

Assignment 11.2

Name :R Ajay

HT.NO : 2303A51072

B.NO : 16

Task Description -1 – (Stack Using AI Guidance)

- **Task:** With the help of AI, design and implement a Stack data structure supporting basic stack operations.

Expected Output:

- A Python Stack class supporting push, pop, peek, and empty-check operations with proper documentation.

The screenshot shows a code editor interface with multiple tabs open. The active tab is 'task1.py' which contains Python code for a stack implementation. The code includes docstrings and example usage. Below the code editor is a terminal window showing the execution of the code. On the right side, there is a sidebar with a 'CHAT' section titled 'DESIGNING A STACK DATA STRUCTURE IN PYTHON'. It includes a tip about using Plan mode, a message from an AI expert, requirements, and a list of tasks completed. At the bottom, there is a GPT-5.1 interface for generating build descriptions.

```
task1.py X task2.py task3.py task4.py task5.py

AI-ASSISTED_CODING > task1.py > Stack

1 """Simple Stack implementation in Python.
2 This module defines a Stack class with the following operations:
3 - push(item)
4 - pop()
5 - peek()
6 - is_empty()
7 It also includes example usage at the bottom.
8 """
9
10 class Stack:
11     """A simple LIFO (Last-In, First-Out) stack data structure.
12     Internally this uses a Python list to store elements.
13     """
14     def __init__(self):
15         """Initialize an empty stack.
16         """
17         self._items = []
18     def push(self, item):
19         """Add an item to the top of the stack.
20         Args:
21             item: The value to be pushed onto the stack.
22
23             Time Complexity: O(1) on average (amortized), because list append is O(1).
24
25             Current stack: Stack([10, 20, 30])
26
27             Top element (peek): 30
28
29             Popped elements:
30             Popped: 30
31             Popped: 20
32             Current stack: Stack([10])
33
34             Is stack empty? False
35             Stack size: 1
36
37             Optimizing tool selection...
38             Designing a Python Stack
39
40             Reviewed lab1.py
41
42             Now I'll write a Stack class with all required methods,
43             comments, and an example usage into lab1.py.
44
45             Updated lab1.py
46
47             Here's what I've added and how it meets your requirements:
48
49             task5.py + task1.py
50
51             Describe what to build next
52             GPT-5.1
53
54             Python 3.13 (64-bit) (v) Go Live Windsurf: (...) Prettier
```

Implemented a Stack class backed by a Python list.

Methods:

- `push(item)`: adds an element to the top.
- `pop()`: removes and returns the top element, raises `IndexError` if empty.
- `peek()`: returns (without removing) the top element, raises `IndexError` if empty.
- `is_empty()`: returns `True` if there are no elements.

Task Description -2 – (Queue Design)

- **Task: Use AI assistance to create a Queue data structure following FIFO principles**

Expected Output:

- **A complete Queue implementation including enqueue, dequeue, front element access, and size calculation**

The screenshot shows a Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows files task1.py, task2.py (the active file), task3.py, task4.py, and task5.py.
- Editor:** Task 2 contains code for a Queue class using collections.deque. The code includes comments explaining the implementation and example usage.
- Terminal:** Shows command-line output of running task2.py, demonstrating enqueue, dequeue, front, and size operations.
- Chat Panel:** Shows AI-assisted coding notes and requirements for the Queue design.
- Status Bar:** Shows Python 3.13 (64-bit) and GPT-5.1.

- Implemented a Queue class using `collections.deque` for efficient operations.
- Methods:

`enqueue(item)`: adds an element to the rear of the queue (FIFO).

`dequeue()`: removes and returns the front element; raises `IndexError` if empty.

`front()`: returns (without removing) the front element; raises `IndexError` if empty.

`size()`: returns the current number of elements.

`is_empty()`: helper to check if the queue has no elements.

Task Description -3 –(Singly Linked List Construction)

- Task: Utilize AI to build a singly linked list supporting insertion and traversal.

Expected Output:

- Correctly functioning linked list with node creation, insertion logic, and display functionality.

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows files task1.py, task2.py, task3.py, task4.py, and task5.py.
- Editor:** The main editor window contains the Python code for a singly linked list implementation. The code includes classes for Node and LinkedList, methods for insertion at the end, and traversal. It also includes comments explaining the implementation.
- Terminal:** The terminal window shows the command: C:\Users\akhil\OneDrive\Documents\Devops>python -u "c:\Users\akhil\OneDrive\Documents\Devops\AI-ASSISTED_CODING\task3.py". The output indicates elements are being inserted at the end: Inserting elements at the end: 10, 20, 30.
- Output:** The output window shows the linked list as [10, 20, 30].
- Problems:** No problems are listed.
- CHAT:** A sidebar titled "DESIGNING A STACK DATA STRUCTURE IN PYTHON" is open, showing a conversation with AI. The user asks for help implementing a singly linked list, and AI provides requirements and steps to implement it.
- Code Snippet Bar:** Task3.py is selected.
- Status Bar:** Shows Ln 129, Col 21, Spaces:4, UTF-8, CRLF, Python 3.13 (64-bit), GPT-5.1, and Prettier.

