

Q1. Explain HashMap?

The screenshot shows a Paint application window with the following content:

Map Hierarchy Diagram:

```
graph TD
    MapI[Map(I)] --> HashMapC[HashMap(C)]
    HashMapC --> LinkedHashMapC[LinkedHashMap(C)]
```

HashMap Code Snippet:

```
HashMap hm=new HashMap();
hm.put("A",1);
hm.put("B",1);
hm.put("C",2);
hm.put("D",3);
hm.put("E",5);
```

Key-Value Table:

name	Mobile
A	1
B	1
C	2
D	3
E	5

Map Definition:

Map: If we want to represent group of elements as key value pair then we should go for Map
It is not a child Interface of Collection

Map Methods:

1. put(key,value)
2. get(key) `get("C");//2`
3. Set keySet() `hm.keySet();//A,B,C,D,E`
4. Collection values() `//hm.values();//1,1,2,3,5`
5. Map.Entry entrySet()

If we want to store data inform of key value pair where insertion order not preserved(on the basis of key) and duplicate keys are not allowed but values can be duplicated.

The diagram illustrates the Map(I) hierarchy and a corresponding data structure. On the left, a tree shows **Map(I)** as the parent interface, with **HashMap(C)** and **LinkedHashMap(C)** as its children. To the right, a table represents a Map with 'name' as the key and 'Mobile' as the value. The table contains five entries: (A, 1), (B, 1), (C, 2), (D, 3), and (E, 5). Below the table, the labels 'Key' and 'value' are underlined. To the right of the table, a list of Java code snippets is provided: `HashMap hm=new HashMap();`, `hm.put("A",1);`, `hm.put("B",1);`, `hm.put("C",2);`, `hm.put("D",3);`, and `hm.put("E",5);`. Below the table, a definition of Map is given: 'Map: If we want to represent group of elements as key value pair then we should go for Map. It is not a child Interface of Collection'. A list of methods follows: 1. `put(key,value)`, 2. `get(key)` with example `get("C");//2`, 3. `Set keySet()` with example `hm.keySet();//A,B,C,D,E`, 4. `Collection values()` with example `//hm.values();//1,1,2,3,5`, and 5. `Map.Entry entrySet()`. The background of the diagram is a screenshot of the Microsoft Paint application window.

Map(I)

HashMap(C)

LinkedHashMap(C)

`HashMap hm=new HashMap();`
`hm.put("A",1);`
`hm.put("B",1);`
`hm.put("C",2);`
`hm.put("D",3);`
`hm.put("E",5);`

name	Mobile
A	1
B	1
C	2
D	3
E	5

Key value

Map: If we want to represent group of elements as key value pair then we should go for Map
 It is not a child Interface of Collection

1. `put(key,value)`
2. `get(key)` `get("C");//2`
3. `Set keySet()` `hm.keySet();//A,B,C,D,E`
4. `Collection values()` `//hm.values();//1,1,2,3,5`
5. `Map.Entry entrySet()`

```
import java.util.HashMap;
import java.util.Map;
import java.util.Set;
```

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
/**
 *
 * @author Admin
 */
```

```
public class H1 {
    public static void main(String[] args) {
        HashMap<String,Integer> hm=new HashMap();
        hm.put("A",1);
        hm.put("B",1);
        hm.put("C",2);
        hm.put("D",3);
        hm.put("E",5);

        Set keys=hm.keySet();

        System.out.println("print element of HasMap using Map.Entry");
        for(Map.Entry<String,Integer> k:hm.entrySet()){
            System.out.println("Key is : "+k.getKey()+" Value is "+k.getValue());
        }
    }
}
```

```

    }
}

import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Set;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author Admin
 */
public class H1 {
    public static void main(String[] args) {
        HashMap<Integer,String> hm=new HashMap<>();
        hm.put(100,"A");
        hm.put(10,"B");
        hm.put(200,"C");
        hm.put(30,"D");
        hm.put(500,"E");

        System.out.println(""+hm);

        Set keys=hm.keySet();

        System.out.println("print element of HasMap using Map.Entry");
        for(Map.Entry<Integer,String> k:hm.entrySet()){
            System.out.println("Key is : "+k.getKey()+" Value is "+k.getValue());
        }
    }
}

```

Q2. Explain LinkedHashMap?

Ans: It is the Child Class of HashMap, where Duplicates keys are not allowed and Insertion Order are preserved then we should go for LinkedHashMap

```
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Set;

/*
 * To change this license header, choose License
Headers in Project Properties.
 * To change this template file, choose Tools |
Templates
 * and open the template in the editor.
 */

/**
 *
 * @author Admin
 */
public class H1 {
    public static void main(String[] args) {
        LinkedHashMap<Integer,String> hm=new
LinkedHashMap<>();
        hm.put(100,"A");
        hm.put(10,"B");
        hm.put(200,"C");
        hm.put(30,"D");
        hm.put(500,"E");
    }
}
```

```
System.out.println(""+hm);

Set keys=hm.keySet();

