

# Birla Institute of Science and Technology

## Artificial and Computational Intelligence

### PS1: Game Playing in Python

#### Abstract:

In this problem, we will be dealing with the **Catch-Up with Numbers** game problem of decision-making and considering the Minmax optimization technique used for selecting the best move.

<b>Task Environment:</b>	Game State, Players, Set of Natural Numbers, Randomization, Strategy, Win/Lose Conditions, Freedom of state, Game Rules, Freedom of moment, Utility function
<b>Algorithm:</b>	Minmax with Alph-beta pruning
<b>Fully vs Partially: Observable:</b>	Fully observable
<b>Single vs Multi-Agent:</b>	Multi-Agent
<b>Deterministic vs Stochastic:</b>	Deterministic
<b>Episodic vs Sequential:</b>	Sequential
<b>Static vs Dynamic:</b>	Dynamic
<b>Discrete vs Continuous:</b>	Discrete

#### PEAS Environment

#### Performance:

- Every player aims to get a higher score than the opponent or secure a tie if winning is not feasible. The efficiency of the Minimax algorithm with Alpha-Beta pruning is crucial for larger game states. It ensures that the algorithm performs well, especially as the size of the number set increases.

#### Environment:

G. Ankur Vatsa	2023aa05727
Nidasanametla Sree Sitamahalakshmi	2023aa05716
Prasenjit Samantha	2023aa05256
Randhawane Santosh	2023aa05828
Vedagiri Sai Krishna	2023aa05348

- The environment consists of the set of numbers  $\{1, 2, 3, \dots, n\}$  where  $n$  will be provided by the player as an input.
- Two player's game.
- It also includes the rules of the game, such as
  - players taking turns choosing numbers
  - numbers being removed from the set after being chosen
  - the condition for ending the game.

**Actuators:**

- It helps to prune unnecessary branches and evaluates different move possibilities for each player
- It is using the mechanism of Minimax with alpha-beta pruning for decision making.

**Sensors:**

- The sensors provide information about the state of the game, including the current set of available numbers, the sum chosen by the opponent in the previous turn, possibly the choices made by the opponent and player scores.

### **Task Environment**

**Players:** Two players take turns making choices from a set of numbers. Each player aims are to maximize their own score or minimize their loss while considering others players moves.

**Game State:** Current state of the game where each player has a current score representing the sum of the numbers they have chosen so far. The current score is important for determining the available move for the opponent player.

**Set of Natural Numbers:** The initial set consists of numbers starting from 1 to a predetermined number  $n$ .

**Game Rules:** Players alternate turns and each player selects one or more numbers from the remaining set, aiming to either exceed or match their opponent player score.

**Objective:** Players aim to maximize their score or minimize their loss or secure a tie if winning is not feasible.

**Randomization:** When making moves, each player must randomize their selection from the available number set, considering the chosen move equal or exceed their opponent's last total. Which ensures the fairness and unpredictability in the game.

**Strategy:** Each player's plan is to anticipate their opponent's moves to maximize their own score and adapt the changing game states based on opponent's move.

**Win/Lose Conditions:** Winning or losing the game depends on,

- Game ends when all numbers have been chosen.
- If one player has a higher sum than the other, that player wins. If not the game ends in a tie.

**Scoring System:** Implement a scoring system based on factors such as the number chosen by each player, the efficiency of operations etc.

---

### **Designing of Catch-Up with Numbers game using Minmax algorithms.**

---

The complete code has been segregated into three major classes,

- MinimaxStrategy
- Player
- CatchUpGame

#### **MinimaxStrategy**

Implements the Minimax algorithm with alpha-beta pruning. It helps to determine the best move for the current player by considering the opponent's potential moves and maximizing the player's score while minimizing the opponent's score.

#### **Key Objectives:**

- **MinimaxStrategy::move( )**

This method implements the core Minimax logic with alpha-beta pruning. It explores the game tree using recursion and evaluates different move likelihoods for each player.

- **MinimaxStrategy::\_alpha\_beta\_pruning( )**

This method helps to prune unnecessary branches in the game tree based on alpha and beta values, refining the proficiency of the search.

This search algorithm helps to reduce the number of nodes evaluated in the minimax algorithm.

- ❖ It explores the game tree in a depth-first manner
- ❖ At each level, the algorithm applies the minimax principle in order to evaluate the potential moves.
- ❖ This algorithm maintains two values, alpha and beta, which represent the minimum score the maximizing player is secure of and the maximum score the minimizing player is secure of, individually, along the process of path exploration.

- ❖ During the search process if, the algorithm finds a move that leads to a score worse than what the other player already has available, it recognizes that further exploration of this branch is unnecessary. Hence, it starts prunes this branch of the game tree.

➤ **MinimaxStrategy::\_minimax( )**

This private method recursively evaluates the game state from the perspective of both players, considering their potential moves and scores.

This algorithm is a decision-making algorithm.

- ❖ Game state and possible moves are represented as a tree
- ❖ Each level of the tree represents a player's turn, and each node represents a particular game state.
- ❖ Evaluation function assigns a score to each game state at the terminal level. This also helps to determine how good a particular game state is for the maximizing player and how bad it is for the minimizing player.
- ❖ This algorithm recursively explores the game tree, considering all possible moves that can be made by both players.
- ❖ As soon as the algorithm travels deeper into the tree, it propagates the scores back up to the root node. This process continues until the algorithm finds the optimal move based on the obtained scores.
- ❖ Once the entire tree evaluation is completed, the algorithm identifies the best move that leads to the best possible result for the maximizing player.

## Player

Depicts a player in the game, including their name, score, and chosen strategy.

### Key Objectives:

➤ **Player::choose\_strategy( )**

This method allows a player to select their approach (maximiser or minimizer) using user input.

➤ **Player::make\_move( )**

This method calls the player's chosen approach to determine the best move considering the current game state.

## CatchUpGame

Manages the overall game flow, including determining the starting player, checking the game state, identifying the winner, and handling turns for each player.

### Key Objectives:

#### ➤ **CatchUpGame::is\_game\_over( )**

This method helps to evaluate the game ending condition by confirming, there are no more available numbers or neither player can make a valid move.

