OUTPUT

```
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
1
Enter a Element to Push : 4
4 Pushed into the stack
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
1
Enter a Element to Push : 6
6 Pushed into the stack
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
1
Enter a Element to Push : 1
1 Pushed into the stack
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
3
```

```
Stack elements :
1 -> 6 -> 4 ->
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
2
1 Popped from the stack

 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
2
6 Popped from the stack

 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
3
Stack elements :
4 ->
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
```

```
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
2
4 Popped from the stack

 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
2
Stack Underflow !!
 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
3
Stack is empty

 Choose an operation :
 1.Push
 2.Pop
 3.dispaly
 4.search
```

OUTPUT

```
1.Insert at begning
2.Insert at end
3.Insert at any position
4.Search
5.Display
6.Delete from begining
7.Delete from end
8.Delete from any position
9.Exit


Enter your choice  1

Enter the data to be insered  45
45

1.Insert at begning
2.Insert at end
3.Insert at any position
4.Search
5.Display
6.Delete from begining
7.Delete from end
8.Delete from any position
9.Exit


Enter your choice  2
Enter the data to be insered  6
45
6

1.Insert at begning
2.Insert at end
3.Insert at any position
```

```
 Enter your choice  3
Enter the data to be insered  2
Enter the position to be insered  12
45
6

 1.Insert at begning
 2.Insert at end
 3.Insert at any position
 4.Search
 5.Display
 6.Delete from begining
 7.Delete from end
 8.Delete from any position
 9.Exit


 Enter your choice  4
Enter the value to be search  2
element not found

 1.Insert at begning
 2.Insert at end
 3.Insert at any position
 4.Search
 5.Display
 6.Delete from begining
 7.Delete from end
 8.Delete from any position
 9.Exit


 Enter your choice  6
6
```

OUTPUT

```
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
1

Enter the element to be inserted: 2
2
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
2

Enter the element to be inserted: 5
2   5
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
3

Enter the element to be inserted: 2
```

```
Enter the position of element to be inserted:
1
2   2   5
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
4
2 is deleted
2   5
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
5
5 is deleted
2
1.Insert at begining
2.Insert at end
3.Insert at position
4.Delete at begining
5.Delete at end
6.Delete at position
7.display
8.exit
7
2
```

OUTPUT

```
1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 1
Enter a Number :28
Root node created!

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 2
Enter a Number :15
Value Inserted!

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 2
Enter a Number :23
Value Inserted!
```

```
1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 2
Enter a Number :26
Value Inserted!

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 2
Enter a Number :30
Value Inserted!

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 2
Enter a Number :20
Value Inserted!
```

```
1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 4
15 20 23 26 28 30

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 5
28 15 23 20 26 30

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 6
20 26 23 15 30 28
```

```
1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 7
Enter Value to be Deleted :30
Item deleted!

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
Select an Option : 4
15 20 23 26 28

1.Create root Node
2.Insert Node
3.Search Node
4.Inorder Traversal
5.preorder Traversal
6.Postorder Traversal
7.Delete Node
8.Exit
```

OUTPUT

```
Enter the size of the circular queue: 3
1. Enqueue
2. Dequeue
3. Display
4. Search
5. Exit

Enter your choice: 1

Enter the element to push: 4

The queue is:
4
1. Enqueue
2. Dequeue
3. Display
4. Search
5. Exit

Enter your choice: 1

Enter the element to push: 5

The queue is:
4 5
1. Enqueue
2. Dequeue
3. Display
4. Search
5. Exit

Enter your choice: 2

Popped element 4 from queue

The queue is:
```

```
The queue is:
5
1. Enqueue
2. Dequeue
3. Display
4. Search
5. Exit

Enter your choice: 4

Enter the element to search: 5
Element 5 found
1. Enqueue
2. Dequeue
3. Display
4. Search
5. Exit
```