- Added Node and LinkedList classes.
 - Node holds data and a next reference that points to the next node (or None for the last node).
 - LinkedList:

- Maintains head (first node).
- `insert_at_end(data)`: creates a new node and links it at the end by walking from head to the last node and updating its next.
- `traverse()`: walks from head via next, collects data values into a Python list, and returns it.
- `display()`: prints the list in the form 10 -> 20 -> 30 -> None.

Task Description -4 – (Binary Search Tree Operations)

- Task: Implement a Binary Search Tree with AI support focusing on insertion and traversal.

Expected Output:

- BST program with correct node insertion and in-order traversal output.

The screenshot shows a code editor with multiple tabs open: task1.py, task2.py, task3.py, task4.py (the active tab), and task5.py. The code in task4.py implements a Binary Search Tree (BST) with Node and BinarySearchTree classes and their respective methods like __init__ and insert. A ChatGPT interface is visible on the right, providing requirements and a list of tasks to implement, such as creating a Node class and implementing BST insertion and traversal methods.

```

1  """Binary Search Tree (BST) Implementation in Python.
2
3  This file defines:
4  - Node class: represents a single node in the BST
5  - BinarySearchTree class: supports insertion and in-order traversal
6
7  Features implemented:
8  - Insert a value into the BST
9  - In-order traversal (left, Root, Right)
10
11 Example usage is provided at the bottom.
12 """
13
14
15 class Node:
16     """A single node in a Binary Search Tree.
17
18     Each node stores:
19         - data: the value stored at this node
20         - left: reference to the left child (values < data)
21         - right: reference to the right child (values > data)
22     """
23
24     def __init__(self, data):
25         self.data = data
26         self.left = None
27         self.right = None
28
29
30 class BinarySearchTree:
31     """Binary Search Tree (BST) with recursive insertion and in-order traversal."""
32
33     def __init__(self):
34         # Root of the BST. Initially, the tree is empty so root is None.
35         self.root = None
36
37     def insert(self, value):
38         """Insert a value into the BST.
39
40             Args:
41                 value (int): The value to be inserted into the BST.
42
43             Returns:
44                 None: The method does not return anything, but the BST is modified in-place.
45
46             Side Effects:
47                 - The BST's structure is modified to maintain the in-order traversal property.
48
49             Invariant:
50                 - All nodes in the left subtree have values less than the root's value.
51                 - All nodes in the right subtree have values greater than the root's value.
52
53             Examples:
54                 - inserting [20, 30, 70, 40, 60, 80] results in an in-order traversal of [20, 30, 40, 60, 70, 80].
55
56             Implementation Notes:
57                 - The insertion is performed recursively.
58                 - The in-order traversal is also implemented recursively.
59
60             Requirements:
61                 - Create a Node class
62                 - Implement BST insertion method
63                 - Implement in-order traversal
64                 - Add proper comments explaining recursion
65                 - Provide example with sample values
66                 - Show expected traversal output
67                 - Explain why in-order traversal gives sorted output
68
69             Test Cases:
70                 - Inserting [20, 30, 70, 40, 60, 80] should result in an in-order traversal of [20, 30, 40, 60, 70, 80].
71
72             Edge Cases:
73                 - Inserting an empty list should result in an empty tree.
74
75             Optimizations:
76                 - The BST is balanced using AVL or Red-Black trees for better performance.
77
78             Error Handling:
79                 - Handles invalid input types and values.
80
81             Performance:
82                 - Insertion and traversal operations are O(n) in the worst case.
83
84             Space Complexity:
85                 - O(n) for the tree structure itself.
86
87             Time Complexity:
88                 - O(n log n) for balanced trees due to rebalancing.
89
90             Memory Usage:
91                 - O(n) for storing the tree nodes.
92
93             API:
94                 - Node: __init__, data, left, right
95                 - BinarySearchTree: __init__, insert, in_order_traversal
96
97             License:
98                 - MIT License
99
100            Authors:
101                - Akhil Patel
102
103            Version:
104                - 1.0
105
106            Last Updated:
107                - 2023-10-15
108
109            Status:
110                - Production
111
112            Issues:
113                - None
114
115            Pull Requests:
116                - None
117
118            Contributors:
119                - None
120
121            Tests:
122                - None
123
124            Documentation:
125                - None
126
127            Changelog:
128                - None
129
130            Credits:
131                - None
132
133            References:
134                - None
135
136            Acknowledgments:
137                - None
138
139            Copyright:
140                - © 2023 Akhil Patel. All rights reserved.
141
142            License:
143                - MIT License
144
145            Version:
146                - 1.0
147
148            Last Updated:
149                - 2023-10-15
150
151            Status:
152                - Production
153
154            Issues:
155                - None
156
157            Pull Requests:
158                - None
159
160            Contributors:
161                - None
162
163            Tests:
164                - None
165
166            Documentation:
167                - None
168
169            Changelog:
170                - None
171
172            Credits:
173                - None
174
175            References:
176                - None
177
178            Acknowledgments:
179                - None
180
181            Copyright:
182                - © 2023 Akhil Patel. All rights reserved.
183
184            License:
185                - MIT License
186
187            Version:
188                - 1.0
189
190            Last Updated:
191                - 2023-10-15
192
193            Status:
194                - Production
195
196            Issues:
197                - None
198
199            Pull Requests:
200                - None
201
202            Contributors:
203                - None
204
205            Tests:
206                - None
207
208            Documentation:
209                - None
210
211            Changelog:
212                - None
213
214            Credits:
215                - None
216
217            References:
218                - None
219
220            Acknowledgments:
221                - None
222
223            Copyright:
224                - © 2023 Akhil Patel. All rights reserved.
225
226            License:
227                - MIT License
228
229            Version:
230                - 1.0
231
232            Last Updated:
233                - 2023-10-15
234
235            Status:
236                - Production
237
238            Issues:
239                - None
240
241            Pull Requests:
242                - None
243
244            Contributors:
245                - None
246
247            Tests:
248                - None
249
250            Documentation:
251                - None
252
253            Changelog:
254                - None
255
256            Credits:
257                - None
258
259            References:
260                - None
261
262            Acknowledgments:
263                - None
264
265            Copyright:
266                - © 2023 Akhil Patel. All rights reserved.
267
268            License:
269                - MIT License
270
271            Version:
272                - 1.0
273
274            Last Updated:
275                - 2023-10-15
276
277            Status:
278                - Production
279
280            Issues:
281                - None
282
283            Pull Requests:
284                - None
285
286            Contributors:
287                - None
288
289            Tests:
290                - None
291
292            Documentation:
293                - None
294
295            Changelog:
296                - None
297
298            Credits:
299                - None
300
301            References:
302                - None
303
304            Acknowledgments:
305                - None
306
307            Copyright:
308                - © 2023 Akhil Patel. All rights reserved.
309
310            License:
311                - MIT License
312
313            Version:
314                - 1.0
315
316            Last Updated:
317                - 2023-10-15
318
319            Status:
320                - Production
321
322            Issues:
323                - None
324
325            Pull Requests:
326                - None
327
328            Contributors:
329                - None
330
331            Tests:
332                - None
333
334            Documentation:
335                - None
336
337            Changelog:
338                - None
339
340            Credits:
341                - None
342
343            References:
344                - None
345
346            Acknowledgments:
347                - None
348
349            Copyright:
350                - © 2023 Akhil Patel. All rights reserved.
351
352            License:
353                - MIT License
354
355            Version:
356                - 1.0
357
358            Last Updated:
359                - 2023-10-15
360
361            Status:
362                - Production
363
364            Issues:
365                - None
366
367            Pull Requests:
368                - None
369
370            Contributors:
371                - None
372
373            Tests:
374                - None
375
376            Documentation:
377                - None
378
379            Changelog:
380                - None
381
382            Credits:
383                - None
384
385            References:
386                - None
387
388            Acknowledgments:
389                - None
390
391            Copyright:
392                - © 2023 Akhil Patel. All rights reserved.
393
394            License:
395                - MIT License
396
397            Version:
398                - 1.0
399
400            Last Updated:
401                - 2023-10-15
402
403            Status:
404                - Production
405
406            Issues:
407                - None
408
409            Pull Requests:
410                - None
411
412            Contributors:
413                - None
414
415            Tests:
416                - None
417
418            Documentation:
419                - None
420
421            Changelog:
422                - None
423
424            Credits:
425                - None
426
427            References:
428                - None
429
430            Acknowledgments:
431                - None
432
433            Copyright:
434                - © 2023 Akhil Patel. All rights reserved.
435
436            License:
437                - MIT License
438
439            Version:
440                - 1.0
441
442            Last Updated:
443                - 2023-10-15
444
445            Status:
446                - Production
447
448            Issues:
449                - None
450
451            Pull Requests:
452                - None
453
454            Contributors:
455                - None
456
457            Tests:
458                - None
459
460            Documentation:
461                - None
462
463            Changelog:
464                - None
465
466            Credits:
467                - None
468
469            References:
470                - None
471
472            Acknowledgments:
473                - None
474
475            Copyright:
476                - © 2023 Akhil Patel. All rights reserved.
477
478            License:
479                - MIT License
480
481            Version:
482                - 1.0
483
484            Last Updated:
485                - 2023-10-15
486
487            Status:
488                - Production
489
490            Issues:
491                - None
492
493            Pull Requests:
494                - None
495
496            Contributors:
497                - None
498
499            Tests:
500                - None
501
502            Documentation:
503                - None
504
505            Changelog:
506                - None
507
508            Credits:
509                - None
510
511            References:
512                - None
513
514            Acknowledgments:
515                - None
516
517            Copyright:
518                - © 2023 Akhil Patel. All rights reserved.
519
520            License:
521                - MIT License
522
523            Version:
524                - 1.0
525
526            Last Updated:
527                - 2023-10-15
528
529            Status:
530                - Production
531
532            Issues:
533                - None
534
535            Pull Requests:
536                - None
537
538            Contributors:
539                - None
540
541            Tests:
542                - None
543
544            Documentation:
545                - None
546
