

TT DS PYTHON MODULE-21

Started on	Friday, 16 May 2025, 8:51 AM
State	Finished
Completed on	Friday, 16 May 2025, 8:57 AM
Time taken	5 mins 22 secs
Grade	80.00 out of 100.00

Question 1

Correct

Mark 20.00 out of 20.00

Flag question

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), (0, 2), (1, 4), (2, 5), (3, 6), (4, 7), (5, 8), (6, 9), (7, 10), (8, 11), (9, 12), (10, 13), (11, 14), (12, 15), (13, 16), (14, 17), (15, 18), (16, 19), (17, 20), (18, 21), (19, 22), (20, 23), (21, 24), (22, 25), (23, 26), (24, 27), (25, 28), (26, 29), (27, 30), (28, 31), (29, 32), (30, 33), (31, 34), (32, 35), (33, 36), (34, 37), (35, 38), (36, 39), (37, 40), (38, 41), (39, 42), (40, 43), (41, 44), (42, 45), (43, 46), (44, 47), (45, 48), (46, 49), (47, 50), (48, 51), (49, 52), (50, 53), (51, 54), (52, 55), (53, 56), (54, 57), (55, 58), (56, 59), (57, 60), (58, 61), (59, 62), (60, 63), (61, 64), (62, 65), (63, 66), (64, 67), (65, 68), (66, 69), (67, 70), (68, 71), (69, 72), (70, 73), (71, 74), (72, 75), (73, 76), (74, 77), (75, 78), (76, 79), (77, 80), (78, 81), (79, 82), (80, 83), (81, 84), (82, 85), (83, 86), (84, 87), (85, 88), (86, 89), (87, 90), (88, 91), (89, 92), (90, 93), (91, 94), (92, 95), (93, 96), (94, 97), (95, 98), (96, 99), (97, 100), (98, 101), (99, 102), (100, 103), (101, 104), (102, 105), (103, 106), (104, 107), (105, 108), (106, 109), (107, 110), (108, 111), (109, 112), (110, 113), (111, 114), (112, 115), (113, 116), (114, 117), (115, 118), (116, 119), (117, 120), (118, 121), (119, 122), (120, 123), (121, 124), (122, 125), (123, 126), (124, 127), (125, 128), (126, 129), (127, 130), (128, 131), (129, 132), (130, 133), (131, 134), (132, 135), (133, 136), (134, 137), (135, 138), (136, 139), (137, 140), (138, 141), (139, 142), (140, 143), (141, 144), (142, 145), (143, 146), (144, 147), (145, 148), (146, 149), (147, 150), (148, 151), (149, 152), (150, 153), (151, 154), (152, 155), (153, 156), (154, 157), (155, 158), (156, 159), (157, 160), (158, 161), (159, 162), (160, 163), (161, 164), (162, 165), (163, 166), (164, 167), (165, 168), (166, 169), (167, 170), (168, 171), (169, 172), (170, 173), (171, 174), (172, 175), (173, 176), (174, 177), (175, 178), (176, 179), (177, 180), (178, 181), (179, 182), (180, 183), (181, 184), (182, 185), (183, 186), (184, 187), (185, 188), (186, 189), (187, 190), (188, 191), (189, 192), (190, 193), (191, 194), (192, 195), (193, 196), (194, 197), (195, 198), (196, 199), (197, 200), (198, 201), (199, 202), (200, 203), (201, 204), (202, 205), (203, 206), (204, 207), (205, 208), (206, 209), (207, 210), (208, 211), (209, 212), (210, 213), (211, 214), (212, 215), (213, 216), (214, 217), (215, 218), (216, 219), (217, 220), (218, 221), (219, 222), (220, 223), (221, 224), (222, 225), (223, 226), (224, 227), (225, 228), (226, 229), (227, 230), (228, 231), (229, 232), (230, 233), (231, 234), (232, 235), (233, 236), (234, 237), (235, 238), (236, 239), (237, 240), (238, 241), (239, 242), (240, 243), (241, 244), (242, 245), (243, 246), (244, 247), (245, 248), (246, 249), (247, 250), (248, 251), (249, 252), (250, 253), (251, 254), (252, 255), (253, 256), (254, 257), (255, 258), (256, 259), (257, 260), (258, 261), (259, 262), (260, 263), (261, 264), (262, 265), (263, 266), (264, 267), (265, 268), (266, 269), (267, 270), (268, 271), (269, 272), (270, 273), (271, 274), (272, 275), (273, 276), (274, 277), (275, 278), (276, 279), (277, 280), (278, 281), (279, 282), (280, 283), (281, 284), (282, 285), (283, 286), (284, 287), (285, 288), (286, 289), (287, 290), (288, 291), (289, 292), (290, 293), (291, 294), (292, 295), (293, 296), (294, 297), (295, 298), (296, 299), (297, 300), (298, 301), (299, 302), (300, 303), (301, 304), (302, 305), (303, 306), (304, 307), (305, 308), (306, 309), (307, 310), (308, 311), (309, 312), (310, 313), (311, 314), (312, 315), (313, 316), (314, 317), (315, 318), (316, 319), (317, 320), (318, 321), (319, 322), (320, 323), (321, 324), (322, 325), (323, 326), (324, 327), (325, 328), (326, 329), (327, 330), (328, 331), (329, 332), (330, 333), (331, 334), (332, 335), (333, 336), (334, 337), (335, 338), (336, 339), (337, 340), (338, 341), (339, 342), (340, 343), (341, 344), (342, 345), (343, 346), (344, 347), (345, 348), (346, 349), (347, 350), (348, 351), (349, 352), (350, 353), (351, 354), (352, 355), (353, 356), (354, 357), (355, 358), (356, 359), (357, 360), (358, 361), (359, 362), (360, 363), (361, 364), (362, 365), (363, 366), (364, 367), (365, 368), (366, 369), (367, 370), (368, 371), (369, 372), (370, 373), (371, 374), (372, 375), (373, 376), (374, 377), (375, 378), (376, 379), (377, 380), (378, 381), (379, 382), (380, 383), (381, 384), (382, 385), (383, 386), (384, 387), (385, 388), (386, 389), (387, 390), (388, 391), (389, 392), (390, 393), (391, 394), (392, 395), (393, 396), (394, 397), (395, 398), (396, 399), (397, 400), (398, 401), (399, 402), (400, 403), (401, 404), (402, 405), (403, 406), (404, 407), (405, 408), (406, 409), (407, 41

