```python
def max_regions_colored(n, edges, k):
    graph = [[] for _ in range(n)]
    for u, v in edges:
        graph[u].append(v)
        graph[v].append(u)

    def color_graph(v, colored):
        if v == n: return len([c for c in colored if c == 1])
        max_count = 0
        for color in range(k):
            if all(colored[nb] != color for nb in graph[v]):
                colored[v] = color
                max_count = max(max_count, color_graph(v + 1, colored))
                colored[v] = -1
        return max_count

    return color_graph(0, [-1] * n)

# Example Usage
print(max_regions_colored(4, [(0, 1), (1, 2), (2, 3), (3, 0), (0, 2)], 3))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.264 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
2

[Done] exited with code=0 in 0.095 seconds
```

```python
def hamiltonian_cycle(n, edges):
    graph = [[] for _ in range(n)]
    for u, v in edges:
        graph[u].append(v)
        graph[v].append(u)

    def dfs(v, visited, count):
        if count == n and v == start: return True
        visited[v] = True
        for nb in graph[v]:
            if not visited[nb] or (nb == start and count == n):
                if dfs(nb, visited, count + 1): return True
        visited[v] = False
        return False

    for start in range(n):
        if dfs(start, [False] * n, 1): return True
    return False

# Example Usage
print(hamiltonian_cycle(5, [(0, 1), (1, 2), (2, 3), (3, 0), (0, 2), (2, 4), (4, 0)]))
print(hamiltonian_cycle(4, [(0, 1), (1, 2), (2, 3), (3, 0), (0, 2)]))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.095 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
False
True

[Done] exited with code=0 in 0.078 seconds
```

```python
def subsets(S):
    S.sort()
    res = [[]]
    for num in S:
        res += [item + [num] for item in res]
    return res

# Example Usage
print(subsets([1, 2, 3]))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.078 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
[[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1, 2, 3]]

[Done] exited with code=0 in 0.08 seconds
```

```python
def subsets_with_element(S, x):
    S.sort()
    res = [[]]
    for num in S:
        res += [item + [num] for item in res if x in (item + [num])]
    return [subset for subset in res if x in subset]

# Example Usage
print(subsets_with_element([2, 3, 4, 5], 3))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.08 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
[[3], [3, 4], [3, 5], [3, 4, 5]]

[Done] exited with code=0 in 0.075 seconds
```

```python
def power_set(nums):
    res = [[]]
    for num in nums: res += [item + [num] for item in res]
    return res

# Example Usage
print(power_set([1, 2, 3]))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.075 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
[[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1, 2, 3]]

[Done] exited with code=0 in 0.078 seconds
```

```python
from collections import Counter

def universal_strings(words1, words2):
    max_b = Counter()
    for b in words2: max_b |= Counter(b)
    return [a for a in words1 if not max_b - Counter(a)]

# Example Usage
print(universal_strings(["amazon","apple","facebook","google","leetcode"], ["e","o"]))
print(universal_strings(["amazon","apple","facebook","google","leetcode"], ["l","e"]))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
['facebook', 'google', 'leetcode']
['apple', 'google', 'leetcode']

[Done] exited with code=0 in 0.127 seconds
```

```python
from collections import Counter

def universal_strings(words1, words2):
    # Create a Counter for the letters in all words in words2
    max_b = Counter()
    for b in words2:
        max_b |= Counter(b)  # Update max_b with the maximum count of letters

    # Filter words1 based on whether they contain all required letters
    return [a for a in words1 if not max_b - Counter(a)]

# Example Usage
words1 = ["amazon", "apple", "facebook", "google", "leetcode"]
words2 = ["e", "o"]
print(universal_strings(words1, words2))  # Output: ["facebook", "google", "leetcode"]

words2 = ["l", "e"]
print(universal_strings(words1, words2))  # Output: ["apple", "google", "leetcode"]

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[Done] exited with code=0 in 0.078 seconds

[Running] python -u "c:\Users\hp\OneDrive\Desktop\project directory\tempCodeRunnerFile.python"
['facebook', 'google', 'leetcode']
['apple', 'google', 'leetcode']

[Done] exited with code=0 in 0.127 seconds
```