System.out.println("print element of HasMap
using Map.Entry");
for(Map.Entry<Integer,String> k:hm.entrySet()){
    System.out.println("Key is : "+k.getKey()+"
Value is "+k.getValue());
}

}

}
```

Q3. Explain TreeMap class in java?

Ans: If we want to represent group of elements in key value pair where duplicates key not allowed. Data stored in default natural sorting order on the basis of key

```
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Set;
import java.util.TreeMap;
```

```
/*
```

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

/**

*

* @author Admin

*/

public class H1 {

 public static void main(String[] args) {

 TreeMap<Integer,String> hm=new
TreeMap<>();

 hm.put(100,"A");

 hm.put(10,"B");

 hm.put(200,"C");

 hm.put(30,"D");

 hm.put(500,"E");

 System.out.println(""+hm);

 Set keys=hm.keySet();

 System.out.println("print element of HasMap
using Map.Entry");

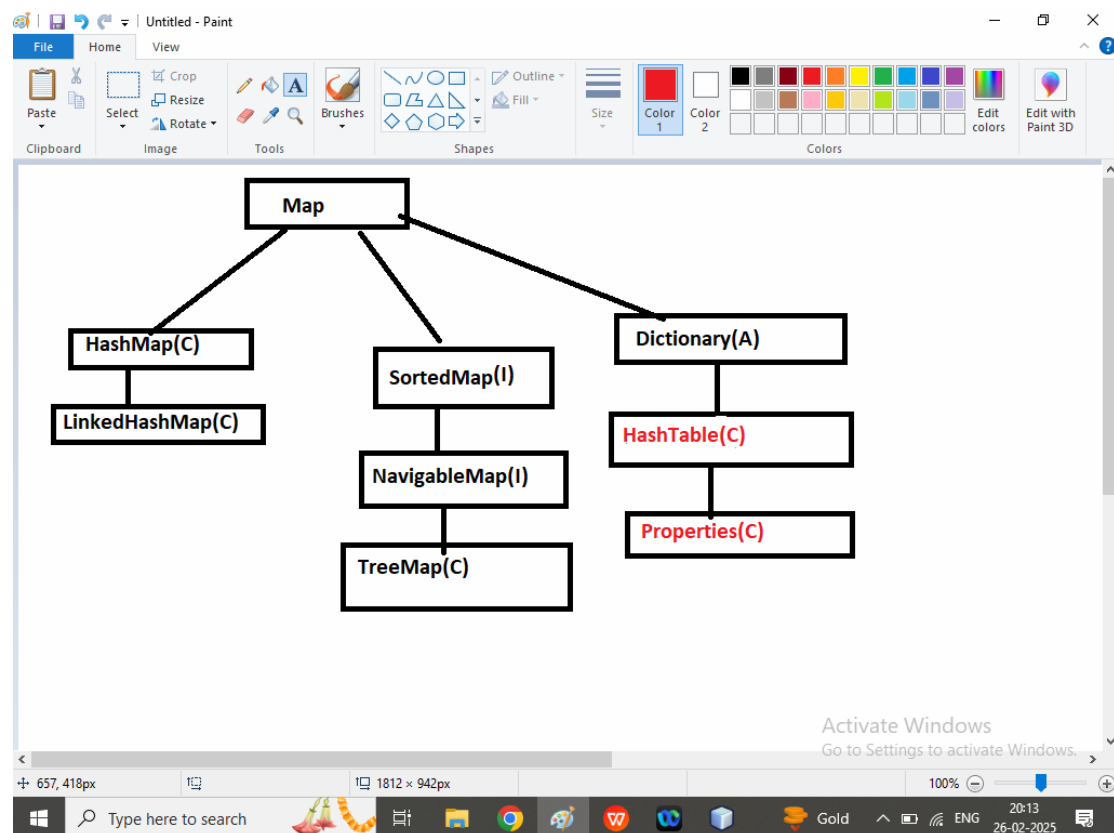
 for(Map.Entry<Integer,String> k:hm.entrySet()){

```

        System.out.println("Key is : "+k.getKey()+"
Value is "+k.getValue());
    }

}
}

```



Q4. Explain HashTable class in java?

Ans: if we want to represent group of entity in key value pair where duplicates keys are not allowed ,insertion order not preserved and thread safety is required then we should go for HashTable

```
import java.util.HashMap;
import java.util.Hashtable;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Set;
import java.util.TreeMap;
```

```
/*
 * To change this license header, choose License
 * Headers in Project Properties.
 * To change this template file, choose Tools |
 * Templates
 * and open the template in the editor.
 */
```

```
/**
 *
 * @author Admin
 */
public class H1 {
    public static void main(String[] args) {
        Hashtable<Integer,String> hm=new
        Hashtable<Integer, String>();
        hm.put(100,"A");
        hm.put(10,"B");
        hm.put(200,"C");
        hm.put(30,"D");
    }
}
```



```
hm.put(500,"E");
hm.put(100, "AAAA");

System.out.println(""+hm);