Answer: (penalty regime: 0 %)

Reset answer

```

1 import sys
2 class KnightsTour:
3     def __init__(self, width, height):
4         self.w = width
5         self.h = height
6         self.board = []
7         self.generate_board()
8
9     def generate_board(self):
10        for i in range(self.h):
11            self.board.append([0]*self.w)
12
13    def print_board(self):
14
15        for elem in self.board:
16            print (elem)
17
18    def generate_legal_moves(self, cur_pos):
19        possible_pos = []
20        move_offsets = [(1, 2), (1, -2), (-1, 2), (-1, -2),
21                        (2, 1), (2, -1), (-2, 1), (-2, -1)]

```

	Input	Expected
	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, Done!]
	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, Done!]

Passed all tests!

Correct

Marks for this submission: 20.00/20.00.

Question 2

Not answered

Mark 0.00 out of 20.00

Flag question

Write a Python program to print the following pattern based on the given input.

input:6

output:

★
★★
★★★
★★★★

For example:

Input	Result
6	* ** *** **** *****

Answer: (penalty regime: 0 %)

1

Question 3

Correct

Mark 20.00 out of 20.00

Flag question

Write a Python program for Bad Character Heuristic of Boyer Moore String Matching Algorithm

For example:

Input	Result
ABAAAABCD ABC	Pattern occur at shift = 5

Answer: (penalty regime: 0 %)

Reset answer

```

1 NO_OF_CHARS = 256
2 def badCharHeuristic(string, size):
3     ##### Add your Code Here #####
4     #Start here
5     badChar = [-1]*NO_OF_CHARS
6     for i in range(size):
7         badChar[ord(string[i])] = i;
8     return badChar
9     #End here
10 def search(txt, pat):
11     m = len(pat)
12     n = len(txt)
13     badChar = badCharHeuristic(pat, m)
14     s = 0
15     while(s <= n-m):
16         j = m-1
17         while j>=0 and pat[j] == txt[s+j]:
18             j -= 1
19         if j<0:
20             print("Pattern occur at shift = {}".format(s))
21             s += (m-badChar[ord(txt[s+m])] if s+m<n else 1)
22         else:

```

	Input	Expected	Got	
	ABAAAABCD ABC	Pattern occur at shift = 5	Pattern occur at shift = 5	

Passed all tests!

Correct

Marks for this submission: 20.00/20.00.

Question 4

Correct

Mark 20.00 out of 20.00

Flag question

Write a python program to implement pattern matching on the given string using Brute Force algorithm.

For example:

Test	Input	Result
BF(a1,a2)	abcaaaabbbbccabcbabdbcsbbbbbnnnccabcbaba	12

Answer: (penalty regime: 0 %)

Reset answer

```

1
2 def BF(s1,s2):
3     ##### Add your code here #####
4     #Start here
5     i = 0
6     j = 0
7     while(i < len(s1) and j < len(s2)):
8         if(s1[i] == s2[j]):
9             i += 1
10            j += 1
11        else:
12            i = i - j + 1
13            j = 0
14        if(j >= len(s2)):
15            return i - len(s2)
16        else:
17            return 0
18    #End here
19 if __name__ == "__main__":
20     a1=input()
21     a2=input()
22     b=BF(a1,a2)

```

Test	Input	Expected	Got
BF(a1,a2)	abcaaaabbbbccabcbabdbcsbbbbbnnnccabcbaba	12	12

Passed all tests!

Correct

Marks for this submission: 20.00/20.00.

Question 5

Correct

Mark 20.00 out of 20.00

Flag question

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

```

1 class Graph():
2     def __init__(self, vertices):
3         self.graph = [[0 for column in range(vertices)]
4                        for row in range(vertices)]
5         self.V = vertices
6     def isSafe(self, v, pos, path):
7         if self.graph[ path[pos-1] ][v] == 0:
8             return False
9         for vertex in path:
10            if vertex == v:
11                return False
12
13        return True
14    def hamCycleUtil(self, path, pos):
15        #####Add your code here#####

```

```
16 #Start here
17 if pos == self.V:
18     if self.graph[ path[pos-1] ][ path[0] ] == 1:
19         return True
20     else:
21         return False
22 for v in range(1,self.V):
```

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!

Submit

Marks for this submission: 20.00/20.00.

[Finish](#)