547            Changelog:
548                - None
549
550            Credits:
551                - None
552
553            References:
554                - None
555
556            Acknowledgments:
557                - None
558
559            Copyright:
560                - © 2023 Akhil Patel. All rights reserved.
561
562            License:
563                - MIT License
564
565            Version:
566                - 1.0
567
568            Last Updated:
569                - 2023-10-15
570
571            Status:
572                - Production
573
574            Issues:
575                - None
576
577            Pull Requests:
578                - None
579
580            Contributors:
581                - None
582
583            Tests:
584                - None
585
586            Documentation:
587                - None
588
589            Changelog:
590                - None
591
592            Credits:
593                - None
594
595            References:
596                - None
597
598            Acknowledgments:
599                - None
600
601            Copyright:
602                - © 2023 Akhil Patel. All rights reserved.
603
604            License:
605                - MIT License
606
607            Version:
608                - 1.0
609
610            Last Updated:
611                - 2023-10-15
612
613            Status:
614                - Production
615
616            Issues:
617                - None
618
619            Pull Requests:
620                - None
621
622            Contributors:
623                - None
624
625            Tests:
626                - None
627
628            Documentation:
629                - None
630
631            Changelog:
632                - None
633
634            Credits:
635                - None
636
637            References:
638                - None
639
640            Acknowledgments:
641                - None
642
643            Copyright:
644                - © 2023 Akhil Patel. All rights reserved.
645
646            License:
647                - MIT License
648
649            Version:
650                - 1.0
651
652            Last Updated:
653                - 2023-10-15
654
655            Status:
656                - Production
657
658            Issues:
659                - None
660
661            Pull Requests:
662                - None
663
664            Contributors:
665                - None
666
667            Tests:
668                - None
669
670            Documentation:
671                - None
672
673            Changelog:
674                - None
675
676            Credits:
677                - None
678
679            References:
680                - None
681
682            Acknowledgments:
683                - None
684
685            Copyright:
686                - © 2023 Akhil Patel. All rights reserved.
687
688            License:
689                - MIT License
690
691            Version:
692                - 1.0
693
694            Last Updated:
695                - 2023-10-15
696
697            Status:
698                - Production
699
700            Issues:
701                - None
702
703            Pull Requests:
704                - None
705
706            Contributors:
707                - None
708
709            Tests:
710                - None
711
712            Documentation:
713                - None
714
715            Changelog:
716                - None
717
718            Credits:
719                - None
720
721            References:
722                - None
723
724            Acknowledgments:
725                - None
726
727            Copyright:
728                - © 2023 Akhil Patel. All rights reserved.
729
730            License:
731                - MIT License
732
733            Version:
734                - 1.0
735
736            Last Updated:
737                - 2023-10-15
738
739            Status:
740                - Production
741
742            Issues:
743                - None
744
745            Pull Requests:
746                - None
747
748            Contributors:
749                - None
750
751            Tests:
752                - None
753
754            Documentation:
755                - None
756
757            Changelog:
758                - None
759
760            Credits:
761                - None
762
763            References:
764                - None
765
766            Acknowledgments:
767                - None
768
769            Copyright:
770                - © 2023 Akhil Patel. All rights reserved.
771
772            License:
773                - MIT License
774
775            Version:
776                - 1.0
777
778            Last Updated:
779                - 2023-10-15
780
781            Status:
782                - Production
783
784            Issues:
785                - None
786
787            Pull Requests:
788                - None
789
790            Contributors:
791                - None
792
793            Tests:
794                - None
795
796            Documentation:
797                - None
798
799            Changelog:
800                - None
801
802            Credits:
803                - None
804
805            References:
806                - None
807
808            Acknowledgments:
809                - None
810
811            Copyright:
812                - © 2023 Akhil Patel. All rights reserved.
813
814            License:
815                - MIT License
816
817            Version:
818                - 1.0
819
820            Last Updated:
821                - 2023-10-15
822
823            Status:
824                - Production
825
826            Issues:
827                - None
828
829            Pull Requests:
830                - None
831
832            Contributors:
833                - None
834
835            Tests:
836                - None
837
838            Documentation:
839                - None
840
841            Changelog:
842                - None
843
844            Credits:
845                - None
846
847            References:
848                - None
849
850            Acknowledgments:
851                - None
852
853            Copyright:
854                - © 2023 Akhil Patel. All rights reserved.
855
856            License:
857                - MIT License
858
859            Version:
860                - 1.0
861
862            Last Updated:
863                - 2023-10-15
864
865            Status:
866                - Production
867
868            Issues:
869                - None
870
871            Pull Requests:
872                - None
873
874            Contributors:
875                - None
876
877            Tests:
878                - None
879
880            Documentation:
881                - None
882
883            Changelog:
884                - None
885
886            Credits:
