

Started on	Monday, 19 May 2025, 8:24 AM
State	Finished
Completed on	Monday, 19 May 2025, 11:03 AM
Time taken	2 hours 39 mins
Overdue	39 mins 9 secs
Grade	80.00 out of 100.00

Question **1**

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

```

1 class Hamiltonian:
2     def __init__(self, start):
3         self.start = start
4         self.hasCycle = False
5         self.cycles = set() # Store unique cycles
6
7     def findCycle(self):
8         # Try finding both clockwise and counterclockwise cycles
9         self.solve([self.start], self.start)
10        self.solve([self.start], self.start, reverse=True)
11
12        # Display the cycles
13        if self.cycles:
14            for cycle in self.cycles:
15                return
16            self.hasCycle = True
17
18    def solve(self, path, vertex, reverse=False):
19        if len(path) == N:
20            if adjacencyM[vertex][self.start] == 1:
21                cycle = path + [self.start] # Form the full cycle
22                cycle_names = [vertices[v] for v in cycle]
```

	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! ✓

Final

Marks for this submission: 20.00/20.00.

Question 2

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	pattern occurs at shift = 0
ABA	pattern occurs at shift = 4

Answer: (penalty regime: 0 %)

Reset answer

```

1 def preprocess_strong_suffix(shift, bpos, pat, m):
2     ##### Add your Code here #####
3     i=m
4     j=m+1
5     bpos[i]=j
6     while i>0:
7         while j<=m and pat[i-1]!=pat[j-1]:
8             if shift[j]==0:
9                 shift[j]=j-i
10                j=bpos[j]
11            i-=1
12            j-=1
13        bpos[i]=j
14 def preprocess_case2(shift, bpos, pat, m):
15     j = bpos[0]
16     for i in range(m + 1):
17         if shift[i] == 0:
18             shift[i] = j
19         if i == j:
20             j = bpos[j]
21 def search(text, pat):
22     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! ✓

Completed

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

```

1 def KMPSearch(pat, txt):
2     M=len(pat)
3     N=len(txt)
4     lps=[0]*M
5     j=0
6     computeLPSArray(pat,M,lps)
7     i=0
8     while(N-i)>=(M-j):
9         if pat[j]==txt[i]:
10             i+=1
11             j+=1
12         if j==M:
13             print("Found pattern at index "+str(i-j))
14             j = lps[j-1]
15         elif i<N and pat[j]!=txt[i]:
16             if j!=0:
17                 j=lps[j-1]
18             else:
19                 i+=1
20
21 def computeLPSArray(pat, M, lps):
22     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 **4**

Not answered

Mark 0.00 out of 20.00

Write a python program to implement merge sort using iterative approach on the given list of float values.

For example:

Test	Input	Result
Merge_Sort(S)	5 10.2 21.3 3.5 7.8 9.8	The Original array is: [10.2, 21.3, 3.5, 7.8, 9.8] Array after sorting is: [3.5, 7.8, 9.8, 10.2, 21.3]
Merge_Sort(S)	6 20.3 41.2 5.3 6.2 8.1 65.2	The Original array is: [20.3, 41.2, 5.3, 6.2, 8.1, 65.2] Array after sorting is: [5.3, 6.2, 8.1, 20.3, 41.2, 65.2]

Answer: (penalty regime: 0 %)

1	
---	--

Question 5

Correct

Mark 20.00 out of 20.00

Write a python program to implement knight tour problem using warnsdorff's algorithm

For example:

Test	Input	Result
a.warnsdorff((x,y))	8 8 3 3	board: [21, 32, 17, 30, 39, 36, 15, 42] [18, 29, 20, 35, 16, 41, 54, 37] [33, 22, 31, 40, 53, 38, 43, 14] [28, 19, 34, 1, 44, 49, 60, 55] [23, 2, 27, 52, 61, 56, 13, 50] [8, 5, 24, 45, 48, 51, 62, 59] [3, 26, 7, 10, 57, 64, 47, 12] [6, 9, 4, 25, 46, 11, 58, 63]

Answer: (penalty regime: 0 %)

Reset answer

```

1 KNIGHT_MOVES = [(2, 1), (1, 2), (-1, 2), (-2, 1), (-2, -1), (-1, -2), (1, -2), (2, -1)]
2 class KnightTour:
3     def __init__(self, board_size):
4         self.board_size = board_size # tuple
5         self.board = []
6         for i in range(board_size[0]):
7             temp = []
8             for j in range(board_size[1]):
9                 temp.append(0)
10            self.board.append(temp) # empty cell
11            self.move = 1
12
13        def print_board(self):
14            print('board:')
15            for i in range(self.board_size[0]):
16                print(self.board[i])
17
18        def warnsdorff(self, start_pos, GUI=False):
19            x,y = start_pos
20            self.board[x][y] = self.move
21
22

```

	Test	Input	Expected	Got	
✓	a.warnsdorff((x,y))	8 8 3 3	board: [21, 32, 17, 30, 39, 36, 15, 42] [18, 29, 20, 35, 16, 41, 54, 37] [33, 22, 31, 40, 53, 38, 43, 14] [28, 19, 34, 1, 44, 49, 60, 55] [23, 2, 27, 52, 61, 56, 13, 50] [8, 5, 24, 45, 48, 51, 62, 59] [3, 26, 7, 10, 57, 64, 47, 12] [6, 9, 4, 25, 46, 11, 58, 63]	board: [21, 32, 17, 30, 39, 36, 15, 42] [18, 29, 20, 35, 16, 41, 54, 37] [33, 22, 31, 40, 53, 38, 43, 14] [28, 19, 34, 1, 44, 49, 60, 55] [23, 2, 27, 52, 61, 56, 13, 50] [8, 5, 24, 45, 48, 51, 62, 59] [3, 26, 7, 10, 57, 64, 47, 12] [6, 9, 4, 25, 46, 11, 58, 63]	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.