

AI Assisted Coding

Assignment – 8.2

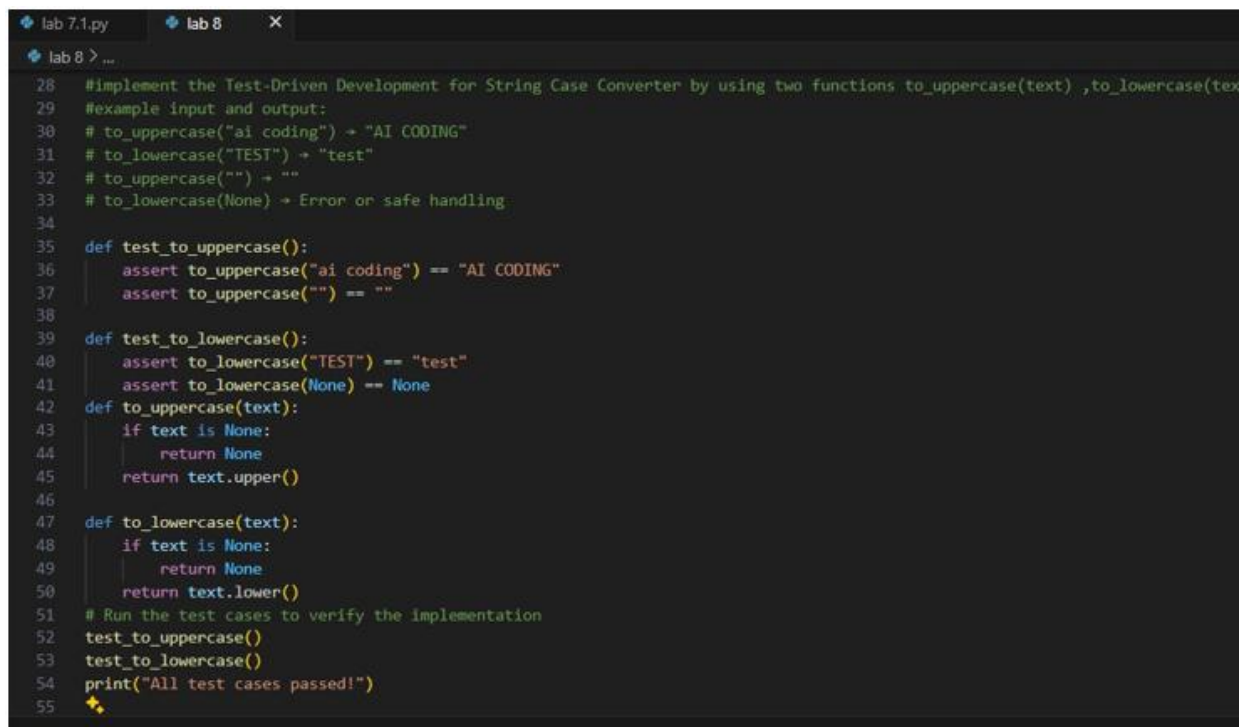
P.Manikanta || 2303A51271 || Batch:- 8

Task 1 – Test-Driven Development for Even/Odd Number Validator

- Use AI tools to first generate test cases for a function `is_even(n)` and then implement the function so that it satisfies all generated tests.

Requirements:

- Input must be an integer
- Handle zero, negative numbers, and large integers



```
lab 7.1.py lab 8 X
lab 8 > ...
28 #implement the Test-Driven Development for String Case Converter by using two functions to_uppercase(text) ,to_lowercase(text)
29 #example input and output:
30 # to_uppercase("ai coding") -> "AI CODING"
31 # to_lowercase("TEST") -> "test"
32 # to_uppercase("") -> ""
33 # to_lowercase(None) -> Error or safe handling
34
35 def test_to_uppercase():
36     assert to_uppercase("ai coding") == "AI CODING"
37     assert to_uppercase("") == ""
38
39 def test_to_lowercase():
40     assert to_lowercase("TEST") == "test"
41     assert to_lowercase(None) == None
42
43 def to_uppercase(text):
44     if text is None:
45         return None
46     return text.upper()
47
48 def to_lowercase(text):
49     if text is None:
50         return None
51     return text.lower()
52
53 # Run the test cases to verify the implementation
54 test_to_uppercase()
55 test_to_lowercase()
56 print("All test cases passed!")
```