OUTPUT

```
enter Universal Set Size : 10
enter 10 elements for the Universal Set:
element 1: 1
element 2: 2
element 3: 3
element 4: 4
element 5: 5
element 6: 6
element 7: 7
element 8: 8
element 9: 9
element 10: 10
enter set a size (max 10): 5
enter 5 elements (must be in the Universal Set):
Element 1: 2
Element 2: 3
Element 3: 4
Element 4: 5
Element 5: 6
enter Set B Size (max 10): 4
enter 4 elements (must be in the Universal Set):
Element 1: 3
Element 2: 4
Element 3: 5
Element 4: 6
set a bit string: {0, 1, 1, 1, 1, 1, 0, 0, 0, 0}
set b bit string: {0, 0, 1, 1, 1, 1, 0, 0, 0, 0}

union: {0, 1, 1, 1, 1, 1, 0, 0, 0, 0}
union result (values): {2, 3, 4, 5, 6}

intersection: {0, 0, 1, 1, 1, 1, 0, 0, 0, 0}
intersection result (values): {3, 4, 5, 6}

difference (a - b): {0, 1, 0, 0, 0, 0, 0, 0, 0, 0}
difference result (a - b, values): {2}

difference (B - A): {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
difference result (B - A, values): {}
```

OUTPUT

```
Enter the size of the set : 5
ess and hold Esc
Enter 5 unique elements for the set:
Enter element 1: 1
Enter element 2: 2
Enter element 3: 3
Enter element 4: 4
Enter element 5: 5

1. set representatives
2. Union operation
3. Find operation
4. Display sets
5. Exit
Enter your choice: 2

Enter first element: 1
Enter second element: 5

1. set representatives
2. Union operation
3. Find operation
4. Display sets
5. Exit
Enter your choice: 2

Enter first element: 2
Enter second element: 3

1. set representatives
2. Union operation
3. Find operation
4. Display sets
5. Exit
Enter your choice: 4

Disjoint Sets:
{ 1, 5 }
{ 2, 3 }
{ 4 }

1. set representatives
2. Union operation
```

OUTPUT

```
Enter the number of vertices: 6
Enter the number of edges: 7
Enter edge 1 (s-d) 0 1
Enter edge 2 (s-d) 0 2
Enter edge 3 (s-d) 1 4
Enter edge 4 (s-d) 1 3
Enter edge 5 (s-d) 2 4
Enter edge 6 (s-d) 3 5
Enter edge 7 (s-d) 4 5

Menu:
1. Display Adjacency Matrix
2. DFS Traversal
3. BFS Traversal
4. Topological Sort
5. Exit
Enter your choice: 1

Graph Representation (Adjacency Matrix):
0 1 1 0 0 0
0 0 0 1 1 0
0 0 0 0 1 0
0 0 0 0 0 1
0 0 0 0 0 1
0 0 0 0 0 0

Menu:
1. Display Adjacency Matrix
2. DFS Traversal
3. BFS Traversal
4. Topological Sort
5. Exit
Enter your choice: 2

Enter the start vertex for DFS: 1
DFS starting from vertex 1: 1 3 5 4 0 2

Menu:
1. Display Adjacency Matrix
2. DFS Traversal
3. BFS Traversal
4. Topological Sort
5. Exit
```

```
Enter your choice: 3

Enter the start vertex for BFS: 0
BFS starting from vertex 0: 0 2 1 4 3 5

Menu:
1. Display Adjacency Matrix
2. DFS Traversal
3. BFS Traversal
4. Topological Sort
5. Exit
Enter your choice: 4
Topological Sort: 0 1 3 2 4 5

Menu:
1. Display Adjacency Matrix
2. DFS Traversal
3. BFS Traversal
4. Topological Sort
5. Exit
Enter your choice:
```

OUTPUT

```
Enter the number of vertices: 8
Enter the number of edges: 9
Enter the edges s and d:
0 1
3 0
1 2
2 3
2 4
4 5
5 6
6 7
6 4

Strongly Connected Components:
SCC 1: 0 3 2 1
SCC 2: 4 6 5
SCC 3: 7
 3 strongly connected components are there.
```

OUTPUT

```
Edge 1: (0 1) cost: 4
Edge 2: (0 7) cost: 8
Edge 3: (7 6) cost: 1
Edge 4: (6 5) cost: 2
Edge 5: (5 2) cost: 4
Edge 6: (2 8) cost: 2
Edge 7: (2 3) cost: 7
Edge 8: (3 4) cost: 9

Minimum cost: 37
```

OUTPUT

```
The edges of the Minimum Cost Spanning Tree are:
1 edge (6,7) = 1
2 edge (2,8) = 2
3 edge (5,6) = 2
4 edge (0,1) = 4
5 edge (2,5) = 4
7 edge (2,3) = 7
9 edge (0,7) = 8
11 edge (3,4) = 9

Minimum cost = 37
```