Set keys=hm.keySet();

System.out.println("print element of HasMap
using Map.Entry");
for(Map.Entry<Integer,String> k:hm.entrySet()){
    System.out.println("Key is : "+k.getKey()+"
Value is "+k.getValue());
}

}

}
```

```
import java.util.HashMap;
import java.util.Hashtable;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.Set;
import java.util.TreeMap;
```

```
/*
 * To change this license header, choose License
 * Headers in Project Properties.
```

* To change this template file, choose Tools |
Templates

* and open the template in the editor.

*/

/**

*

* @author Admin

*/

public class H1 {

 public static void main(String[] args) {

 Hashtable<Integer,String> hm=new
Hashtable<Integer, String>();

 hm.put(100,"A");

 hm.put(10,"B");

 hm.put(200,"C");

 hm.put(30,"D");

 hm.put(500,"E");

 hm.put(100, "AAAA");

 System.out.println(""+hm);

 Set keys=hm.keySet();

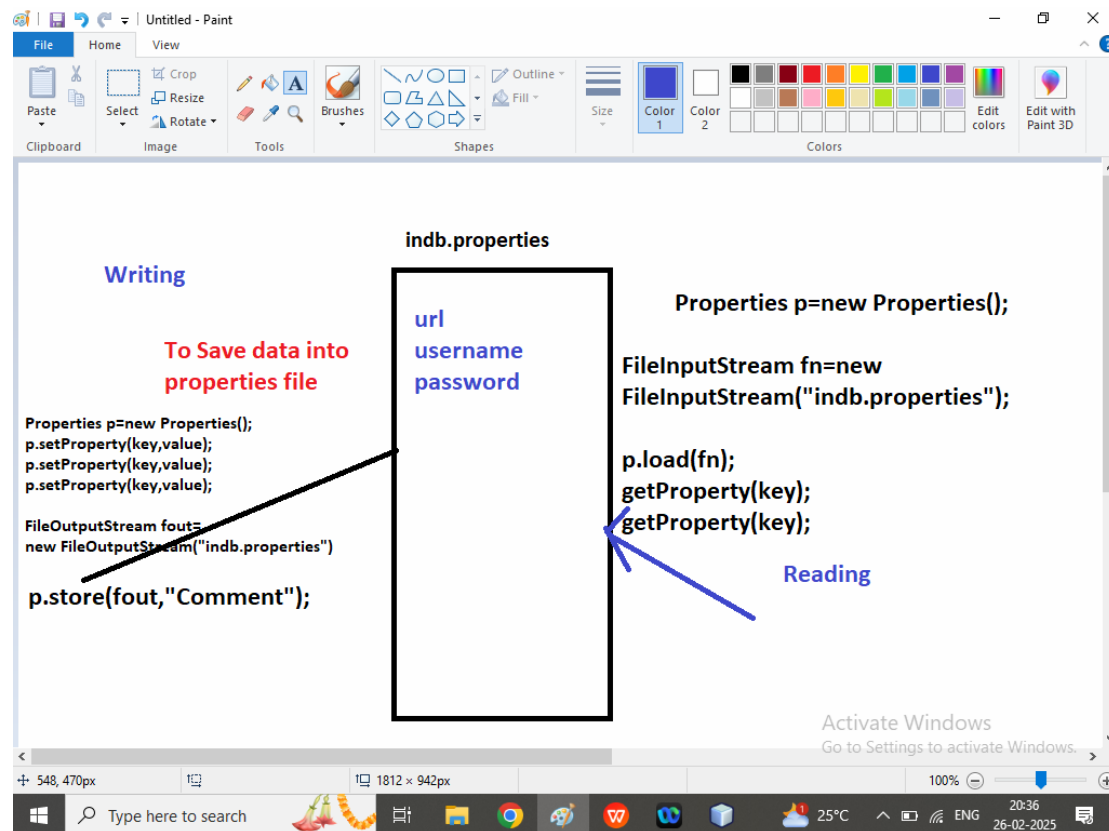
 System.out.println("Contains Key :
"+hm.containsKey(30));

 System.out.println("value check
"+hm.containsValue("D"));

```
        System.out.println("print element of HasMap  
using Map.Entry");  
        for(Map.Entry<Integer,String> k:hm.entrySet()){  
            System.out.println("Key is : "+k.getKey()+"  
Value is "+k.getValue());  
        }  
  
    }  
}
```

Q3.Explain Properties class in java?

Ans: if we want to represent data in form of key value pair where key and value must String type then we should go for Properties class
Properties class/ properties file is used to set configuration of particular application



Example : Write data into indb.properties file

```
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Properties;
```

```
/*
```

```
 * To change this license header, choose License
  Headers in Project Properties.
```

```
 * To change this template file, choose Tools |
  Templates
```

```
 * and open the template in the editor.
```

```
*/
```

```
/**
 *
 * @author Admin
 */
public class P1 {
    public static void main(String[] args) throws
FileNotFoundException, IOException {