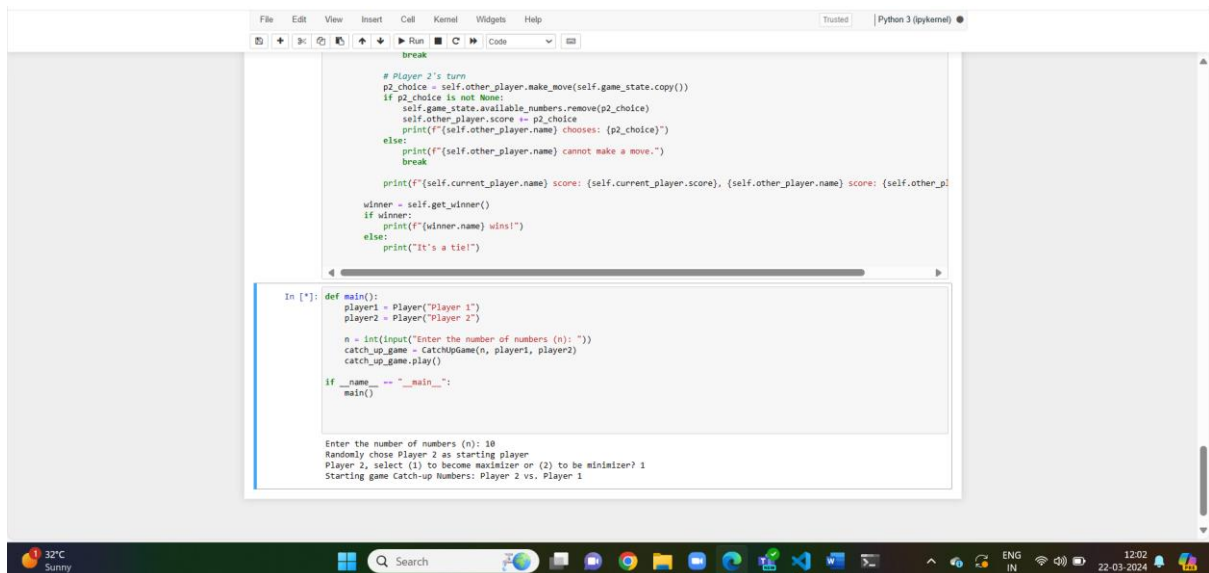
#### ➤ **CatchUpGame::get\_winner( )**

This method helps to governs the winner based on the player's scores.

#### ➤ **CatchUpGame::play( )**

This method handles the entire game loop, counting, exposing available numbers, prompting players for moves, scores pupation, and declaring the game output.

### Output:



```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)
Run Stop Restart Clear Code
# Player 2's turn
p2_choice = self.other_player.make_move(self.game_state.copy())
if p2_choice is not None:
    self.game_state.available_numbers.remove(p2_choice)
    self.other_player.score += p2_choice
    print(f"{self.other_player.name} chooses: {p2_choice}")
else:
    print(f"{self.other_player.name} cannot make a move.")
    break
print(f"({self.current_player.name} score: {self.current_player.score}, {self.other_player.name} score: {self.other_p

winner = self.get_winner()
if winner:
    print(f"{winner.name} wins!")
else:
    print("It's a tie!")

In [ ]: def main():
    player1 = Player("Player 1")
    player2 = Player("Player 2")
    n = int(input("Enter the number of numbers (n): "))
    catch_up_game = CatchUpGame(n, player1, player2)
    catch_up_game.play()

if __name__ == "__main__":
    main()

Enter the number of numbers (n): 10
Randomly chose Player 2 as starting player
Player 2, select (1) to become maximizer or (2) to be minimizer? 1
Starting game Catch-up Numbers: Player 2 vs. Player 1
```

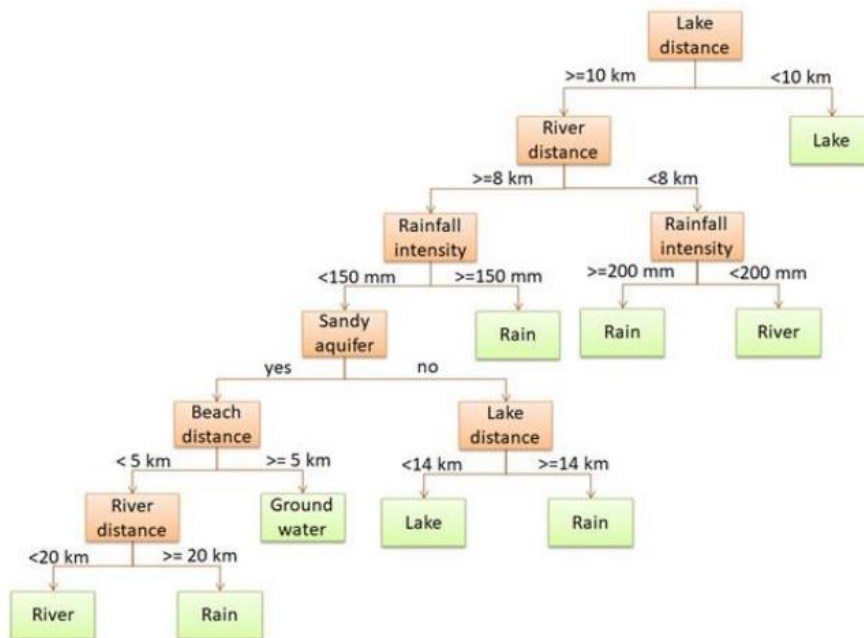
## PS2: Logic design and inferencing in Prolog.