887                - None
888
889            References:
890                - None
891
892            Acknowledgments:
893                - None
894
895            Copyright:
896                - © 2023 Akhil Patel. All rights reserved.
897
898            License:
899                - MIT License
900
901            Version:
902                - 1.0
903
904            Last Updated:
905                - 2023-10-15
906
907            Status:
908                - Production
909
910            Issues:
911                - None
912
913            Pull Requests:
914                - None
915
916            Contributors:
917                - None
918
919            Tests:
920                - None
921
922            Documentation:
923                - None
924
925            Changelog:
926                - None
927
928            Credits:
929                - None
930
931            References:
932                - None
933
934            Acknowledgments:
935                - None
936
937            Copyright:
938                - © 2023 Akhil Patel. All rights reserved.
939
940            License:
941                - MIT License
942
943            Version:
944                - 1.0
945
946            Last Updated:
947                - 2023-10-15
948
949            Status:
950                - Production
951
952            Issues:
953                - None
954
955            Pull Requests:
956                - None
957
958            Contributors:
959                - None
960
961            Tests:
962                - None
963
964            Documentation:
965                - None
966
967            Changelog:
968                - None
969
970            Credits:
971                - None
972
973            References:
974                - None
975
976            Acknowledgments:
977                - None
978
979            Copyright:
980                - © 2023 Akhil Patel. All rights reserved.
981
982            License:
983                - MIT License
984
985            Version:
986                - 1.0
987
988            Last Updated:
989                - 2023-10-15
990
991            Status:
992                - Production
993
994            Issues:
995                - None
996
997            Pull Requests:
998                - None
999
1000            Contributors:
1001                - None
1002
1003            Tests:
1004                - None
1005
1006            Documentation:
1007                - None
1008
1009            Changelog:
1010                - None
1011
1012            Credits:
1013                - None
1014
1015            References:
1016                - None
1017
1018            Acknowledgments:
1019                - None
1020
1021            Copyright:
1022                - © 2023 Akhil Patel. All rights reserved.
1023
1024            License:
1025                - MIT License
1026
1027            Version:
1028                - 1.0
1029
1030            Last Updated:
1031                - 2023-10-15
1032
1033            Status:
1034                - Production
1035
1036            Issues:
1037                - None
1038
1039            Pull Requests:
1040                - None
1041
1042            Contributors:
1043                - None
1044
1045            Tests:
1046                - None
1047
1048            Documentation:
1049                - None
1050
1051            Changelog:
1052                - None
1053
1054            Credits:
1055                - None
1056
1057            References:
1058                - None
1059
1060            Acknowledgments:
1061                - None
1062
1063            Copyright:
1064                - © 2023 Akhil Patel. All rights reserved.
1065
1066            License:
1067                - MIT License
1068
1069            Version:
1070                - 1.0
1071
1072            Last Updated:
1073                - 2023-10-15
1074
1075            Status:
1076                - Production
1077
1078            Issues:
1079                - None
1080
1081            Pull Requests:
1082                - None
1083
1084            Contributors:
1085                - None
1086
1087            Tests:
1088                - None
1089
1090            Documentation:
1091                - None
1092
1093            Changelog:
1094                - None
1095
1096            Credits:
1097                - None
1098
1099            References:
1100                - None
1101
1102            Acknowledgments:
1103                - None
1104
1105            Copyright:
1106                - © 2023 Akhil Patel. All rights reserved.
1107
1108            License:
1109                - MIT License
1110
1111            Version:
1112                - 1.0
1113
1114            Last Updated:
1115                - 2023-10-15
1116
1117            Status:
1118                - Production
1119
1120            Issues:
1121                - None
1122
1123            Pull Requests:
1124                - None
1125
1126            Contributors:
1127                - None
1128
1129            Tests:
1130                - None
1131
1132            Documentation:
1133                - None
1134
1135            Changelog:
1136                - None
1137
1138            Credits:
1139                - None
1140
1141            References:
1142                - None
1143
1144            Acknowledgments:
1145                - None
1146
1147            Copyright:
1148                - © 2023 Akhil Patel. All rights reserved.
1149
1150            License:
1151                - MIT License
1152
1153            Version:
1154                - 1.0
1155
1156            Last Updated:
1157                - 2023-10-15
1158
1159            Status:
1160                - Production
1161
1162            Issues:
1163                - None
1164
1165            Pull Requests:
1166                - None
1167
1168            Contributors:
1169                - None
1170
1171            Tests:
1172                - None
1173
1174            Documentation:
1175                - None
1176
1177            Changelog:
1178                - None
1179
1180            Credits:
1181                - None
1182
1183            References:
1184                - None
1185
1186            Acknowledgments:
1187                - None
1188
1189            Copyright:
1190                - © 2023 Akhil Patel. All rights reserved.
1191
1192            License:
1193                - MIT License
1194
1195            Version:
1196                - 1.0
1197
1198            Last Updated:
1199                - 2023-10-15
1200
1201            Status:
1202                - Production
1203
1204            Issues:
1205                - None
1206
1207            Pull Requests:
1208                - None
1209
1210            Contributors:
1211                - None
1212
1213            Tests:
1214                - None
1215
