Started on	Thursday, 15 May 2025, 9:54 AM
State	Finished
Completed on	Thursday, 15 May 2025, 11:36 AM
Time taken	1 hour 42 mins
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a python program to find minimum steps to reach to specific cell in minimum moves by knight.

Answer: (penalty regime: 0 %)

Reset answer

```
    queue.append(cell(knightpos[0], knightpos[1], 0))
    visited = [[False for i in range(N + 1)] for j in range(N + 1)]
    visited[knightpos[0]][knightpos[1]] = True
    while(len(queue) > 0):
        t = queue[0]
        queue.pop(0)
        if(t.x == targetpos[0] and
           t.y == targetpos[1]):
            return t.dist
        for i in range(8):
            x = t.x + dx[i]
            y = t.y + dy[i]
            if(isInside(x, y, N) and not visited[x][y]):
                visited[x][y] = True
                queue.append(cell(x, y, t.dist + 1))

if __name__=='__main__':
    N = 30
    knightpos = [1, 1]
    targetpos = [30, 30]
    print(minStepToReachTarget(knightpos,
                               targetpos, N))
```

	Input	Expected	Got	
✓	30	20	20	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Write a python program to implement pattern matching on the given string using Brute Force algorithm.

For example:

Test	Input	Result
BF(a1,a2)	abcaaaabbbbcccabcbabdbcsbbbbbnnn ccabcba	12

Answer: (penalty regime: 0 %)

Reset answer

```python
def BF(s1,s2):
    n = len(s1)
    m = len(s2)

    for i in range(n - m + 1):
        j = 0
        while j < m and s1[i + j] == s2[j]:
            j += 1
        if j == m:
            return i
    return -1
if __name__ == "__main__":
    a1=input()
    a2=input()
    b=BF(a1,a2)
    print(b)

```

	Test	Input	Expected	Got	
✓	BF(a1,a2)	abcaaaabbbbcccabcbabdbcsbbbbbnnn ccabcba	12	12	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a python program to implement KMP (Knuth Morris Pratt).

For example:

Input	Result
ABABDABACDABABCABAB ABABCABAB	Found pattern at index 10

Answer: (penalty regime: 0 %)

Reset answer

```
def KMPSearch(pat, txt):
    M = len(pat)
    N = len(txt)

    lps = [0] * M
    computeLPSArray(pat, M, lps)

    i = 0
    j = 0
    while i < N:
        if pat[j] == txt[i]:
            i += 1
            j += 1

        if j == M:
            print("Found pattern at index", i - j)
            j = lps[j - 1]

        elif i < N and pat[j] != txt[i]:
            if j != 0:
                j = lps[j - 1]
            else:
```

	Input	Expected	Got	
✓	ABABDABACDABABCABAB ABABCABAB	Found pattern at index 10	Found pattern at index 10	✓
✓	SAVEETHAENGINEERING VEETHA	Found pattern at index 2	Found pattern at index 2	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Not answered

Mark 0.00 out of 20.00

Write a python program to implement merge sort without using recursive function on the given list of float values.

For example:

Input	Result

5
6.2
4.1
3.2
5.6
7.4
	`left:  [6.2]`
	`Right:  [4.1]`
	`left:  [3.2]`
	`Right:  [5.6]`
	`left:  [7.4]`
	`Right:  []`
	`left:  [4.1, 6.2]`
	`Right:  [3.2, 5.6]`
	`left:  [7.4]`
	`Right:  []`
	`left:  [3.2, 4.1, 5.6, 6.2]`
	`Right:  [7.4]`
	`[3.2, 4.1, 5.6, 6.2, 7.4]`

6
3.2
8.9
4.5
6.2
1.5
8.0
	`left:  [3.2]`
	`Right:  [8.9]`
	`left:  [4.5]`
	`Right:  [6.2]`
	`left:  [1.5]`
	`Right:  [8.0]`
	`left:  [3.2, 8.9]`
	`Right:  [4.5, 6.2]`
	`left:  [1.5, 8.0]`
	`Right:  []`
	`left:  [3.2, 4.5, 6.2, 8.9]`
	`Right:  [1.5, 8.0]`
	`[1.5, 3.2, 4.5, 6.2, 8.0, 8.9]`

Answer: (penalty regime: 0 %)

Question **5**

Correct

Mark 20.00 out of 20.00

Write a python program to check whether Hamiltonian path exits in the given graph.

For example:

Test	Result
`Hamiltonian_path(adj, N)`	YES

Answer: (penalty regime: 0 %)

Reset answer

```
def Hamiltonian_path(adj, N):
    dp = [[False for i in range(1 << N)] for j in range(N)]
    for i in range(N):
         dp[i][1 << i]=True
    for i in range(1 << N):
        for j in range(N):
            if ((i & (1 << j))!=0):
                for k in range(N):
                    if((i & (1 << k)) != 0 and
                            adj[k][j] == 1 and
                                    j != k and
                          dp[k][i ^ (1 << j)]):
                       dp[j][i]=True
                       break
    for i in range(N):
        if (dp[i][(1 << N)-1]):
            return True
    return False
adj = [ [ 0, 1, 1, 1, 0 ] ,
        [ 1, 0, 1, 0, 1 ],
        [ 1, 1, 0, 1, 1 ],
        [ 1, 0, 1, 0, 0 ] ]
```

	Test	Expected	Got	
✓	`Hamiltonian_path(adj, N)`	YES	YES	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.