Task 2 – Test-Driven Development for String Case Converter

- Ask AI to generate test cases for two functions:
- `to_uppercase(text)`

- to_lowercase(text) Requirements:
- Handle empty strings
- Handle mixed-case input
- Handle invalid inputs such as numbers or None

```

8  # Implement Test-Driven Development for List Sum Calculator
9  # Function: sum_list(numbers)
10
11 # Example inputs and outputs:
12 # sum_list([1, 2, 3]) → 6
13 # sum_list([]) → 0
14 # sum_list([-1, 5, -4]) → 0
15 # sum_list([2, "a", 3]) → 5
16
17
18 # Step 1: Write test cases
19 def test_sum_list():
20     assert sum_list([1, 2, 3]) == 6
21     assert sum_list([]) == 0
22     assert sum_list([-1, 5, -4]) == 0
23     assert sum_list([2, "a", 3]) == 5
24
25
26 # Step 2: Implement the function
27 def sum_list(numbers):
28     total = 0
29     for num in numbers:
30         total += num
31     return total
32
33
34 # Run tests
35 if __name__ == '__main__':
36     test_sum_list()
37     print("All test cases passed!")
38
39
40 # Run the function
41 if __name__ == '__main__':
42     numbers = [1, 2, 3]
43     result = sum_list(numbers)
44     print(f"Sum of {numbers} is {result}")
45
46
47 # Run the function with mixed input
48 if __name__ == '__main__':
49     numbers = [2, "a", 3]
50     result = sum_list(numbers)
51     print(f"Sum of {numbers} is {result}")
52
53
54 # Run the function with empty list
55 if __name__ == '__main__':
56     numbers = []
57     result = sum_list(numbers)
58     print(f"Sum of {numbers} is {result}")
59
60
61 # Run the function with negative numbers
62 if __name__ == '__main__':
63     numbers = [-1, 5, -4]
64     result = sum_list(numbers)
65     print(f"Sum of {numbers} is {result}")
66
67
68 # Run the function with None
69 if __name__ == '__main__':
70     numbers = [None, 1, 2, 3]
71     result = sum_list(numbers)
72     print(f"Sum of {numbers} is {result}")
73
74
75 # Run the function with string
76 if __name__ == '__main__':
77     numbers = ["1", "2", "3"]
78     result = sum_list(numbers)
79     print(f"Sum of {numbers} is {result}")
80
81
82 # Run the function with float
83 if __name__ == '__main__':
84     numbers = [1.5, 2.5, 3.5]
85     result = sum_list(numbers)
86     print(f"Sum of {numbers} is {result}")
87
88
89 # Run the function with mixed types
90 if __name__ == '__main__':
91     numbers = [1, 2, 3, "a", 4, 5, "b", 6, 7, 8, 9, 10]
92     result = sum_list(numbers)
93     print(f"Sum of {numbers} is {result}")
94
95
96 # Run the function with large numbers
97 if __name__ == '__main__':
98     numbers = [1000000, 2000000, 3000000]
99     result = sum_list(numbers)
100    print(f"Sum of {numbers} is {result}")
101
102