1216            Documentation:
1217                - None
1218
1219            Changelog:
1220                - None
1221
1222            Credits:
1223                - None
1224
1225            References:
1226                - None
1227
1228            Acknowledgments:
1229                - None
1230
1231            Copyright:
1232                - © 2023 Akhil Patel. All rights reserved.
1233
1234            License:
1235                - MIT License
1236
1237            Version:
1238                - 1.0
1239
1240            Last Updated:
1241                - 2023-10-15
1242
1243            Status:
1244                - Production
1245
1246            Issues:
1247                - None
1248
1249            Pull Requests:
1250                - None
1251
1252            Contributors:
1253                - None
1254
1255            Tests:
1256                - None
1257
1258            Documentation:
1259                - None
1260
1261            Changelog:
1262                - None
1263
1264            Credits:
1265                - None
1266
1267            References:
1268                - None
1269
1270            Acknowledgments:
1271                - None
1272
1273            Copyright:
1274                - © 2023 Akhil Patel. All rights reserved.
1275
1276            License:
1277                - MIT License
1278
1279            Version:
1280                - 1.0
1281
1282            Last Updated:
1283                - 2023-10-15
1284
1285            Status:
1286                - Production
1287
1288            Issues:
1289                - None
1290
1291            Pull Requests:
1292                - None
1293
1294            Contributors:
1295                - None
1296
1297            Tests:
1298                - None
1299
1300            Documentation:
1301                - None
1302
1303            Changelog:
1304                - None
1305
1306            Credits:
1307                - None
1308
1309            References:
1310                - None
1311
1312            Acknowledgments:
1313                - None
1314
1315            Copyright:
1316                - © 2023 Akhil Patel. All rights reserved.
1317
1318            License:
1319                - MIT License
1320
1321            Version:
1322                - 1.0
1323
1324            Last Updated:
1325                - 2023-10-15
1326
1327            Status:
1328                - Production
1329
1330            Issues:
1331                - None
1332
1333            Pull Requests:
1334                - None
1335
1336            Contributors:
1337                - None
1338
1339            Tests:
1340                - None
1341
1342            Documentation:
1343                - None
1344
1345            Changelog:
1346                - None
1347
1348            Credits:
1349                - None
1350
1351            References:
1352                - None
1353
1354            Acknowledgments:
1355                - None
1356
1357            Copyright:
1358                - © 2023 Akhil Patel. All rights reserved.
1359
1360            License:
1361                - MIT License
1362
1363            Version:
1364                - 1.0
1365
1366            Last Updated:
1367                - 2023-10-15
1368
1369            Status:
1370                - Production
1371
1372            Issues:
1373                - None
1374
1375            Pull Requests:
1376                - None
1377
1378            Contributors:
1379                - None
1380
1381            Tests:
1382                - None
1383
1384            Documentation:
1385                - None
1386
1387            Changelog:
1388                - None
1389
1390            Credits:
1391                - None
1392
1393            References:
1394                - None
1395
1396            Acknowledgments:
1397                - None
1398
1399            Copyright:
1400                - © 2023 Akhil Patel. All rights reserved.
1401
1402            License:
1403                - MIT License
1404
1405            Version:
1406                - 1.0
1407
1408            Last Updated:
1409                - 2023-10-15
1410
1411            Status:
1412                - Production
1413
1414            Issues:
1415                - None
1416
1417            Pull Requests:
1418                - None
1419
1420            Contributors:
1421                - None
1422
1423            Tests:
1424                - None
1425
1426            Documentation:
1427                - None
1428
1429            Changelog:
1430                - None
1431
1432            Credits:
1433                - None
1434
1435            References:
1436                - None
1437
1438            Acknowledgments:
1439                - None
1440
1441            Copyright:
1442                - © 2023 Akhil Patel. All rights reserved.
1443
1444            License:
1445                - MIT License
1446
1447            Version:
1448                - 1.0
1449
1450            Last Updated:
1451                - 2023-10-15
1452
1453            Status:
1454                - Production
1455
1456            Issues:
1457                - None
1458
1459            Pull Requests:
1460                - None
1461
1462            Contributors:
1463                - None
1464
1465            Tests:
1466                - None
1467
1468            Documentation:
1469                - None
1470
1471            Changelog:
1472                - None
1473
1474            Credits:
1475                - None
1476
1477            References:
1478                - None
1479
1480            Acknowledgments:
1481                - None
1482
1483            Copyright:
1484                - © 2023 Akhil Patel. All rights reserved.
1485
1486            License:
1487                - MIT License
1488
1489            Version:
1490                - 1.0
1491
1492            Last Updated:
1493                - 2023-10-15
1494
1495            Status:
1496                - Production
1497
1498            Issues:
1499                - None
1500
1501            Pull Requests:
1502                - None
1503
1504            Contributors:
1505                - None
1506
1507            Tests:
1508                - None
1509
1510            Documentation:
1511                - None
1512
1513            Changelog:
1514                - None
1515
1516            Credits:
1517                - None
1518
1519            References:
1520                - None
1521
1522            Acknowledgments:
1523                - None
1524
1525            Copyright:
1526                - © 2023 Akhil Patel. All rights reserved.
1527
1528            License:
1529                - MIT License
1530
1531            Version:
1532                - 1.0
1533
1534            Last Updated:
1535                - 2023-10-15
1536
1537            Status:
1538               
```

