7.3 APPROXIMATION ALGORITHM FOR VERTEX COVER

Question:
Implement an approximation algorithm for the Vertex Cover problem. Compare the performance of the approximation algorithm with the exact solution obtained through brute-force. Consider the following graph G=(V,E) where V={1,2,3,4,5} and E={(1,2),(1,3),(2,3),(3,4),(4,5)}.

Input:
• Graph G = (V, E) with V = {1, 2, 3, 4, 5}, E = {(1,2), (1,3), (2,3), (3,4), (4,5)}

Output:
• Approximation Vertex Cover: {2, 3, 4}
• Exact Vertex Cover (Brute-Force): {2, 4}
• Performance Comparison: Approximation solution is within a factor of 1.5 of the optimal solution.

AIM
To implement both an approximation algorithm and an exact brute-force algorithm for the Vertex Cover problem and compare their performance.

ALGORITHM
1. Approximation Algorithm (2-Approximation):
2. Initialize cover = ∅.
3. While E is not empty:
    - Select an arbitrary edge (u, v) ∈ E.
    - Add both u and v to cover.
    - Remove all edges incident on u or v.
4. Return cover (guaranteed to be within factor 2 of optimal).
5. Brute-Force Algorithm:
6. Enumerate all subsets of vertices.
7. For each subset, check if every edge has at least one endpoint in the subset.
8. Track the minimum-sized valid subset as the exact vertex cover.
9. Comparison: Compute ratio |Approximation| / |Optimal| as performance factor.

PROGRAM

```
from itertools import combinations

def vertex_cover_approx(edges):
    cover = set()
    used = set()
    for u, v in edges:
        if u not in used and v not in used:
            cover.update([u, v])
            used.update([u, v])
    return cover

def vertex_cover_exact(n, edges):
    for r in range(1, n+1):
        for combo in combinations(range(1, n+1), r):
            if all(u in combo or v in combo for u, v in edges):
                return set(combo)
    return set()

n = int(input("Enter number of vertices: "))
m = int(input("Enter number of edges: "))
edges = []
for _ in range(m):
    u, v = map(int, input("Edge: ").split())
    edges.append((u, v))

approx = vertex_cover_approx(edges)
exact = vertex_cover_exact(n, edges)
print("Approximation Vertex Cover:", approx)
print("Exact Vertex Cover:", exact)
print("Performance Comparison: Approximation size =", len(approx), ", Exact size =", len(exact))
```

Input:

Graph G = (V, E) with V = {1, 2, 3, 4, 5}, E = {(1,2), (1,3), (2,3), (3,4), (4,5)}

Output:

```
Enter number of vertices: 5
Enter number of edges: 5
Edge: 1 2
Edge: 1 3
Edge: 2 3
Edge: 3 4
Edge: 4 5
Approximation Vertex Cover: {1, 2, 3, 4}
Exact Vertex Cover: {1, 2, 4}
Performance Comparison: Approximation size = 4 , Exact size = 3
>>>
```

RESULT:

The program is executed successfully and the output is verified.

PERFORMANCE ANALYSIS:

Time Complexity: Runs in O(|E|), efficient for large graphs.

Space Complexity: Runs in O(2^|V|), feasible only for small graphs.Performance