103 # Run the function with small numbers
104 if __name__ == '__main__':
105    numbers = [0.1, 0.2, 0.3]
106    result = sum_list(numbers)
107    print(f"Sum of {numbers} is {result}")
108
109
110 # Run the function with negative large numbers
111 if __name__ == '__main__':
112    numbers = [-1000000, -2000000, -3000000]
113    result = sum_list(numbers)
114    print(f"Sum of {numbers} is {result}")
115
116
117 # Run the function with negative small numbers
118 if __name__ == '__main__':
119    numbers = [-0.1, -0.2, -0.3]
120    result = sum_list(numbers)
121    print(f"Sum of {numbers} is {result}")
122
123
124 # Run the function with mixed large and small numbers
125 if __name__ == '__main__':
126    numbers = [1000000, 0.1, 1000000000, 0.0000001]
127    result = sum_list(numbers)
128    print(f"Sum of {numbers} is {result}")
129
130
131 # Run the function with mixed positive and negative numbers
132 if __name__ == '__main__':
133    numbers = [1, -2, 3, -4, 5, -6, 7, -8, 9, -10]
134    result = sum_list(numbers)
135    print(f"Sum of {numbers} is {result}")
136
137
138 # Run the function with mixed positive and negative large numbers
139 if __name__ == '__main__':
140    numbers = [1000000, -2000000, 3000000, -4000000, 5000000, -6000000, 7000000, -8000000, 9000000, -10000000]
141    result = sum_list(numbers)
142    print(f"Sum of {numbers} is {result}")
143
144
145 # Run the function with mixed positive and negative small numbers
146 if __name__ == '__main__':
147    numbers = [0.1, -0.2, 0.3, -0.4, 0.5, -0.6, 0.7, -0.8, 0.9, -1.0]
148    result = sum_list(numbers)
149    print(f"Sum of {numbers} is {result}")
150
151
152 # Run the function with mixed positive and negative mixed types
153 if __name__ == '__main__':
154    numbers = [1, -2, 3, -4, 5, -6, 7, -8, 9, -10, "a", 11, -12, 13, -14, 15, -16, 17, -18, 19, -20]
155    result = sum_list(numbers)
156    print(f"Sum of {numbers} is {result}")
157
158
159 # Run the function with mixed positive and negative float
160 if __name__ == '__main__':
161    numbers = [1.5, -2.5, 3.5, -4.5, 5.5, -6.5, 7.5, -8.5, 9.5, -10.5]
162    result = sum_list(numbers)
163    print(f"Sum of {numbers} is {result}")
164
165
166 # Run the function with mixed positive and negative float and integer
167 if __name__ == '__main__':
168    numbers = [1.5, 2, -3.5, 4, -5.5, 6, -7.5, 8, -9.5, 10, -11.5, 12, -13.5, 14, -15.5, 16, -17.5, 18, -19.5, 20, -21.5]
169    result = sum_list(numbers)
170    print(f"Sum of {numbers} is {result}")
171
172
173 # Run the function with mixed positive and negative float and integer and string
174 if __name__ == '__main__':
175    numbers = [1.5, 2, -3.5, 4, -5.5, 6, -7.5, 8, -9.5, 10, -11.5, 12, -13.5, 14, -15.5, 16, -17.5, 18, -19.5, 20, -21.5, "a", 22, -23.5, 24, -25.5, 26, -27.5, 28, -29.5, 30, -31.5, 32, -33.5, 34, -35.5, 36, -37.5, 38, -39.5, 40, -41.5, 42, -43.5, 44, -45.5, 46, -47.5, 48, -49.5, 50, -51.5, 52, -53.5, 54, -55.5, 56, -57.5, 58, -59.5, 60, -61.5, 62, -63.5, 64, -65.5, 66, -67.5, 68, -69.5, 70, -71.5, 72, -73.5, 74, -75.5, 76, -77.5, 78, -79.5, 80, -81.5, 82, -83.5, 84, -85.5, 86, -87.5, 88, -89.5, 90, -91.5, 92, -93.5, 94, -95.5, 96, -97.5, 98, -99.5, 100, -101.5, 102, -103.5, 104, -105.5, 106, -107.5, 108, -109.5, 110, -111.5, 112, -113.5, 114, -115.5, 116, -117.5, 118, -119.5, 120, -121.5, 122, -123.5, 124, -125.5, 126, -127.5, 128, -129.5, 130, -131.5, 132, -133.5, 134, -135.5, 136, -137.5, 138, -139.5, 140, -141.5, 142, -143.5, 144, -145.5, 146, -147.5, 148, -149.5, 150, -151.5, 152, -153.5, 154, -155.5, 156, -157.5, 158, -159.5, 160, -161.5, 162, -163.5, 164, -165.5, 166, -167.5, 168, -169.5, 170, -171.5, 172, -173.5, 174, -175.5, 176, -177.5, 178, -179.5, 180, -181.5, 182, -183.5, 184, -185.5, 186, -187.5, 188, -189.5, 190, -191.5, 192, -193.5, 194, -195.5, 196, -197.5, 198, -199.5, 200, -201.5, 202, -203.5, 204, -205.5, 206, -207.5, 208, -209.5, 210, -211.5, 212, -213.5, 214, -215.5, 216, -217.5, 218, -219.5, 220, -221.5, 222, -223.5, 224, -225.5, 226, -227.5, 228, -229.5, 230, -231.5, 232, -233.5, 234, -235.5, 236, -237.5, 238, -239.5, 240, -241.5, 242, -243.5, 244, -245.5, 246, -247.5, 248, -249.5, 250, -251.5, 252, -253.5, 254, -255.5, 256, -257.5, 258, -259.5, 260, -261.5, 262, -263.5, 264, -265.5, 266, -267.5, 268, -269.5, 270, -271.5, 272, -273.5, 274, -275.5, 276, -277.5, 278, -279.5, 280, -281.5, 282, -283.5, 284, -285.5, 286, -287.5, 288, -289.5, 290, -291.5, 292, -293.5, 294, -295.5, 296, -297.5, 298, -299.5, 300, -301.5, 302, -303.5, 304, -305.5, 306, -307.5, 308, -309.5, 310, -311.5, 312, -313.5, 314, -315.5, 316, -317.5, 318, -319.5, 320, -321.5, 322, -323.5, 324, -325.5, 326, -327.5, 328, -329.5, 330, -331.5, 332, -333.5, 334, -335.5, 336, -337.5, 338, -339.5, 340, -341.5, 342, -343.5, 344, -345.5, 346, -347.5, 348, -349.5, 350, -351.5, 352, -353.5, 354, -355.5, 356, -357.5, 358, -359.5, 360, -361.5, 362, -363.5, 364, -365.5, 366, -367.5, 368, -369.5, 370, -371.5, 372, -373.5, 374, -375.5, 376, -377.5, 378, -379.5, 380, -381.5, 382, -383.5, 384, -385.5, 386, -387.5, 388, -389.5, 390, -391.5, 392, -393.5, 394, -395.5, 396, -397.5, 398, -399.5, 400, -401.5, 402, -403.5, 404, -405.5, 406, -407.5, 408, -409.5, 410, -411.5, 412, -413.5, 414, -415.5, 416, -417.5, 418, -419.5, 420, -421.5, 422, -423.5, 424, -425.5, 426, -427.5, 428, -429.5, 430, -431.5, 432, -433.5, 434, -435.5, 436, -437.5, 438, -439.5, 440, -441.5, 442, -443.5, 444, -445.5, 446, -447.5, 448, -449.5, 450, -451.5, 452, -453.5, 454, -455.5, 456, -457.5, 458, -459.5, 460, -461.5, 462, -463.5, 464, -465.5, 466, -467.5, 468, -469.5, 470, -471.5, 472, -473.5, 474, -475.5, 476, -477.5, 478, -479.5, 480, -481.5, 482, -483.5, 484, -485.5, 486, -487.5, 488, -489.5, 490, -491.5, 492, -493.5, 494, -495.5, 496, -497.5, 498, -499.5, 500, -501.5, 502, -503.5, 504, -505.5, 506, -507.5, 508, -509.5, 510, -511.5, 512, -513.5, 514, -515.5, 516, -517.5, 518, -519.5, 520, -521.5, 522, -523.5, 524, -525.5, 526, -527.5, 528, -529.5, 530, -531.5, 532, -533.5, 534, -535.5, 536, -537.5, 538, -539.5, 540, -541.5, 542, -543.5, 544, -545.5, 546, -547.5, 548, -549.5, 550, -551.5, 552, -553.5, 554, -555.5, 556, -557.5, 558, -559.5, 560, -561.5, 562, -563.5, 564, -565.5, 566, -567.5, 568, -569.5, 570, -571.5, 572, -573.5, 574, -575.5, 576, -577.5, 578, -579.5, 580, -581.5, 582, -583.5, 584, -585.5, 586, -587.5, 588, -589.5, 590, -591.5, 592, -593.5, 594, -595.5, 596, -597.5, 598, -599.5, 600, -601.5, 602, -603.5, 604, -605.5, 606, -607.5, 608, -609.5, 610, -611.5, 612, -613.5, 614, -615.5, 616, -617.5, 618, -619.5, 620, -621.5, 622, -623.5, 624, -625.5, 626, -627.5, 628, -629.5, 630, -631.5, 632, -633.5, 634, -635.5, 636, -637.5, 638, -639.5, 640, -641.5, 642, -643.5, 644, -645.5, 646, -647.5, 648, -649.5, 650, -651.5, 652, -653.5, 654, -655.5, 656, -657.5, 658, -659.5, 660, -661.5, 662, -663.5, 664, -665.5, 666, -667.5, 668, -669.5, 670, -671.5, 672, -673.5, 674, -675.5, 676, -677.5, 678, -679.5, 680, -681.5, 682, -683.5, 684, -685.5, 686, -687.5, 688, -689.5, 690, -691.5, 692, -693.5, 694, -695.5, 696, -697.5, 698, -699.5, 700, -701.5, 702, -703.5, 704, -705.5, 706, -707.5, 708, -709.5, 710, -711.5, 712, -713.5, 714, -715.5, 716, -717.5, 718, -719.5, 720, -721.5, 722, -723.5, 724, -725.5, 726, -727.5, 728, -729.5, 730, -731.5, 732, -733.5, 734, -735.5, 736, -737.5, 738, -739.5, 740, -741.5, 742, -743.5, 744, -745.5, 746, -747.5, 748, -749.5, 750, -751.5, 752, -753.5, 754, -755.5, 756, -757.5, 758, -759.5, 760, -761.5, 762, -763.5, 764, -765.5, 766, -767.5, 768, -769.5, 770, -771.5, 772, -773.5, 774, -775.5, 776, -777.5, 778, -779.5, 780, -781.5, 782, -783.5, 784, -785.5, 786, -787.5, 788, -789.5, 790, -791.5, 792, -793.5, 794, -795.5, 796, -797.5, 798, -799.5, 800, -801.5, 802, -803.5, 804, -805.5, 806, -807.5, 808, -809.5, 810, -811.5, 812, -813.5, 814, -815.5, 816, -817.5, 818, -819.5, 820, -821.5, 822, -823.5, 824, -825.5, 826, -827.5, 828, -829.5, 830, -831.5, 832, -833.5, 834, -835.5, 836, -837.5, 838, -839.5, 840, -841.5, 842, -843.5, 844, -845.5, 846, -847.5, 848, -849.5, 850, -851.5, 852, -853.5, 854, -855.5, 856, -857.5, 858, -859.5, 860, -861.5, 862, -863.5, 864, -865.5, 866, -867.5, 868, -869.5, 870, -871.5, 872, -873.5, 874, -875.5, 876, -877.5, 878, -879.5, 880, -881.5, 882, -883.5, 884, -885.5, 886, -887.5, 888, -889.5, 890, -891.5, 892, -893.5, 894, -895.5, 896, -897.5, 898, -899.5, 900, -901.5, 902, -903.5, 904, -905.5, 906, -907.5, 908, -909.5, 910, -911.5, 912, -913.5, 914, -915.5, 916, -917.5, 918, -919.5, 920, -921.5, 922, -923.5, 924, -925.5, 926, -927.5, 928, -929.5, 930, -931.5, 932, -933.5, 934, -935.5, 936, -937.5, 938, -939.5, 940, -941.5, 942, -943.5, 944, -945.5, 946, -947.5, 948, -949.5, 950, -951.5, 952, -953.5, 954, -955.5, 956, -957.5, 958, -959.5, 960, -961.5, 962, -963.5, 964, -965.5, 966, -967.5, 968, -969.5, 970, -971.5, 972, -973.5, 974, -975.5, 976, -977.5, 978, -979.5, 980, -981.5, 982, -983.5, 984, -985.5, 986, -987.5, 988, -989.5, 990, -991.5, 992, -993.5, 994, -995.5, 996, -997.5, 998, -999.5, 1000, -1001.5, 1002, -1003.5, 1004, -1005.5, 1006, -1007.5, 1008, -1009.5, 1010, -1011.5, 1012, -1013.5, 1014, -1015.5, 1016, -1017.5, 1018, -1019.5, 1020, -1021.5, 1022, -1023.5, 1024, -1025.5, 1026, -1027.5, 1028, -1029.5, 1030, -1031.5, 1032, -1033.5, 1034, -1035.5, 1036, -1037.5, 1038, -1039.5, 1040, -1041.5, 1042, -1043.5, 1044, -1045.5, 1046, -1047.5, 1048, -1049.5, 1050, -1051.5, 1052, -1053.5, 1054, -1055.5, 1056, -1057.5, 1058, -1059.5, 1060, -1061.5, 1062, -1063.5, 1064, -1065.5, 1066, -1067.5, 1068, -1069.5, 1070, -1071.5, 1072, -1073.5, 1074, -1075.5, 1076, -1077.5, 1078, -1079.5, 1080, -1081.5, 1082, -1083.5, 1084, -1085.5, 1086, -1087.5, 1088, -1089.5, 1090, -1091.5, 1092, -1093.5, 1094, -1095.5, 1096, -1097.5, 1098, -1099.5, 1100, -1101.5, 1102, -1103.5, 1104, -1105.5, 1106, -1107.5, 1108, -1109.5, 1110, -1111.5, 1112, -1113.5, 1114, -1115.5, 1116, -1117.5, 1118, -1119.5, 1120, -1121.5, 1122, -1123.5, 1124, -1125.5, 1126, -1127.5, 1128, -1129.5, 1130, -1131.5, 1132, -1133.5, 1134, -1135.5, 1136, -1137.5, 1138, -1139.5, 1140, -1141.5, 1142, -1143.5, 1144, -1145.5, 1146, -1147.5, 1148, -1149.5, 1150, -1151.5, 1152, -1153.5, 1154, -1155.5, 1156, -1157.5, 1158, -1159.5, 1160, -1161.5, 1162, -1163.5, 1164, -1165.5, 1166, -1167.5, 1168, -1169.5, 1170, -1171.5, 1172, -1173.5, 1174, -1175.5, 1176, -1177.5, 1178, -1179.5, 1180, -1181.5, 1182, -1183.5, 1184, -1185.5, 1186, -1187.5, 1188, -1189.5, 1190, -1191.5, 1192, -1193.5, 1194, -1195.5, 1196, -1197.5, 1198, -1199.5, 1200, -1201.5, 1202, -1203.5, 1204, -1205.5, 1206, -1207.5, 1208, -1209.5, 1210, -1211.5, 1212, -1213.5, 1214, -1215.5, 1216, -1217.5, 1218, -1219.5, 1220, -1221.5, 1222, -1223.5, 1224, -1225.5, 1226, -1227.5, 1228, -1229.5, 1230, -1231.5, 1232, -1233.5, 1234, -1235.5, 1236, -1237.5, 