### Abstract:

A decision-making process to determine the source of water based on various parameters like distance from a lake or river, rainfall intensity, and others.

In this problem, we are determining the best water source for a community using logic programming considering various parameters like distance from a lake or river, rainfall intensity, and others.

Decision tree provided for creation of Prolog rules:



## Designing logic and prolog inferencing

### Logic implementation:

#### ➤ Prolog Rules:

```
1 predict_water_source(Distance_from_Lake, Distance_from_River, Rainfall_intensity, Sandy_aquifer, Distance_from_Beach, WaterSource) :-
2 (
3     Distance_from_Lake < 10,
4     WaterSource = lake
5 );
6 (
7     Distance_from_Lake >= 10,
8     Distance_from_River < 8,
9     Rainfall_intensity < 200,
10    WaterSource = river
11 );
12 (
13    Distance_from_Lake >= 10,
14    Distance_from_River < 8,
15    Rainfall_intensity >= 200,
16    WaterSource = rain
17 );
18 (
19    Distance_from_Lake >= 10,
20    Distance_from_River >= 8,
21    Rainfall_intensity >= 150,
22    WaterSource = rain
23 );
24 (
25    Distance_from_Lake < 14,
26    Distance_from_River >= 8,
27    Rainfall_intensity < 150,
28    Sandy_aquifer == no,
29    WaterSource = lake
30 );
31 (
32    Distance_from_Lake >= 14,
33    Distance_from_River >= 8,
34    Rainfall_intensity < 150,
35    Sandy_aquifer == no,
36    WaterSource = rain
37 );
38 (
39    Distance_from_Lake >= 10,
40    Distance_from_River >= 8,
41    Rainfall_intensity < 150,
42    Sandy_aquifer == yes,
43    Distance_from_Beach >= 5,
44    WaterSource = groundwater
45 );
46 (
47    Distance_from_Lake >= 10,
48    Distance_from_River >= 20,
49    Rainfall_intensity < 150,
50    Sandy_aquifer == yes,
51    Distance_from_Beach < 5,
52    WaterSource = rain
53 );
54 (
55    Distance_from_Lake >= 10,
56    Distance_from_River < 20,
57    Rainfall_intensity < 150,
58    Sandy_aquifer == yes,
59    Distance_from_Beach < 5,
60    WaterSource = river
61 );
62 % default to groundwater
63 WaterSource = groundwater.
64
```

- **Get\_user\_input:**  
Deals with user provided input at runtime

```
65 get_user_input :-  
66     write('Distance from Lake (km): '),  
67     read(Distance_from_Lake),  
68     write('Distance from River (km): '),  
69     read(Distance_from_River),  
70     write('Enter Rainfall intensity (mm/month): '),  
71     read(Rainfall_intensity),  
72     write('Is the Sandy aquifer Sandy? (yes/no): '),  
73     read(Sandy_aquifer),  
74     write('Distance from Beach (km): '),  
75     read(Distance_from_Beach),  
76     predict_water_source(Distance_from_Lake, Distance_from_River, Rainfall_intensity, Sandy_aquifer, Distance_from_Beach, WaterSource),  
77     write('Recommended Water Source: '),  
78     write(WaterSource).  
79
```

- **Start:**  
Act as a main function or initiate rule execution program

*% Start the program*

```
start :-  
    write('Welcome to the Water Source Prediction System!'), nl,  
    get_user_input,  
    nl,  
    write('Thank you!').
```

**Example:** if there is a case where a location has features: 120 mm/month rainfall, sandy aquifer, 10 km away from the perennial river, 20 km away from the lake, and 2 km away from the beach, could you decide which water resource suitable for the community to take from, rainfall, river water, lake, or groundwater?

