Exercises:

Exercise 1:

A bear starts at point P, walks 1 mile south, 1 mile east, and then 1 mile north, returning to point P. What is the color of the bear?

Understanding of the Problem

- The only place on Earth where this scenario is possible is near the North Pole.
- Since polar bears are found in the Arctic, the bear must be white.

Algorithm

1. Initialize the starting point P.
2. Move 1 mile south to reach point Q.
3. Move 1 mile east, following the circular latitude line at point Q.
4. Move 1 mile north to return to the original starting point P.
5. Check if the bear is at the North Pole (since only there does this movement return to the start).
6. If the bear is at the North Pole, conclude that it is a Polar Bear (White in color).
7. End.

Answer:

Since the only type of bear found at the North Pole is a Polar Bear.

Answer : The bear is white.

Exercise 2:

Towns A and B are 3 km apart. A school needs to be placed such that total student travel is minimized. Town A has 100 students, and town B has 50 students. Where should the school be located?

Understanding of the Problem

- This is a weighted average problem (i.e) weighted by the number of students.
- If we put the school closer to Town A, students from B will have to travel more.
  If we put the school closer to Town B, students from A will have to travel more.
  So, we should balance the travel distance based on the number of students.
- The school should be closer to Town A because it has more students.

Algorithm

Step 1: Initialize

- Let D = 3 km (Distance between towns A and B).
- Let A = 100 (Students in town A).

- Let B = 50 (Students in town B).
- Let X be the distance from town A to the school (to be determined).

Step 2: Define Total Travel Distance Formula

- Total travel distance = (A × X) + (B × (D - X))
- Substitute values: (100X) + (50(3−X))

Step 3: Minimize Total Travel Distance

- Expand the equation: 100X + 150 − 50X
- Simplify: 50X + 150
- To minimize, 100X = 50(3 − X)

  100X + 50X = 150

  150X = 150

  X = 1

Step 4: Output the Result

- The optimal school location is **1 km from town A** (or)
- **2 km from town B**.

Step 5: End

Exercise 3:

A traveller has a **6-link silver chain** and must pay **one link per day** at a hotel, but only **one link can be broken**. How should the traveler cut the chain to make daily payments?

Understanding of the Problem

You have three jugs:

- Jug A (8 cups) - Initially filled with 8 cups of water.
- Jug B (5 cups) - Empty.
- Jug C (3 cups) - Empty.
- Split the water equally between Jug A and Jug B, so both have 4 cups each.

Algorithm

Step 1: Initialize the Problem

- Define the jugs and their capacities:
  - Jug A (Capacity = 8 cups)

- Jug B (Capacity = 5 cups)
- Jug C (Capacity = 3 cups)

- Set the initial state as (8,0,0).
- Define the target state as (4,4,0).
- Create a queue to track the states.

Step 2: Defining the Operations

- Transfer water between jugs while obeying their capacities:
  1. Pour from A → B (until B is full or A is empty).
  2. Pour from A → C (until C is full or A is empty).
  3. Pour from B → A (until A is full or B is empty).
  4. Pour from B → C (until C is full or B is empty).
  5. Pour from C → A (until A is full or C is empty).
  6. Pour from C → B (until B is full or C is empty).

Step 3: Implementation

1. Enqueue the initial state *(8,0,0)* into the queue.
2. Repeat until the queue is empty or the goal is reached:
   - Dequeue the first state from the queue.
   - Check if it matches the target state *(4,4,0)*. If yes, stop.

Step 4: End Condition

- If we reach (4,4,0), print the steps taken.
- If the queue is exhausted without finding the solution, print "No solution found".

Step-by-Step Approach

- Pour water from A to C (C fills up with 3 cups).
  (A=5, B=0, C=3)
- Pour water from C to B (B gets 3 cups from C).
  (A=5, B=3, C=0)
- Pour water from A to C again (C fills up with 3 cups).
  (A=2, B=3, C=3)
- Pour water from C to B (B already has 3, so it gets 2 more to become 5).
  (A=2, B=5, C=1)
- Pour water from B to A (B has 5, pour 4 back to A).
  (A=6, B=1, C=1)
- Pour water from C to B (B gets 1 more cup, making it 4).
  (A=4, B=4, C=0)

Exercise 4:

Understanding the Problem:

- A traveller has a silver chain with 100 links.
- He must pay one link per day to the hotel manager for 100 days.
- The hotel manager will not accept more than one broken link.
- The goal is to determine the minimum number of links to cut so that the traveller can pay exactly one link per day without breaking the rules.

Algorithm

1. **Initialize:** Start with n links and remaining days = n.

Find Segments: Begin with 1-link, then 2-links, then 4-links, doubling each time until the sum reaches or exceeds n.

2. **Cut Links:**

- First cut separates 1-link
- Second cut separates 2-links
- Third cut separates 4-links
- Continue doubling (8, 16, 32...) until total adds up to n.
- If sum exceeds n, adjust the last segment to fit exactly.

3. **End: The number of cuts = the number of segments used.**

- For n = 100 → Cuts at (1, 2, 4, 8, 16, 32, 37) → 6 cuts needed.
- General Case → Keep doubling segments until total = n.

Step-by-Step Approach

- 1-link segment → Covers 1 day
  2-link segment → Covers 2 more days (1 + 2 = 3)
  4-link segment → Covers 4 more days (1 + 2 + 4 = 7)
  8-link segment → Covers 8 more days (1 + 2 + 4 + 8 = 15)
  16-link segment → Covers 16 more days (1 + 2 + 4 + 8 + 16 = 31)
  32-link segment → Covers 32 more days (1 + 2 + 4 + 8 + 16 + 32 = 63)
  Remaining 37 links (Since we need 100 days, and 63 is covered, we need an additional 37 links).
- Total cuts required = 6 (each cut creates a power of 2 segment).

Exercise 5:

Rearrange the letters in “new door” to make one word.

Algorithm

1. **Initialize**

- Input: The given phrase **"new door"**
- Convert it into a list of characters: ['n', 'e', 'w', 'd', 'o', 'o', 'r']

2. **Generate Possible Rearrangements**

- Use an anagram generation method to find all possible words that can be formed.

3. **Check for Valid Words**
4. **Identify the meaningful word**
5. **End**

Solution:

- Rearranging gives **"oneword"**.

Exercise 6:

Sort **6, 5, 1, 4, 3, 2** using the **Divide and Conquer** method.

Algorithm (Merge Sort Approach)

Initialize

- Input array: [6, 5, 1, 4, 3, 2]
- Start Merge Sort.

Divide: Split the array into two halves:

- Left half: [6, 5, 1]
- Right half: [4, 3, 2]

Recursive Sorting:

- Apply Merge Sort to [6, 5, 1]
    - Divide into [6] and [5, 1]
    - [5, 1] → Split into [5] and [1], merge to [1, 5]
    - Merge [6] and [1, 5] → [1, 5, 6]
- Apply Merge Sort to [4, 3, 2]
    - Divide into [4] and [3, 2]

- [3, 2] → Split into [3] and [2], merge to [2, 3]
- Merge [4] and [2, 3] → [2, 3, 4]

Merge Sorted Halves:

- Merge [1, 5, 6] and [2, 3, 4]
- Compare and merge: [1, 2, 3, 4, 5, 6]

End

- Sorted Output: [1, 2, 3, 4, 5, 6]

Exercise 7:

- Flowchart for Simple Interest Calculation

Formula:

Simple Interest = (Principal × N(Time) × Rate) / 100

Flowchart: