Started on	Thursday, 15 May 2025, 1:38 PM
State	Finished
Completed on	Thursday, 15 May 2025, 9:24 PM
Time taken	7 hours 46 mins
Overdue	5 hours 46 mins
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a python program to implement knight tour problem

For example:

Input	Result
5 5	[1, 12, 25, 18, 3] [22, 17, 2, 13, 24] [11, 8, 23, 4, 19] [16, 21, 6, 9, 14] [7, 10, 15, 20, 5] [(0, 0), (1, 2), (0, 4), (2, 3), (4, 4), (3, 2), (4, 0), (2, 1), (3, 3), (4, 1), (2, 0), (0, 1), (1, 3), (3, 4), (4, 2), (3, 0), (1, 1), (0, 3), (2, 4), (4, 3), (3, 1), (1, 0), (2, 2), (1, 4), (0, 2)] Done!

Answer: (penalty regime: 0 %)

Reset answer

```python
import sys
class KnightsTour:
    def __init__(self, width, height):
        self.w = width
        self.h = height
        self.board = []
        self.generate_board()

    def generate_board(self):
        for i in range(self.h):
            self.board.append([0]*self.w)

    def print_board(self):

        for elem in self.board:
            print (elem)

    def generate_legal_moves(self, cur_pos):
        possible_pos = []
        move_offsets = [(1, 2), (1, -2), (-1, 2), (-1, -2),
                        (2, 1), (2, -1), (-2, 1), (-2, -1)]
        ############### Add your code here ##############################
```

	Input	Expected	Got	
✓	5 5	[1, 12, 25, 18, 3] [22, 17, 2, 13, 24] [11, 8, 23, 4, 19] [16, 21, 6, 9, 14] [7, 10, 15, 20, 5] [(0, 0), (1, 2), (0, 4), (2, 3), (4, 4), (3, 2), (4, 0), (2, 1), (3, 3), (4, 1), (2, 0), (0, 1), (1, 3), (3, 4), (4, 2), (3, 0), (1, 1), (0, 3), (2, 4), (4, 3), (3, 1), (1, 0), (2, 2), (1, 4), (0, 2)] Done!	[1, 12, 25, 18, 3] [22, 17, 2, 13, 24] [11, 8, 23, 4, 19] [16, 21, 6, 9, 14] [7, 10, 15, 20, 5] [(0, 0), (1, 2), (0, 4), (2, 3), (4, 4), (3, 2), (4, 0), (2, 1), (3, 3), (4, 1), (2, 0), (0, 1), (1, 3), (3, 4), (4, 2), (3, 0), (1, 1), (0, 3), (2, 4), (4, 3), (3, 1), (1, 0), (2, 2), (1, 4), (0, 2)] Done!	✓

	Input	Expected	Got	
✓	6 6	[1, 32, 9, 18, 3, 34] [10, 19, 2, 33, 26, 17] [31, 8, 25, 16, 35, 4] [20, 11, 36, 27, 24, 15] [7, 30, 13, 22, 5, 28] [12, 21, 6, 29, 14, 23] [(0, 0), (1, 2), (0, 4), (2, 5), (4, 4), (5, 2), (4, 0), (2, 1), (0, 2), (1, 0), (3, 1), (5, 0), (4, 2), (5, 4), (3, 5), (2, 3), (1, 5), (0, 3), (1, 1), (3, 0), (5, 1), (4, 3), (5, 5), (3, 4), (2, 2), (1, 4), (3, 3), (4, 5), (5, 3), (4, 1), (2, 0), (0, 1), (1, 3), (0, 5), (2, 4), (3, 2)] Done!	[1, 32, 9, 18, 3, 34] [10, 19, 2, 33, 26, 17] [31, 8, 25, 16, 35, 4] [20, 11, 36, 27, 24, 15] [7, 30, 13, 22, 5, 28] [12, 21, 6, 29, 14, 23] [(0, 0), (1, 2), (0, 4), (2, 5), (4, 4), (5, 2), (4, 0), (2, 1), (0, 2), (1, 0), (3, 1), (5, 0), (4, 2), (5, 4), (3, 5), (2, 3), (1, 5), (0, 3), (1, 1), (3, 0), (5, 1), (4, 3), (5, 5), (3, 4), (2, 2), (1, 4), (3, 3), (4, 5), (5, 3), (4, 1), (2, 0), (0, 1), (1, 3), (0, 5), (2, 4), (3, 2)] Done!	✓

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

```
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.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a python program to implement quick sort using tha last element as pivot on the list of float values.

For example:

Test	Input	Result
`quickSort(arr,0,n-1)`	5 3.2 1.5 9.6 4.1 5.9	Sorted array is: 1.5 3.2 4.1 5.9 9.6

Answer: (penalty regime: 0 %)

```python
def quickSort(nums,l,r):
    if len(nums)==1:
        return nums
    if l<r:
        pi=part(nums,l,r)
        quickSort(nums,l,pi-1)
        quickSort(nums,pi+1,r)
def part(nums,l,r):
    pi,ptr=nums[r],l
    for i in range(l,r):
        if nums[i]<=pi:
            nums[i],nums[ptr]=nums[ptr],nums[i]
            ptr+=1
    nums[ptr],nums[r]=nums[r],nums[ptr]
    return ptr
n=int(input())
arr=[float(input()) for i in range(n)]
quickSort(arr,0,n-1)
print("Sorted array is:")
for i in range(n):
    print(arr[i])
```

	Test	Input	Expected	Got	
✓	`quickSort(arr,0,n-1)`	5 3.2 1.5 9.6 4.1 5.9	Sorted array is: 1.5 3.2 4.1 5.9 9.6	Sorted array is: 1.5 3.2 4.1 5.9 9.6	✓
✓	`quickSort(arr,0,n-1)`	6 2.3 50.4 9.8 7.6 3.4 1.5	Sorted array is: 1.5 2.3 3.4 7.6 9.8 50.4	Sorted array is: 1.5 2.3 3.4 7.6 9.8 50.4	✓
✓	`quickSort(arr,0,n-1)`	8 2.3 1.5 6.4 9.8 7.6 4.2 3.8 1.4	Sorted array is: 1.4 1.5 2.3 3.8 4.2 6.4 7.6 9.8	Sorted array is: 1.4 1.5 2.3 3.8 4.2 6.4 7.6 9.8	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **4**

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 **5**

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

```python
def KMPSearch(pat, txt):
    ############### Add your code here ##################
    lp=len(pat)
    ls=len(txt)
    lps=[0]*lp
    computeLPSArray(pat,lp,lps)
    i=0
    j=0

    while(i!=ls):
        if txt[i]==pat[j]:
            i+=1
            j+=1
        else:
            j=lps[j-1]
        if j==lp:
            print("Found pattern at index",i-j)
            j=lps[j-1]
        elif j==0:
            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.