**Features identified:**

Rainfall → 120 mm/month

Sandy aquifer → yes

River → 10 km

Lake → 20 km

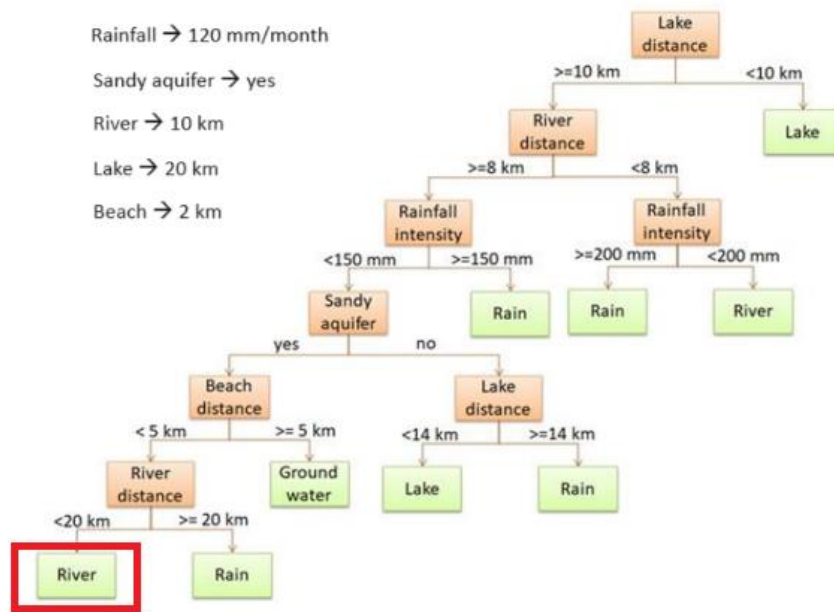
Beach → 2 km

---

G. Ankur Vatsa	2023aa05727
Nidasanametla Sree Sitamahalakshmi	2023aa05716
Prasenjit Samantha	2023aa05256
Randhawane Santosh	2023aa05828
Vedagiri Sai Krishna	2023aa05348



