Started on	Thursday, 15 May 2025, 1:43 PM
State	Finished
Completed on	Thursday, 15 May 2025, 6:17 PM
Time taken	4 hours 33 mins
Overdue	2 hours 33 mins
Grade	**80.00** out of 100.00

Question **1**

Not answered

Mark 0.00 out of 20.00

Write a python program to implement the quick sort using recursion on the given list of float values.

For example:

Input	Result
5 6.3 1.2 4.6 5.8 9.7	pivot: 9.7 pivot: 5.8 pivot: 4.6 [1.2, 4.6, 5.8, 6.3, 9.7]
6 2.3 7.8 9.5 4.2 3.6 5.4	pivot: 5.4 pivot: 3.6 pivot: 7.8 [2.3, 3.6, 4.2, 5.4, 7.8, 9.5]

Answer: (penalty regime: 0 %)

```
1
```

Question **2**

Correct

Mark 20.00 out of 20.00

Create a python program to find the Hamiltonian path using Depth First Search for traversing the graph .

For example:

Test	Result
`hamiltonian.findCycle()`	`['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'A']` `['A', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']`

Answer: (penalty regime: 0 %)

Reset answer

```
class Hamiltonian:
    def __init__(self, start):
        self.start = start
        self.cycle = []
        self.hasCycle = False

    def findCycle(self):
        self.cycle.append(self.start)
        self.solve(self.start)

    def solve(self, vertex):
        ############  Add your code here ##############
        if vertex==self.start and len(self.cycle)==N+1:
            self.hasCycle=True
            self.displayCycle()
        for i in range(len(vertices)):
            if adjacencyM[vertex][i]==1 and visited[i]==0:
                nbr=i
                self.cycle.append(nbr)
                visited[nbr]=1
                self.solve(nbr)
                visited[nbr]=0
```

	Test	Expected	Got	
✔	`hamiltonian.findCycle()`	`['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'A']` `['A', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']`	`['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'A']` `['A', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']`	✔

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 Boyer Moore Algorithm with Good Suffix heuristic to find pattern in given text string.

For example:

Input	Result
ABAAABAACD ABA	pattern occurs at shift = 0 pattern occurs at shift = 4

Answer: (penalty regime: 0 %)

Reset answer

```
def preprocess_strong_suffix(shift, bpos, pat, m):
    ###############  Add your Code here ######################
    i = m
    j = m + 1
    bpos[i] = j
    while i > 0:
        while j <= m and pat[i - 1] != pat[j - 1]:
            if shift[j] == 0:
                shift[j] = j - i
            j = bpos[j]
        i -= 1
        j -= 1
        bpos[i] = j

def preprocess_case2(shift, bpos, pat, m):
    j = bpos[0]
    for i in range(m + 1):
        if shift[i] == 0:
            shift[i] = j
        if i == j:
            j = bpos[j]
def search(text, pat):
```

	Input	Expected	Got	
✔	ABAAABAACD ABA	pattern occurs at shift = 0 pattern occurs at shift = 4	pattern occurs at shift = 0 pattern occurs at shift = 4	✔
✔	SaveethaEngineering Saveetha veetha	pattern occurs at shift = 2 pattern occurs at shift = 22	pattern occurs at shift = 2 pattern occurs at shift = 22	✔

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 knight tour problem using backtracking

For example:

Input	Result
5	Found a solution 01 20 11 14 03 10 15 02 19 12 21 24 13 04 07 16 09 06 23 18 25 22 17 08 05

Answer: (penalty regime: 0 %)

Reset answer

```
BOARD_SIZE = int(input())
board = [[0 for i in range(BOARD_SIZE)] for j in range(BOARD_SIZE)]
STEPS = [[-1, 2], [1, 2], [-2, 1], [2, 1], [1, -2], [-1, -2], [2, -1], [-2, -1]]


def solve_knights_tour(x, y, step_count):
    ######################### Add your code here ############################3
    if step_count==BOARD_SIZE**2+1:
        return True
    for step in STEPS:
        x_new=x+step[0]
        y_new=y+step[1]
        if is_safe(x_new,y_new):
            board[x_new][y_new]=step_count
            if solve_knights_tour(x_new, y_new, step_count+1):
                return True
            board[x_new][y_new]=0
    return False

def is_safe(x, y):
    return 0 <= x < BOARD_SIZE and 0 <= y < BOARD_SIZE and board[x][y] == 0

```

	Input	Expected	Got	
✔	5	Found a solution 01 20 11 14 03 10 15 02 19 12 21 24 13 04 07 16 09 06 23 18 25 22 17 08 05	Found a solution 01 20 11 14 03 10 15 02 19 12 21 24 13 04 07 16 09 06 23 18 25 22 17 08 05	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

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 ##############
    m=len(s1)
    n=len(s2)
    for i in range(m-n+1):
        j=0
        while j<n and s1[i+j]==s2[j]:
            j+=1
        if j==n:
            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.