1238, -1239.5, 1240, -1241.5, 1242, -1243.5, 1244, -1245.5, 1246, -1247.5, 1248, -1249.5, 1250, -1251.5, 1252, -1253.5, 1254, -1255.5, 1256, -1257.5, 1258, -1259.5, 1260, -1261.5, 1262, -1263.5, 1264, -1265.5, 1266, -1267.5, 1268, -1269.5, 1270, -1271.5, 1272, -1273.5, 1274, -1275.5, 1276, -1277.5, 1278, -1279.5, 1280, -1281.5, 1282, -1283.5, 1284, -1285.5, 1286, -1287.5, 1288, -1289.5, 1290, -1291.5, 1292, -1293.5, 1294, -1295.5, 1296, -1297.5, 1298, -1299.5, 1300, -1301.5, 1302, -1303.5, 1304, -1305.5, 1306, -1307.5, 1308, -1309.5, 1310, -1311.5, 1312, -1313.5, 1314, -1315.5, 1316, -1317.5, 1318, -1319.5, 1320, -1321.5, 1322, -1323.5, 1324, -1325.5, 1326, -1327.5, 1328, -1329.5, 1330, -1331.5, 1332, -1333.5, 1334, -1335.5, 1336, -1337.5, 1338, -1339.5, 1340, -1341.5, 1342, -1343.5, 1344, -1345.5, 1346, -1347.5, 1348, -1349.5, 1350, -1351.5, 1352, -1353.5, 1354, -1355.5, 1356, -1357.5, 1358, -1359.5, 1360, -1361.5, 1362, -1363.5, 1364, -1365.5, 1366, -1367.5, 1368, -1369.5, 1370, -1371.5, 1372, -1373.5, 1374, -1375.5, 1376, -1377.5, 1378, -1379.5, 1380, -1381.5, 1382, -1383.5, 1384, -1385.5, 1386, -1387.5, 1388, -1389.5, 1390, -1391.5, 1392, -1393.5, 1394, -1395.5, 1396, -1397.5, 1398, -1399.5, 1400, -1401.5, 1402, -1403.5, 1404, -1405.5, 1406, -1407.5, 1408, -1409.5, 1410, -1411.5, 1412, -1413.5, 1414, -1415.5, 1416, -1417.5, 1418, -1419.5, 1420, -1421.5, 1422, -1423.5, 1424, -1425.5, 1426, -1427.5, 1428, -1429.5, 1430, -1431.5, 1432, -1433.5, 1434, -1435.5, 1436, -1437.5, 1438, -1439.5, 1440, -1441.5, 1442, -1443.5, 1444, -1445.5, 1446, -1447.5, 1448, -1449.5, 1450, -1451.5, 1452, -1453.5, 1454, -1455.5, 1456, -1457.5, 1458, -1459.5, 1460, -1461.5, 1462, -1463.5, 1464, -1465.5, 1466, -1467.5, 1468, -1469.5, 1470, -1471.5, 1472, -1473.5, 1474, -1475.5, 1476, -1477.5, 1478, -1479.5, 1480, -1481.5, 1482, -1483.5, 1484, -1485.5, 1486, -1487.5, 1488, -1489.5, 1490, -1491.5, 1492, -1493.5, 1494, -1495.5, 1496, -1497.5, 1498, -1499.5, 1500, -1501.5, 1502, -1503.5, 1504, -1505.5, 1506, -1507.5, 1508, -1509.5, 1510, -1511.5, 1512, -1513.5, 1514, -1515.5, 1516, -1517.5, 1518, -1519.5, 1520, -1521.5, 1522, -1523.5, 1524, -1525.5, 1526, -1527.5, 1528, -1529.5, 1530, -1531.5, 1532, -1533.5, 1534, -1535.5, 1536, -1537.5, 1538, -1539.5, 1540, -1541.5, 1542, -1543.5, 1544, -1545.5, 1546, -1547.5, 1548, -1549.5, 1550, -1551.5, 1552, -1553.5, 1554, -1555.5, 1556, -1557.5, 1558, -1559.5, 1560, -1561.5, 1562, -1563.5, 1564, -1565.5, 1566, -1567.5, 1568, -1569.5, 1570, -1571.5, 1572, -1573.5, 1574, -1575.5, 1576, -1577.5, 1578, -1579.5, 1580, -1581.5, 1582, -1583.5, 1584, -1585.5, 1586, -1587.5, 1588, -1589.5, 1590, -1591.5, 1592, -1593.5, 1594, -1595.5, 1596, -1597.5, 15
```