**Output:** River is the expected output considering the given decision tree.



**User Input/Output:**

```

1 // Water Source Prediction System
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 int main()
6 {
7     WaterSource = rain;
8     // User Input
9     // Distance from Lake (km)
10    Distance_from_Lake >= 10;
11    // Distance from River (km)
12    Distance_from_River < 20;
13    // Rainfall intensity (mm/month)
14    Rainfall_intensity < 150;
15    // Sandy aquifer = yes;
16    Sandy_aquifer == yes;
17    // Distance from Beach (km)
18    Distance_from_Beach < 5;
19    WaterSource = river;
20    // Default to groundwater
21    WaterSource = groundwater;
22    // User Input
23    // Distance from Lake (km)
24    read(Distance_from_Lake);
25    // Distance from River (km)
26    read(Distance_from_River);
27    // Enter Rainfall intensity (mm/month)
28    write("Enter Rainfall intensity (mm/month): ");
29    read(Rainfall_intensity);
30    // Is the Sandy aquifer Sandy? (yes/no)
31    write("Is the Sandy aquifer Sandy? (yes/no): ");
32    read(Sandy_aquifer);
33    // Distance from Beach (km)
34    write("Distance from Beach (km): ");
35    read(Distance_from_Beach);
36    predict_water_source(Distance_from_Lake, Distance_from_River, Rainfall_intensity, Sandy_aquifer, Distance_from_Beach);
37    write("Recommended Water Source: ");
38    write(WaterSource);
39    // End of program
40    return 0;
41 }
42
43 // Start the program
44 int start()
45 {
46     write("Welcome to the Water Source Prediction System!\n");
47     get_user_input();
48     return 0;
49 }
50
51 // End of program
52
53 // User Input/Output
54
55 start();
56
57 // User Input/Output
58
59 start();
60
61 // User Input/Output
62
63 start();
64
65 // User Input/Output
66
67 start();
68
69 // User Input/Output
70
71 start();
72
73 // User Input/Output
74
75 start();
76
77 // User Input/Output
78
79 start();
80
81 // User Input/Output
82
83 start();
84
85 // User Input/Output
86
87 start();
88
89 // User Input/Output
90
91 start();
92
93 // User Input/Output
94
95 start();
96
97 // User Input/Output
98
99 start();
100
101 // User Input/Output
102
103 start();
104
105 // User Input/Output
106
107 start();
108
109 // User Input/Output
110
111 start();
112
113 // User Input/Output
114
115 start();
116
117 // User Input/Output
118
119 start();
120
121 // User Input/Output
122
123 start();
124
125 // User Input/Output
126
127 start();
128
129 // User Input/Output
130
131 start();
132
133 // User Input/Output
134
135 start();
136
137 // User Input/Output
138
139 start();
140
141 // User Input/Output
142
143 start();
144
145 // User Input/Output
146
147 start();
148
149 // User Input/Output
150
151 start();
152
153 // User Input/Output
154
155 start();
156
157 // User Input/Output
158
159 start();
160
161 // User Input/Output
162
163 start();
164
165 // User Input/Output
166
167 start();
168
169 // User Input/Output
170
171 start();
172
173 // User Input/Output
174
175 start();
176
177 // User Input/Output
178
179 start();
180
181 // User Input/Output
182
183 start();
184
185 // User Input/Output
186
187 start();
188
189 // User Input/Output
190
191 start();
192
193 // User Input/Output
194
195 start();
196
197 // User Input/Output
198
199 start();
200
201 // User Input/Output
202
203 start();
204
205 // User Input/Output
206
207 start();
208
209 // User Input/Output
210
211 start();
212
213 // User Input/Output
214
215 start();
216
217 // User Input/Output
218
219 start();
220
221 // User Input/Output
222
223 start();
224
225 // User Input/Output
226
227 start();
228
229 // User Input/Output
230
231 start();
232
233 // User Input/Output
234
235 start();
236
237 // User Input/Output
238
239 start();
240
241 // User Input/Output
242
243 start();
244
245 // User Input/Output
246
247 start();
248
249 // User Input/Output
250
251 start();
252
253 // User Input/Output
254
255 start();
256
257 // User Input/Output
258
259 start();
260
261 // User Input/Output
262
263 start();
264
265 // User Input/Output
266
267 start();
268
269 // User Input/Output
270
271 start();
272
273 // User Input/Output
274
275 start();
276
277 // User Input/Output
278
279 start();
280
281 // User Input/Output
282
283 start();
284
285 // User Input/Output
286
287 start();