        FileOutputStream fout=new
FileOutputStream("indb.properties");
        Properties p=new Properties();
        p.setProperty("url",
"jdbc:mysql://localhost:3306/db");
        p.setProperty("un", "root");
        p.setProperty("ps", "Ram@1234");
        p.store(fout,"This File is Created By Me");
        System.out.println("Data Written success");

    }
}
```

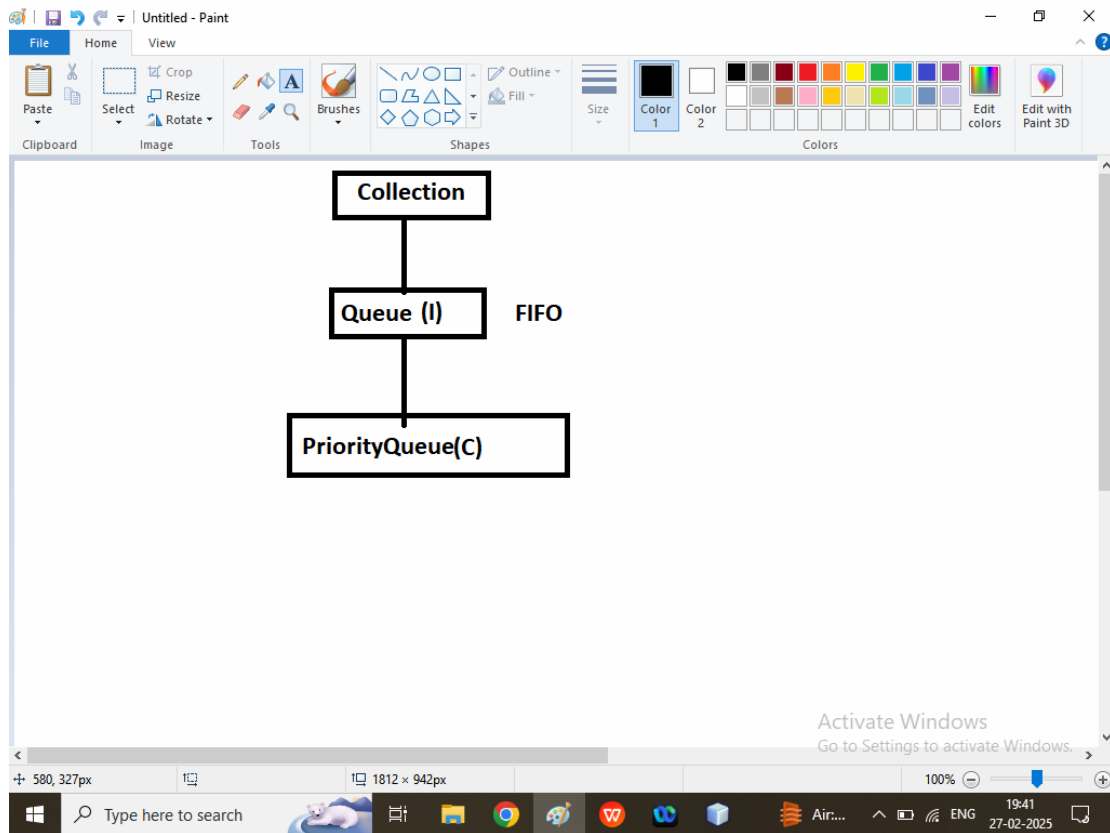
```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Properties;
```

```
/*
 * To change this license header, choose License
Headers in Project Properties.
 * To change this template file, choose Tools |
Templates
 * and open the template in the editor.
 */
```

```
/**
 *
 * @author Admin
 */
public class P1 {
    public static void main(String[] args) throws
FileNotFoundException, IOException {

        FileInputStream fin=new
FileInputStream("indb.properties");
        Properties p=new Properties();
        p.load(fin);
        System.out.println("UserName :
"+p.getProperty("un"));
        System.out.println("Password :
"+p.getProperty("ps"));
        System.out.println("URL :
"+p.getProperty("url"));
    }
}
```

```
}  
}
```



```
import java.util.PriorityQueue;
```

```
public class P2 {  
    public static void main(String[] args) {  
  
        PriorityQueue<Integer> pq=new PriorityQueue<Integer>();  
  
        pq.add(1);  
        pq.add(2);  
        pq.add(3);  
  
        System.out.println(""+pq);  
        System.out.println("Delete Element : "+pq.remove());  
        System.out.println(""+pq);  
    }  
}
```

```
import java.util.ArrayDeque;
import java.util.PriorityQueue;
```

```
public class P2 {
    public static void main(String[] args) {

        PriorityQueue<Integer> pq=new PriorityQueue<Integer>();

        pq.add(1);
        pq.add(2);
        pq.add(3);

        System.out.println(""+pq);
        System.out.println("Delete Element : "+pq.remove());
        System.out.println(""+pq);
        ArrayDeque<Integer> ad=new ArrayDeque<>();
        ad.add(10);
        ad.add(20);
        ad.add(30);