Base case: if node is None, create and return a new Node.

Recursive case: go left if value <node.data, right if value >node.data, then return node to maintain links.

inorder_traversal(): returns a list of values; uses _inorder_recursive(node, result):

Base case: node is None → return.

Recursive case: traverse left, visit node (append data), traverse right.

Comments in both recursive helpers explain base/recursive cases clearly.

Task Description -5 – (Hash Table Implementation)

- Task: Create a hash table using AI with collision handling

Expected Output:

- Hash table supporting insert, search, and delete using chaining or open

The screenshot shows the Microsoft DevOps AI-assisted coding interface. On the left, there are tabs for task1.py, task2.py, task3.py, task4.py, and task5.py. The task5.py tab is active, displaying Python code for a HashTable class using chaining for collision handling. The code includes comments explaining the implementation, such as the use of separate chaining and the creation of a fixed-size Python list for each bucket. It defines methods for insertion, searching, and deletion. On the right, there is a sidebar titled "DESIGNING A STACK DATA STRUCTURE IN PYTHON" with the following requirements:

- Act as a Python programming expert.
- Help me implement a Hash Table in Python with collision handling using chaining.
- Requirements:
 - Create a HashTable class
 - Implement insert(), search(), and delete() methods
 - Use chaining (list of lists)
 - Add detailed comments
 - Provide sample test cases with output
 - Explain how collision handling works
 - Mention average and worst-case time complexity

Below the requirements, there are several status indicators and notes:

- ✓ Planning hash table implementation
- I'll open task5.py and then implement a chained HashTable class with insert/search/delete, comments, example tests, and complexity notes.
- ✓ Reviewed and updated task5.py
- I've implemented your chained hash table and example tests in AI-ASSISTED_CODING/task5.py.
- What's implemented:
 - HashTable class with:
 - __init__(capacity=10) creates capacity buckets, each an empty list.
 - _hash(key): uses Python hash(key) % capacity to pick a bucket index.

The bottom of the interface shows the Python environment: Python 3.13 (64-bit), Go Live, and Prettier.

```
task5.py
def __init__(self, capacity=10):
    """Initialize the hash table with a given number of buckets.

    Args:
        capacity: Number of buckets (slots) in the hash table.

    Time Complexity: O(capacity) to allocate buckets (once at start).
    """
    self.capacity = capacity
    # Create a list of empty lists (buckets) for chaining.
    self.table = [[] for _ in range(capacity)]

def _hash(self, key):
    """Compute hash index for a given key.

    We use Python's built-in hash() and then take modulo capacity
    to map the hash value into the bucket range [0, capacity-1].
    """
    return hash(key) % self.capacity

def insert(self, key, value):
    """Insert or update a key-value pair in the hash table.

    Steps:
    1. Compute the bucket index using the hash function.
    2. Look through the bucket to see if the key already exists.
       - If found, update its value.
       - If not found, append a new (key, value) pair to the bucket.
    """
    Average Time Complexity: O(1)
    Worst-case Time Complexity: O(n) when many keys collide in one bucket.
    """
    index = self._hash(key)
    bucket = self.table[index]

    # Check if key already exists in this bucket; if so, update it.
    for k, v in enumerate(bucket):
        if k == key:
            bucket[k] = (key, value)
            return

    # Append the new key-value pair to the bucket.
    bucket.append((key, value))

def search(self, key):
    """Search for a key in the hash table and return its value.

    Args:
        key: The key to search for.

    Returns:
        The value associated with the key, or None if the key is not found.
    """
    index = self._hash(key)
    bucket = self.table[index]

    for k, v in bucket:
        if k == key:
            return v

    return None

def delete(self, key):
    """Delete a key from the hash table.

    Args:
        key: The key to delete.

    Returns:
        The value associated with the key, or None if the key is not found.
    """
    index = self._hash(key)
    bucket = self.table[index]

    for k, v in enumerate(bucket):
        if k == key:
            del bucket[k]
            return v

    return None
```

