Started on	Wednesday, 30 July 2025, 3:19 PM
State	Finished
Completed on	Wednesday, 30 July 2025, 3:58 PM
Time taken	38 mins 42 secs
Grade	**100.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Write a Program for Implementing merge sort using python recursion.

For example:

Test	Input	Result
merge_sort(inp_arr)	8 11 31 7 41 101 56 77 2	Input Array: [11, 31, 7, 41, 101, 56, 77, 2] Sorted Array: [2, 7, 11, 31, 41, 56, 77, 101]
merge_sort(inp_arr)	5 61 2 41 80 9	Input Array: [61, 2, 41, 80, 9] Sorted Array: [2, 9, 41, 61, 80]

Answer: (penalty regime: 0 %)

```
def merge_sort(inp_arr):
    n = len(inp_arr)
    current_size = 1

    while current_size < n:
        left = 0
        while left < n - 1:
            mid = min(left + current_size - 1, n - 1)
            right = min(left + 2 * current_size - 1, n - 1)

            merge(inp_arr, left, mid, right)
            left += 2 * current_size

```

```
14          current_size *= 2
15
16
17  def merge(inp_arr, left, mid, right):
18      n1 = mid - left + 1
19      n2 = right - mid
20
21      L = [inp_arr[left + i] for i in range(n1)]
22      R = [inp_arr[mid + 1 + i] for i in range(n2)]
```

	Test	Input	Expected	Got	
✔	merge_sort(inp_arr)	8 11 31 7 41 101 56 77 2	Input Array: [11, 31, 7, 41, 101, 56, 77, 2] Sorted Array: [2, 7, 11, 31, 41, 56, 77, 101]	Input Array: [11, 31, 7, 41, 101, 56, 77, 2] Sorted Array: [2, 7, 11, 31, 41, 56, 77, 101]	✔
✔	merge_sort(inp_arr)	5 61 2 41 80 9	Input Array: [61, 2, 41, 80, 9] Sorted Array: [2, 9, 41, 61, 80]	Input Array: [61, 2, 41, 80, 9] Sorted Array: [2, 9, 41, 61, 80]	✔
✔	merge_sort(inp_arr)	6 100 30 29 5 600 21	Input Array: [100, 30, 29, 5, 600, 21] Sorted Array: [5, 21, 29, 30, 100, 600]	Input Array: [100, 30, 29, 5, 600, 21] Sorted Array: [5, 21, 29, 30, 100, 600]	✔

	Test	Input	Expected	Got	
✔	`merge_sort(inp_arr)`	4 21 10 30 5	Input Array: [21, 10, 30, 5] Sorted Array: [5, 10, 21, 30]	Input Array: [21, 10, 30, 5] Sorted Array: [5, 10, 21, 30]	✔

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 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):
    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):
    len = 0
```

	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 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):
    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):
    s = 0
```

	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):
    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 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 is_valid(v,pos,path,adj,N):
    if adj[path[pos-1]][v]==0:
        return False
    if v in path:
        return False
    return True
def hamUtil(adj,path,pos,N):
    if pos==N:
        return True
    for v in range(N):
        if is_valid(v,pos,path,adj,N):
            path[pos]=v
            if hamUtil(adj,path,pos+1,N):
                return True
            path[pos]=-1
    return True
def Hamiltonian_path(adj,N):
    path=[-1]*N
    path[0]=0

    if hamUtil(adj,path,1,N) == False:
        print ("Solution does not exist\n")
```

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

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.