        System.out.println(""+ad);
        ad.addFirst(5);//insert data using front end
        ad.addLast(40);//insert data using rear end
        System.out.println(""+ad);
    }
}
```


Untitled - Paint

File Home View

Paste Select Crop Resize Rotate Image Tools Brushes Shapes Outline Fill Size Color 1 Color 2 Colors Edit colors Edit with Paint 3D

Graph Representation

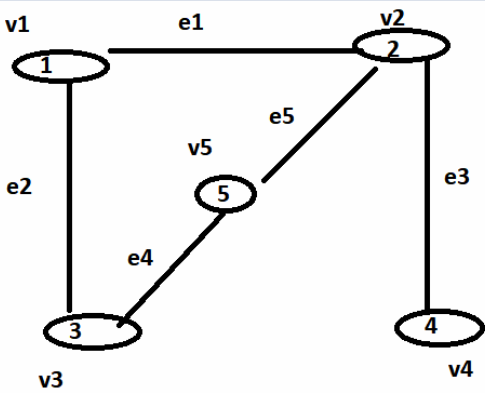
A. Matrix

B. AdjacencyList

Graph Traversal

1. BFS(Queue)

2. DFS(Stack)



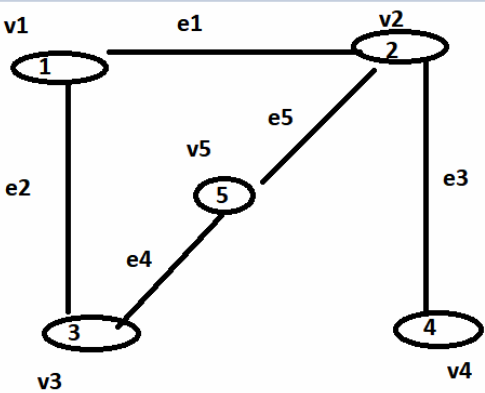