The screenshot shows a code editor with multiple tabs open. The active tab is `task5.py`, which contains Python code for a HashTable class. The code includes methods for inserting key-value pairs, searching for keys, and deleting keys. It also includes a `def example_usage()` block and a `Collision handling explanation` section. The Chat interface on the right shows a conversation about designing a stack data structure in Python, with requirements listed and a note about planning the hash table implementation.

```

AI-ASSISTED_CODING > task5.py ...
113     def _example_usage():
114         """Run sample test cases to demonstrate chaining and basic operations.
115
116         Collision handling explanation:
117         - Different keys can produce the same hash index (after modulo).
118         - All such keys are stored in the same bucket list.
119         - When we search or delete, we restrict our scan to that bucket only.
120
121     # Small capacity to increase chances of collision for demonstration
122     ht = HashTable(capacity=5)
123
124     print("Inserting key-value pairs:")
125     ht.insert("apple", 1)
126     ht.insert("banana", 2)
127     ht.insert("grape", 3)
128     ht.insert("grape", 3)
129
130     # Insert keys that are likely to collide by design (Integers)
131     ht.insert(10, "ten")
132     ht.insert(15, "fifteen") # 10 and 15 collide when capacity=5 (10% == 15%)
133
134     print("Hash table internal state (buckets):")
135     print(ht)
136
137     print("Searching for keys:")
138     print(search("apple")) -> ht.search("apple")
139     print(search("banana")) -> ht.search("banana")
140     print(search("grape")) -> ht.search("grape")
141     print(search("grape")) -> ht.search("grape")
142     print(search(10)) -> ht.search(10)
143     print(search("unknown")) -> ht.search("unknown")
144     print(search("unknown")) -> ht.search("unknown") # Not present
145
146     print("Deleting keys:")
147     print(delete("banana")) -> ht.delete("banana")
148     print(delete(10) -> ht.delete(10))
149     print(delete("unknown")) -> ht.delete("unknown") # Not present
150
151     print("Hash table after deletions:")
152     print(ht)
153
154     # Expected sample output (exact bucket order may differ due to hash values):
155     # Inserting key-value pairs:
156     # Hash table internal state (buckets):
157     # HashTable([(('apple', 1), [(10, 'ten'), (15, 'fifteen')], []),
158     # Searching for keys:
159     # search('apple') -> 1
160     # search('banana') -> 2
161     # search('grape') -> 3
162     # search(10) -> ten
163     # search(15) -> fifteen
164     # search('unknown') -> None
165
166     # Deleting keys:
167     # delete('banana') -> True
168     # delete(10) -> True
169     # delete('unknown') -> False

```

Inserting values into BST: [50, 30, 70, 20, 40, 60, 80]

In-order traversal result:
[20, 30, 40, 50, 60, 70, 80]

Expected sorted order:
[20, 30, 40, 50, 60, 70, 80]

In-order traversal result:
[20, 30, 40, 50, 60, 70, 80]

Expected sorted order:
[20, 30, 40, 50, 60, 70, 80]

C:\Users\akhil\OneDrive\Documents\Devops>python -u "c:\Users\akhil\OneDrive\Documents\Devops\AI-ASSISTED_CODING\task5.py"

Inserting key-value pairs:

Hash table internal state (buckets):

HashTable([(('apple', 1), [(10, 'ten'), (15, 'fifteen')], [], [], [(('grape', 3)], [(('banana', 2)]))

Searching for keys:
 search('apple') -> 1
 search('banana') -> 2
 search('grape') -> 3
 search(10) -> ten
 search(15) -> fifteen
 search('unknown') -> None

Deleting keys:
 delete('banana') -> True
 delete(10) -> True
 delete('unknown') -> False

- HashTable class with:

- `__init__(capacity=10)`: creates capacity buckets, each an empty list.
- `_hash(key)`: uses Python `hash(key) % capacity` to pick a bucket index.
- `insert(key, value)`: updates existing key or appends (key, value) into the bucket.
- `search(key)`: scans the bucket for key, returns the value or None.

- delete(key): removes (key, value) from the bucket, returns True/False.
- Chaining:
 - self.table is a list of lists (buckets).
 - Each bucket stores multiple (key, value) pairs that share the same index → this is collision handling by chaining.