1

reference\_sentences = [

"Stores the stuffed array",

"Variables to traverse arrays",

"Loop to traverse in the range [0, N)",

"If the current bit is a set bit",

"Stores the count of consecutive ones",

"Insert into array brr[]",

"Loop to check for next 5 bits",

"If 5 consecutive set bits are found insert a 0 bit",

"Otherwise insert arr[i] into the array brr[]",

"Print Answer"

]

candidate\_sentences = [

"Initializes a new integer array brr[] with a fixed size of 30 to store the result of bit stuffing. ",

"Declares three integers i, j, and k to be used as counters in the loops.",

"Starts a loop that iterates over each element of the input array until it reaches the end.",

"Checks if the current element in the input array is 1.",

"Initializes a counter named count to 1, to track the number of consecutive 1s.",

"Copies the current element to the result array.",

"Starts a nested loop to check for up to 4 consecutive 1s following the current 1.",

"Checks if there are 5 consecutive 1s.",

"If the current element is not 1, simply copies it to the result array.",

"Prints each element of the result array to the console."

]

reference\_sentences = ["Function for bit stuffing",

"Driver Code"]

scandidate\_sentences = ["Declares a static method named bitStuffing that takes an integer N (size of the input array) and an array of integers arr[] (input array for bit stuffing).",

"The main method - entry point of the program."]

2.

reference\_sentences = [

"This function returns value of a Roman symbol",

"Initialize result",

"Getting value of symbol s[i]",

"Getting value of symbol s[i+1]",

"Comparing both values",

"Value of current symbol is greater or equalto the next symbol",

"Value of current symbol is less than the next symbol"

]

candidate\_sentences = [

"Method to return the value of a Roman character",

"Initializes the result integer",

"Gets the decimal value of the current Roman numeral character",

"Checks if there is a next character to compare with",

"If the current numeral is greater than or equal to the next, add its value to the result",

"If the current numeral is less than the next, it means this is a subtractive notation. Subtract the current from the next and add to the result",

"Skip the next numeral since it's already processed"

]

reference\_sentences = ["This function returns value of a Roman symbol","Finds decimal value of a given roman numeral","Function to implement the algorithm","Driver code","Driver Code"]

candidate\_sentences = ["Method to return the value of a Roman character","Method to convert a Roman numeral string to its decimal form","Main method to balance a given BST","Main driver method to demonstrate the balancing of a BST", "The main method to execute the conversion"]

3

reference\_sentences = ["Stores location of the 1st character of the current word",

"Loop to traverse string",

"If the current word ends at index i",

"Update ptr",

"Count of characters is at least K",

"Return answer"

]

candidate\_sentences = [

"Stores the index of the first character of the current word.",

"Iterate through the character array.",

"If the current character is a space, indicating the end of a word, ",

"update 'ptr' to point to the start of the next word",

"Convert the character array back to a string.",

"Return the modified string.",

]

#reference\_sentences = ["Function to capitalize the first character of each word in a string that has at least K characters. The function takes a string 'str' and an integer 'K' as input. It returns a new string with the appropriate characters capitalized."]

#candidate\_sentences = ["Function to capitalize the 1st character of all words having at least K characters"]

4.

reference\_sentences = [

"res variable stores the number of matchsticks initially picked by A",

"If res or N%5 is 0 then there is no chance of A to win the game",

"else return the remainder value after dividing N by 5",

"Total number of matchsticks is N",

"Function call",

"",

"",

"",

"",

"",

]

candidate\_sentences = [

"Calculates the remainder of N divided by 5, casts the result to int, and stores it in the variable res.",

"Checks if the remainder (res) is 0. This condition means N is perfectly divisible by 5.",

"Returns the remainder (res) if it's not 0, which is the result of N % 5.",

"Declares and initializes a long variable N with the value 48, representing the input to the matchGame method.",

"Calls the matchGame method with N as the argument and prints the result. This demonstrates how the method behaves with the given input N = 48.",

"",

"",

"",

"",

]

reference\_sentences = ["Function to calculate the number of matchsticks A should pick initially", "Driver code"]

scandidate\_sentences = ["Defines a static method named matchGame that takes a long N and returns an integer. This method calculates the remainder of N divided by 5.", "The main method, which is the entry point of the program."]

5.

reference\_sentences = [

"Will keep the track of the element position to shifted to their correct position",

"Here we are finding the next Bingo Element for the next pass",

]

candidate\_sentences = [

"Continues the sorting process as long as 'bingo' is less than 'nextBingo'.",

"Checks if the current element is less than 'nextBingo' and updates 'nextBingo' if true.",

]

reference\_sentences = ["Function for finding the maximum and minimum element of the Array", "Function to sort the array","Function to print the array"]

candidate\_sentences = ["A static method named maxMin that takes an integer array 'vec' and its size 'n' as parameters. It updates the 'bingo' and 'nextBingo' variables to the minimum and maximum values of the array, respectively.", "A static method named bingoSort that takes an integer array 'vec' and its size 'n' as parameters. It sorts the array in ascending order and returns the sorted array.","A static method to print the elements of an array 'arr' of size 'n'."]