Activate Windows
Go to Settings to activate Windows.

1812 x 942px 100% 19:48 27-02-2025

Untitled - Paint

File Home View

Paste Select Crop Resize Rotate Image Tools Brushes Shapes Outline Fill Size Color 1 Color 2 Colors Edit colors Edit with Paint 3D

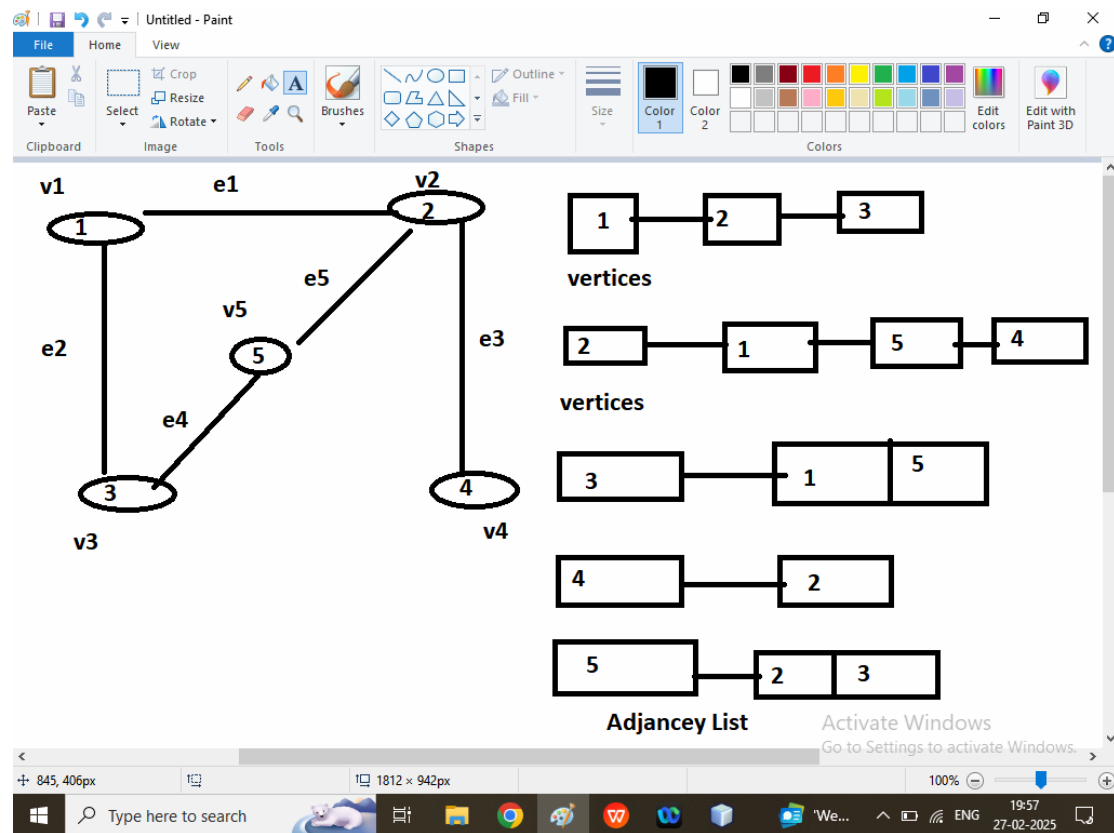


	1	2	3	4	5
1	0	1	1	0	0
2	1	0	0	1	1
3	1	0	0	0	1
4	0	1	0	0	0
5	0	1	1	0	0

Matrix

Activate Windows
Go to Settings to activate Windows.

779, 465px 1812 x 942px 100% 19:53 27-02-2025



Algorithm of BFS:(Queue)

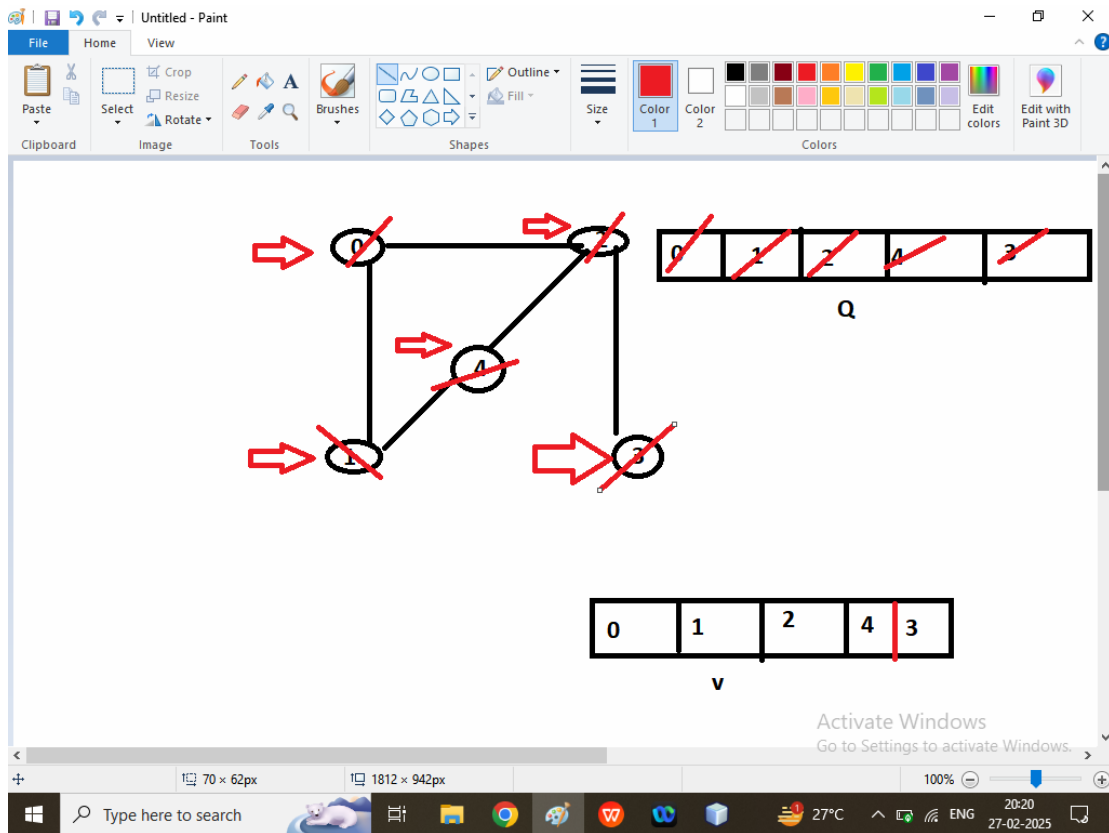
