n)	No
Merge	O(n log n)	Yes
Heap	O(n log n)	No

---

10. Searching

Searching ka matlab data find karna.

Types of Searching

1. Sequential Search
2. Binary Search

Sequential Search

Elements ek-ek karke check hote hain.

Algorithm

```
1. Start from first element
2. Compare with target
3. Continue until found
```

Advantages

- Simple implementation

Disadvantages

- Slow for large data

Time Complexity

```
O(n)
```

Binary Search

Binary search sorted array par apply hoti hai.

Middle element compare hota hai.

Algorithm

```
1. Find middle element
2. Compare with target
3. Search left or right half
4. Repeat
```

Example

Array:

```
10 20 30 40 50
```

Search 30:

Middle = 30

Found.

Advantages

- Very fast

Disadvantages

- Requires sorted data

Time Complexity

```
O(log n)
```

Difference Between Sequential and Binary Search

Sequential	Binary
Linear checking	Middle checking
Slow	Fast
No sorting required	Sorting required
O(n)	O(log n)

Hashing

Hashing fast searching technique hai.

Hash function key ko index me convert karta hai.

Example

```
Index = key % size
```

Advantages

- Very fast searching

Disadvantages

- Collision problem

Indexing

Indexing data retrieval fast banata hai.

Used in:

- Databases
- Search engines

Applications of Data Structures

Operating System

- Queue → CPU scheduling
- Stack → function calls
- Tree → file system

DBMS

- B Tree indexing
- Hashing for searching

Applications of Sorting

- Database sorting
- Student records
- Search optimization