Started on	Saturday, 26 April 2025, 8:20 AM
State	Finished
Completed on	Saturday, 26 April 2025, 9:02 AM
Time taken	41 mins 16 secs
Grade	**80.00** out of 100.00

Question **1**

Not answered

Mark 0.00 out of 20.00

Write a python program to convert the given decimal number to binary number using recursive function.

For example:

Input	Result
10	1010
15	1111

Answer: (penalty regime: 0 %)

```
1
```

	Input	Expected	Got	
✖	10	1010	86400	✖

Some hidden test cases failed, too.

Your code must pass all tests to earn any marks. Try again.

Show differences

Incorrect

Marks for this submission: 0.00/20.00.

Question **2**

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):
    ############## Add your code here #################
    M = len(pat)
    N = len(txt)
    lps = [0]*M
    j = 0
    computeLPSArray(pat, M, lps)
    i = 0
    while (N - i) >= (M - j):
        if pat[j] == txt[i]:
            i += 1
            j += 1
        if j == M:
            print ("Found pattern at index " + str(i-j))
            j = lps[j-1]
        elif i < N and pat[j] != txt[i]:
            if j != 0:
                j = lps[j-1]
            else:
                i += 1

def computeLPSArray(pat, M, lps):
```

	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 **3**

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

```
class cell:

    def __init__(self, x = 0, y = 0, dist = 0):
        self.x = x
        self.y = y
        self.dist = dist

def isInside(x, y, N):
    if (x >= 1 and x <= N and
        y >= 1 and y <= N):
        return True
    return False
def minStepToReachTarget(knightpos,
                         targetpos, N):
    # add your code here
    dx = [-2, -1, 1, 2, -2, -1, 1, 2]
    dy = [-1, -2, -2, -1, 1, 2, 2, 1]
    queue = []
    queue.append(cell(knightpos[0], knightpos[1], 0))
    visited = [[False for _ in range(N + 1)] for _ in range(N + 1)]
    visited[knightpos[0]][knightpos[1]] = True
    while len(queue) > 0:
```

	Input	Expected	Got	
✔	30	20	20	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

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

```
def BF(s1,s2):
    ##############  Add your code here #############
    i = 0
    j = 0
    while(i < len(s1) and j < len(s2)):
        if(s1[i] ==  s2[j]):
            i += 1
            j += 1
        else:
            i = i - j + 1
            j = 0
    if(j >= len(s2)):
        return i - len(s2)
    else:
        return 0

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 **5**

Correct

Mark 20.00 out of 20.00

Create a python program to implement Hamiltonian circuit problem using Backtracking.

For example:

Result
Solution Exists: Following is one Hamiltonian Cycle 0 1 2 4 3 0

Answer: (penalty regime: 0 %)

Reset answer

```
class Graph():
    def __init__(self, vertices):
        self.graph = [[0 for column in range(vertices)]
                          for row in range(vertices)]
        self.V = vertices
    def isSafe(self, v, pos, path):
        if self.graph[ path[pos-1] ][v] == 0:
            return False
        for vertex in path:
            if vertex == v:
                return False

        return True
    def hamCycleUtil(self, path, pos):
        ######################Add your code here#################################
        if pos == self.V:
            if self.graph[ path[pos-1] ][ path[0] ] == 1:
                return True
            else:
                return False
        for v in range(1,self.V):
            if self.isSafe(v, pos, path) == True:
```

	Expected	Got	
✔	Solution Exists: Following is one Hamiltonian Cycle 0 1 2 4 3 0	Solution Exists: Following is one Hamiltonian Cycle 0 1 2 4 3 0	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.