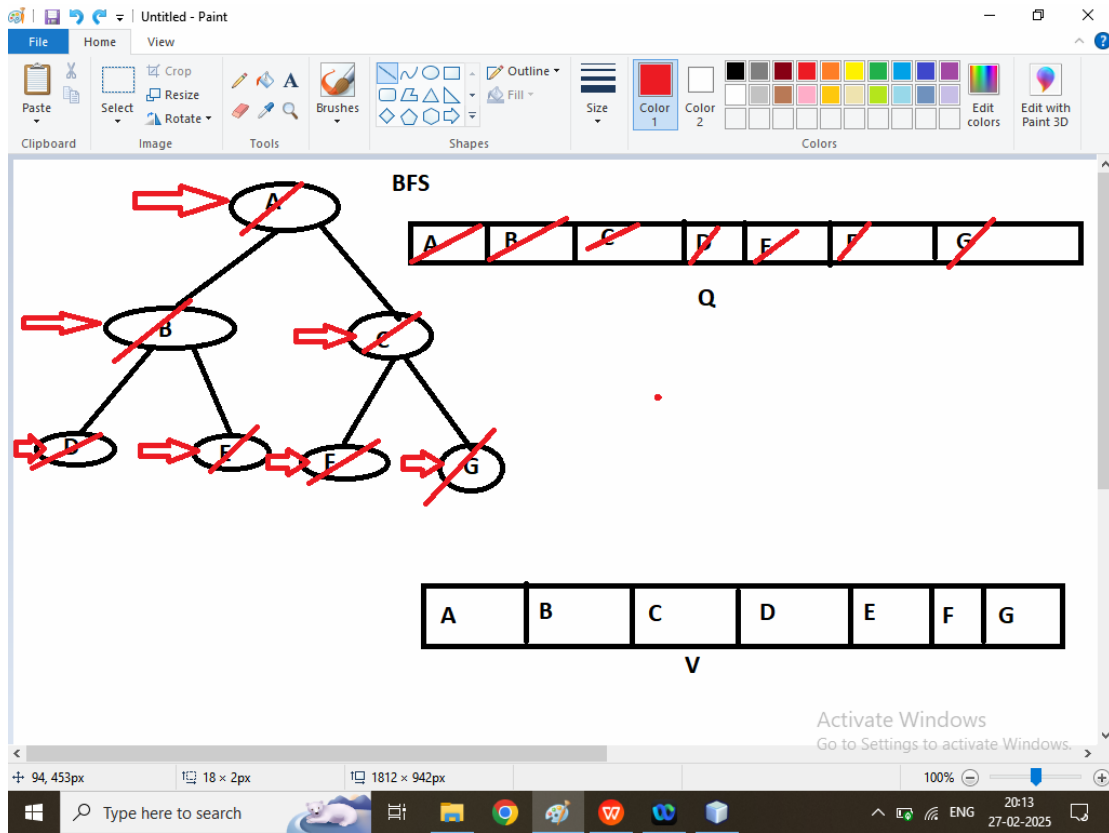
Step1:
Declare Queue as Q variable and Declare Visited as v

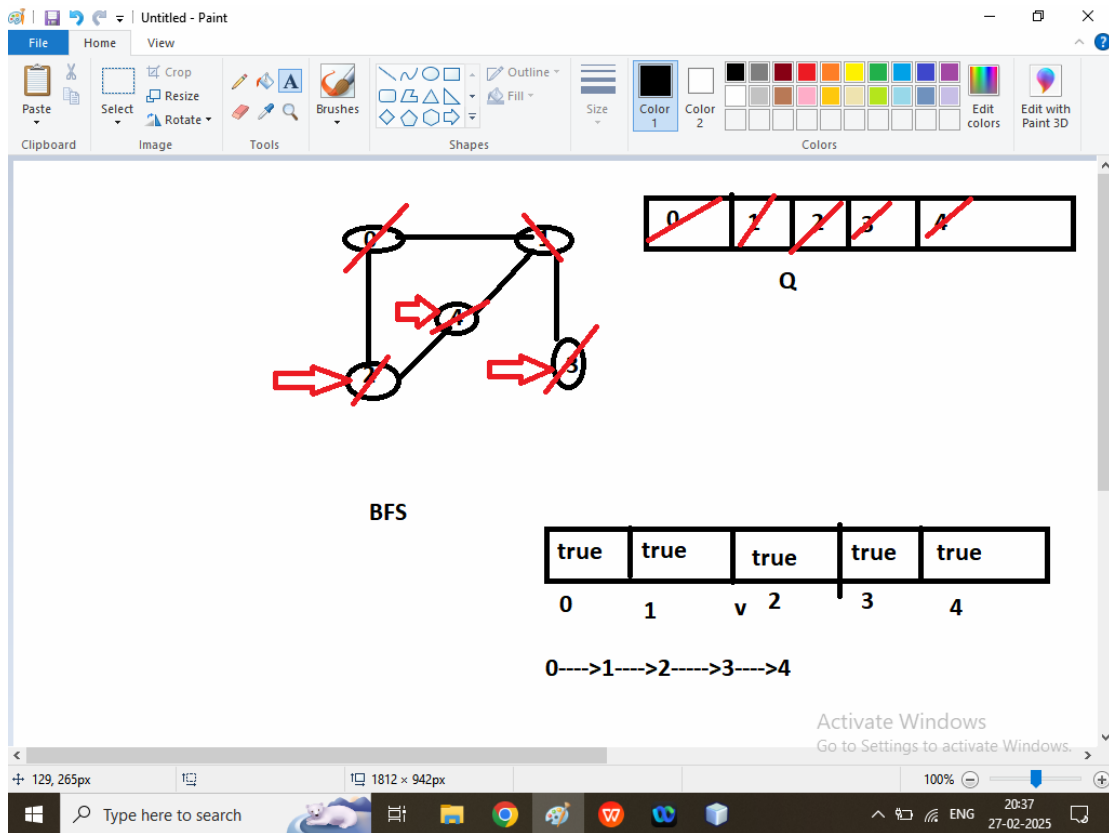
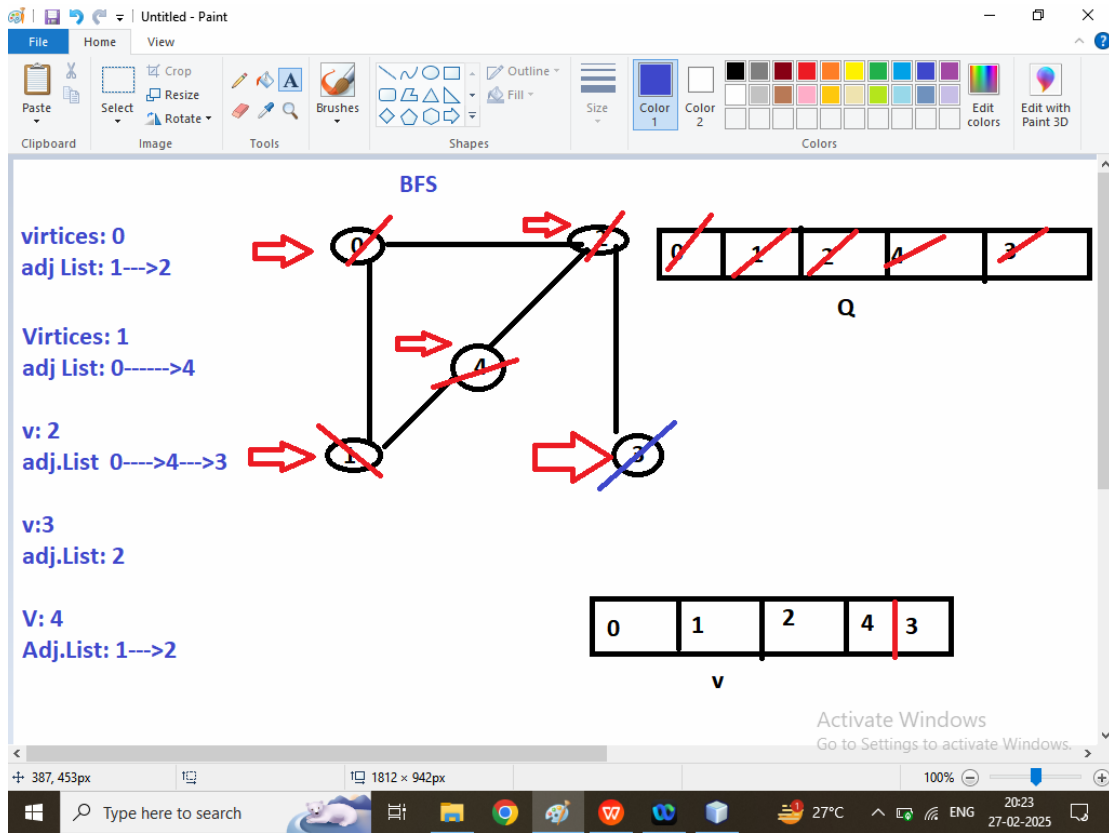
Step2: Enqueue current element in the Q
add current element into visited array v

Step3: Loop until Q is empty

- check if current node in visited array if not then first add into visited array
- Dequeue the current element from the Q and add(enqueue) its unvisited adjacent into the Q

step4: If queue is the empty then stop the algorithm