288
289 // User Input/Output
290
291 start();
292
293 // User Input/Output
294
295 start();
296
297 // User Input/Output
298
299 start();
300
301 // User Input/Output
302
303 start();
304
305 // User Input/Output
306
307 start();
308
309 // User Input/Output
310
311 start();
312
313 // User Input/Output
314
315 start();
316
317 // User Input/Output
318
319 start();
320
321 // User Input/Output
322
323 start();
324
325 // User Input/Output
326
327 start();
328
329 // User Input/Output
330
331 start();
332
333 // User Input/Output
334
335 start();
336
337 // User Input/Output
338
339 start();
340
341 // User Input/Output
342
343 start();
344
345 // User Input/Output
346
347 start();
348
349 // User Input/Output
350
351 start();
352
353 // User Input/Output
354
355 start();
356
357 // User Input/Output
358
359 start();
360
361 // User Input/Output
362
363 start();
364
365 // User Input/Output
366
367 start();
368
369 // User Input/Output
370
371 start();
372
373 // User Input/Output
374
375 start();
376
377 // User Input/Output
378
379 start();
380
381 // User Input/Output
382
383 start();
384
385 // User Input/Output
386
387 start();
388
389 // User Input/Output
390
391 start();
392
393 // User Input/Output
394
395 start();
396
397 // User Input/Output
398
399 start();
400
401 // User Input/Output
402
403 start();
404
405 // User Input/Output
406
407 start();
408
409 // User Input/Output
410
411 start();
412
413 // User Input/Output
414
415 start();
416
417 // User Input/Output
418
419 start();
420
421 // User Input/Output
422
423 start();
424
425 // User Input/Output
426
427 start();
428
429 // User Input/Output
430
431 start();
432
433 // User Input/Output
434
435 start();
436
437 // User Input/Output
438
439 start();
440
441 // User Input/Output
442
443 start();
444
445 // User Input/Output
446
447 start();
448
449 // User Input/Output
450
451 start();
452
453 // User Input/Output
454
455 start();
456
457 // User Input/Output
458
459 start();
460
461 // User Input/Output
462
463 start();
464
465 // User Input/Output
466
467 start();
468
469 // User Input/Output
470
471 start();
472
473 // User Input/Output
474
475 start();
476
477 // User Input/Output
478
479 start();
480
481 // User Input/Output
482
483 start();
484
485 // User Input/Output
486
487 start();
488
489 // User Input/Output
490
491 start();
492
493 // User Input/Output
494
495 start();
496
497 // User Input/Output
498
499 start();
500
501 // User Input/Output
502
503 start();
504
505 // User Input/Output
506
507 start();
508
509 // User Input/Output
510
511 start();
512
513 // User Input/Output
514
515 start();
516
517 // User Input/Output
518
519 start();
520
521 // User Input/Output
522
523 start();
524
525 // User Input/Output
526
527 start();
528
529 // User Input/Output
530
531 start();
532
533 // User Input/Output
534
535 start();
536
537 // User Input/Output
538
539 start();
540
541 // User Input/Output
542
543 start();
544
545 // User Input/Output
546
547 start();
548
549 // User Input/Output
550
551 start();
552
553 // User Input/Output
554
555 start();
556
557 // User Input/Output
558
559 start();
560
561 // User Input/Output
562
563 start();
564
565 // User Input/Output
566
567 start();
568
569 // User Input/Output
570
571 start();
572
573 // User Input/Output
574
575 start();
576
577 // User Input/Output
578
579 start();
580
581 // User Input/Output
582
583 start();
584
585 // User Input/Output
586
587 start();
588
589 // User Input/Output
590
591 start();
592
593 // User Input/Output
594
595 start();
596
597 // User Input/Output
598
599 start();
600
601 // User Input/Output
602
603 start();
604
605 // User Input/Output
606
607 start();
608
609 // User Input/Output
610
611 start();
612
613 // User Input/Output
614
615 start();
616
617 // User Input/Output
618
619 start();
620
621 // User Input/Output
622
623 start();
624
625 // User Input/Output
626
627 start();
628
629 // User Input/Output
630
631 start();
632
633 // User Input/Output
634
635 start();
636
637 // User Input/Output
638
639 start();
640
641 // User Input/Output
642
643 start();
644
645 // User Input/Output
646
647 start();
648
649 // User Input/Output
650
651 start();
652
653 // User Input/Output
654
655 start();
656
657 // User Input/Output
658
659 start();
660