6.

reference\_sentences = [

"Function to convert input BST to right linked list known as vine or backbone.",

"Make tmp pointer to traverse and right flatten the given BST.",

"Traverse until tmp becomes NULL",

"If left exist for node pointed by tmp then right rotate it.",

"If left dont exists add 1 to count and traverse further right to flatten remaining BST.",

"Make tmp pointer to traverse and compress the given BST.",

"Traverse and left-rotate root m times to compress given vine form of BST.",

"calculate log2 N indirectly using log() method",

"create dummy node with value 0",

"assign the right of dummy node as our input BST",

"get the number of nodes in input BST and simultaneously convert it into right linked list.",

"gets the height of tree in which all levels are completely filled.",

"get number of nodes until second last level",

"Left rotate for excess nodes at last level",

"Left rotation till m becomes 0 Step is done as mentioned in algo to make BST balanced.",

"return the balanced tree",

"Function to print preorder traversal of Binary tree.",

"Function call to implement Day-Stout-Warren algorithm",

"To print the preorder traversal of BST"

]

candidate\_sentences = [

"Converts a BST to a vine (right-skewed tree) and returns the count of nodes",

"Temporary pointer for traversing the tree",

"Loop through the tree nodes",

"If there is a left child, perform rotations to move it up",

"If no left child, move to the right child",

"Temporary pointer for traversing the tree",

"Perform rotations m times to balance the tree",

"Use log base change rule",

"Create a new TreeNode as a placeholder",

"Attach the root to the right of the placeholder",

"Convert the BST to a vine",

"Calculate log base 2 of (count + 1)",

"Calculate the number of nodes for the perfect tree",

"Initial compression to adjust tree structure",

"Further compressions to fully balance the tree",

"Return the balanced tree",

"Performs a preorder traversal of the tree, printing node values",

"Balance the BST",

"Perform preorder traversal"

]

reference\_sentences = ["Function to compress given tree with its root as grand.right.","Function to calculate the log base 2 of an integer","Function to implement the algorithm","Driver code"]

candidate\_sentences = ["Compresses the vine to balance the tree, performing rotations","Calculates the base-2 logarithm of an integer","Main method to balance a given BST","Main driver method to demonstrate the balancing of a BST"]

7.

reference\_sentences = [

"Sort the array",

"Count of triangles",

"The three loops select three different values from array",

"Function call"

]

candidate\_sentences = [

"Sorts the array in ascending order. This is crucial for the logic that checks for the triangle inequality theorem efficiently.",

"Sorts the array in ascending order. This is crucial for the logic that checks for the triangle inequality theorem efficiently.",

"Initializes a counter to keep track of the number of triangles that can be formed.",

"Prints out the total number of triangles that can be formed by calling the findNumberOfTriangles method and passing the array and its size."

]

reference\_sentences = ["Function to count all possible triangles with arr[] elements","Driver code"]

candidate\_sentences = ["Defines a class named GFG.","Main method to run the program."]

8.

reference\_sentences = [

"Check if the input is small enough to solve directly",

"Divide the problem into two smaller problems",

"Combine the solutions of the two smaller problems",

"Define the input",

"Solve the problem using the inverse Ackermann algorithm",

"Print the result"

]

candidate\_sentences = [

"Base case: if n is less than or equal to 4, the method simply returns n.",

"Recursively calls itself with n - 1 and stores the result in variable a.Recursively calls itself with n - 2 and stores the result in variable b.",

"Returns the sum of a and b. This logic resembles that of the Fibonacci sequence rather than any direct relation to the Ackermann function or its inverse.",

"Initializes an integer n with a value of 10.",

"Calls the inverseAckermann method with n as an argument and stores the returned value in the result variable.",

"Prints the result to the console."

]

reference\_sentences = ["Introduces a public class called InverseAckermann.","Driver Code"]

candidate\_sentences = ["Declares a public class named InverseAckermann.","The main method, which is the entry point of the program."]

9.

reference\_sentences = [

"Check if cost for number of digits from 1 to N is less than equal to N",

"ignore right search space",

"check if high can be the answer",

"else low can be the answer",

"else answer will be zero.",

"Function call",

"Function call"

]

candidate\_sentences = [

"If the total digits up to mid are less than or equal to N, narrow the search towards the higher half.",

"Otherwise, adjust the search towards the lower half.",

"After exiting the loop, check which of the bounds closely matches the requirement.",

"If not high but low is a valid answer, print it.",

"If neither is valid, return 0 indicating no number in the range meets the criteria.",

"Calls the method with the total digits N.",

"Calls the method with the total digits N1."

]

#reference\_sentences = ["Function to count total number of digits from numbers 1 to N","Function to find Maximum length of Sequence that can be formed from cost N"]

#candidate\_sentences = ["A static method that calculates the total number of digits up to N.","A static method to find the maximum number of digits in a range that adds up to a total of N digits."]

METEOR scores for easy, medium, and hard code comment generation tasks are XX/XX/XX, respectively. Similarly, the scores for easy, medium, and hard function description generation tasks are XX/XX/XX. These findings suggest that comment generation and function description generation exhibit acceptable performance levels.