```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
/**
 *
 * @author Admin
 */
```

```
import java.util.LinkedList;
import java.util.Queue;
```

```
// Class to represent a graph using adjacency list
class Graph {
```

```
    int vertices;//5
    LinkedList<Integer>[] adjList;

    Graph(int vertices) {//v=5
        this.vertices = vertices;
        adjList = new LinkedList[vertices];
        for (int i = 0; i < vertices; ++i) {
            adjList[i] = new LinkedList<>();
        }
    }
```

```
// Function to add an edge to the graph
void addEdge(int u, int v) {
    adjList[u].add(v);
}
```

```
// Function to perform Breadth First Search on a graph
// represented using adjacency list
void bfs(int startNode) {
    // Create a queue for BFS
    Queue<Integer> queue = new LinkedList<>();
    boolean[] visited = new boolean[vertices];
```

```
    // Mark the current node as visited and enqueue it
    visited[startNode] = true;
    queue.add(startNode);
```

```
    // Iterate over the queue
    while (!queue.isEmpty()) {
        // Dequeue a vertex from queue and print it
        int currentNode = queue.poll();
        System.out.print(currentNode + " ");//0
```

```
        // Get all adjacent vertices of the dequeued
        // vertex currentNode If an adjacent has not
        // been visited, then mark it visited and
        // enqueue it
        for (int neighbor : adjList[currentNode]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
```

```
        queue.add(neighbor);
    }
}
}
```

```
class Main {

    public static void main(String[] args) {
        // Number of vertices in the graph
        int vertices = 5;

        // Create a graph
        Graph graph = new Graph(vertices);

        // Add edges to the graph
        graph.addEdge(0, 1);
        graph.addEdge(0, 2);
        graph.addEdge(1, 3);
        graph.addEdge(1, 4);
        graph.addEdge(2, 4);

        // Perform BFS traversal starting from vertex 0
        System.out.print(
            "Breadth First Traversal starting from vertex 0: ");
        graph.bfs(0);
    }
}
```