661 // User Input/Output
662
663 start();
664
665 // User Input/Output
666
667 start();
668
669 // User Input/Output
670
671 start();
672
673 // User Input/Output
674
675 start();
676
677 // User Input/Output
678
679 start();
680
681 // User Input/Output
682
683 start();
684
685 // User Input/Output
686
687 start();
688
689 // User Input/Output
690
691 start();
692
693 // User Input/Output
694
695 start();
696
697 // User Input/Output
698
699 start();
700
701 // User Input/Output
702
703 start();
704
705 // User Input/Output
706
707 start();
708
709 // User Input/Output
710
711 start();
712
713 // User Input/Output
714
715 start();
716
717 // User Input/Output
718
719 start();
720
721 // User Input/Output
722
723 start();
724
725 // User Input/Output
726
727 start();
728
729 // User Input/Output
730
731 start();
732
733 // User Input/Output
734
735 start();
736
737 // User Input/Output
738
739 start();
740
741 // User Input/Output
742
743 start();
744
745 // User Input/Output
746
747 start();
748
749 // User Input/Output
750
751 start();
752
753 // User Input/Output
754
755 start();
756
757 // User Input/Output
758
759 start();
760
761 // User Input/Output
762
763 start();
764
765 // User Input/Output
766
767 start();
768
769 // User Input/Output
770
771 start();
772
773 // User Input/Output
774
775 start();
776
777 // User Input/Output
778
779 start();
780
781 // User Input/Output
782
783 start();
784
785 // User Input/Output
786
787 start();
788
789 // User Input/Output
790
791 start();
792
793 // User Input/Output
794
795 start();
796
797 // User Input/Output
798
799 start();
800
801 // User Input/Output
802
803 start();
804
805 // User Input/Output
806
807 start();
808
809 // User Input/Output
810
811 start();
812
813 // User Input/Output
814
815 start();
816
817 // User Input/Output
818
819 start();
820
821 // User Input/Output
822
823 start();
824
825 // User Input/Output
826
827 start();
828
829 // User Input/Output
830
831 start();
832
833 // User Input/Output
834
835 start();
836
837 // User Input/Output
838
839 start();
840
841 // User Input/Output
842
843 start();
844
845 // User Input/Output
846
847 start();
848
849 // User Input/Output
850
851 start();
852
853 // User Input/Output
854
855 start();
856
857 // User Input/Output
858
859 start();
860
861 // User Input/Output
862
863 start();
864
865 // User Input/Output
866
867 start();
868
869 // User Input/Output
870
871 start();
872
873 // User Input/Output
874
875 start();
876
877 // User Input/Output
878
879 start();
880
881 // User Input/Output
882
883 start();
884
885 // User Input/Output
886
887 start();
888
889 // User Input/Output
890
891 start();
892
893 // User Input/Output
894
895 start();
896
897 // User Input/Output
898
899 start();
900
901 // User Input/Output
902
903 start();
904
905 // User Input/Output
906
907 start();
908
909 // User Input/Output
910
911 start();
912
913 // User Input/Output
914
915 start();
916
917 // User Input/Output
918
919 start();
920
921 // User Input/Output
922
923 start();
924
925 // User Input/Output
926
927 start();
928
929 // User Input/Output
930
931 start();
932
933 // User Input/Output
934
935 start();
936
937 // User Input/Output
938
939 start();
940
941 // User Input/Output
942
943 start();
944
945 // User Input/Output
946
947 start();
948
949 // User Input/Output
950
951 start();
952
953 // User Input/Output
954
955 start();
956
957 // User Input/Output
958
959 start();
960
961 // User Input/Output
962
963 start();
964
965 // User Input/Output
966
967 start();
968
969 // User Input/Output
970
971 start();
972
973 // User Input/Output
974
975 start();
976
977 // User Input/Output
978
979 start();
980
981 // User Input/Output
982
983 start();
984
985 // User Input/Output
986
987 start();
988
989 // User Input/Output
990
991 start();
992
993 // User Input/Output
994
995 start();
996
997 // User Input/Output
998
999 start();
1000

```

**GIT Reference:** [mtech/semester\\_1/03\\_assignments/aci/ASSIGNMENT\\_2](https://github.com/vatsaa/mtech_semester_1_03_assignments/tree/main/aci/ASSIGNMENT_2) at main · vatsaa/mtech · GitHub

G. Ankur Vatsa  
Nidasanametla Sree Sitamahalakshmi  
Prasenjit Samantha  
Randhawane Santosh  
Vedagiri Sai Krishna

2023aa05727  
2023aa05716  
2023aa05256  
2023aa05828  
2023aa05348