- Ignore or safely handle non-numeric values

Task 4 – Test Cases for Student Result Class

- Generate test cases for a StudentResult class with the following methods:

- add_marks(mark)
- calculate_average()
- get_result()

Requirements:

- Marks must be between 0 and 100
- Average $\geq 40 \rightarrow$ Pass, otherwise Fail

```
lab 7.1.py lab 8
lab 8 > ...
83 # Generate code for checking Test Cases for Student Result Class by using functions add_marks(mark), calculate_average(), get_result().
84 #Average ≥ 40 → Pass, otherwise Fail.
85 #example input and output:
86 # Marks: [60, 70, 80] → Average: 70 → Result: Pass
87 # Marks: [30, 35, 40] → Average: 35 → Result: Fail
88 # Marks: [-10] → Error
89 class StudentResult:
90     def __init__(self):
91         self.marks = []
92
93     def add_marks(self, mark):
94         if mark < 0:
95             raise ValueError("Marks cannot be negative")
96         self.marks.append(mark)
97
98     def calculate_average(self):
99         if not self.marks:
100             return 0
101         return sum(self.marks) / len(self.marks)
102
103     def get_result(self):
104         average = self.calculate_average()
105         return "Pass" if average >= 40 else "Fail"
106 # Test cases
107 def test_student_result():
108     student = StudentResult()
109     student.add_marks(60)
110     student.add_marks(70)
111     student.add_marks(80)
112     assert student.calculate_average() == 70
113     assert student.get_result() == "Pass"
114
115     student = StudentResult()
116     student.add_marks(30)
117     student.add_marks(35)
118     student.add_marks(40)
119     assert student.calculate_average() == 35
120     assert student.get_result() == "Fail"
121
122     student = StudentResult()
123     student.add_marks(-10)
124     assert student.get_result() == "Fail"
125
126 if __name__ == '__main__':
127     test_student_result()
128
129 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
All test cases passed!
PS C:\Users\komma\Desktop\3rd YEAR\AI-AC>
```

Task 5 – Test-Driven Development for Username Validator

Requirements:

- Minimum length: 5 characters
- No spaces allowed
- Only alphanumeric characters

```

132
133 #Generate Code for Test-Driven Development for Username Validator by following the rules:
134 # Minimum length: 5 characters
135 # No spaces allowed
136 # Only alphanumeric characters
137 #Example input and output:
138 # is_valid_username("user01") → True
139 # is_valid_username("ai") → False
140 # is_valid_username("user name") → False
141 # is_valid_username("user@123") → False
142 def test_is_valid_username():
143     assert is_valid_username("user01") == True
144     assert is_valid_username("ai") == False
145     assert is_valid_username("user name") == False
146     assert is_valid_username("user@123") == False
147
148 def is_valid_username(username):
149     if len(username) < 5:
150         return False
151     if " " in username:
152         return False
153     if not username.isalnum():
154         return False
155     return True
156 # Run the test cases to verify the implementation
157 test_is_valid_username()
158 print("All test cases passed!")
159
160

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

All test cases passed!
PS C:\Users\komma\Desktop\3rd YEAR\AI-AC>