**Question**

A class Mixer has been defined to merge two sorted integer arrays in ascending order.  
Some of the members of the class are given below:  
Classname:Mixer  
Data members/instance variables:  
int arr[ ]:to store the elements of an array  
int n:to store the size of the array  
Member functions:  
Mixer(int nn):constructor to assign n=nn  
void accept():to accept the elements of the array in ascending order without any duplicates  
Mixer mix(Mixer A) : to merge the current object array elements with the parameterized array elements and return the resultant object  
void display():to display the elements of the array

Specify the class Mixer, giving details of the constructor(int), void accept( ), Mixer mix(Mixer) and void display() . Define the main() function to create an object and call the function accordingly to enable the task.

**Algorithm**

1. Start

2. Define a class `Mixer` with two instance variables:

- `arr` of type `int[]` to store the elements of an array.

- `n` of type `int` to store the size of the array.

3. Define a parameterized constructor for the class `Mixer`:

- Accept an integer `nn` as a parameter.

- Initialize `n` with `nn`.

- Initialize `arr` with a new array of size `n`.

4. Define a method `accept()` to accept elements of the array in ascending order without duplicates:

- Create a `Scanner` object to read input from the user.

- Prompt the user to enter `n` elements in ascending order without duplicates.

- Use a loop to read and store the elements in the array `arr`.

5. Define a method `mix(Mixer A)` to merge the current object array with the parameterized array:

- Calculate the size of the resultant array as `this.n + A.n`.

- Create a new `Mixer` object `result` with the calculated size.

- Use three pointers (`i`, `j`, `k`) to traverse and merge the arrays `this.arr` and `A.arr`.

- While both arrays have elements left to process, compare and add the smaller element to `result.arr`.

- If elements are equal, add one and move both pointers.

- Copy any remaining elements from `this.arr` or `A.arr` to `result.arr`.

- Return the `result` object.

6. Define a method `display()` to display the elements of the array:

- Use a loop to print each element of the array `arr`.

7.In the main method:

- Create a `Scanner` object to read input from the user.

- Prompt the user to enter the size of the first array and create a `Mixer` object `obj1` with this size.

- Call `obj1.accept()` to read the elements of the first array.

- Prompt the user to enter the size of the second array and create a `Mixer` object `obj2` with this size.

- Call `obj2.accept()` to read the elements of the second array.

- Call `obj1.mix(obj2)` to merge the arrays and store the result in a new `Mixer` object `merged`.

- Call `merged.display()` to print the merged array.

- Close the `Scanner` object.

8. End

**Variable Description**

|  |  |  |
| --- | --- | --- |
| **Variable** | **Data type** | **Purpose** |
| arr[][] | int | To store elements of the array |
| n | int | To store the size of the array |
| i | int | To transverse and merge the arrays |
| nn | int | To initialize value of n |
| size | int | To determine total size of merged array |
| size1 | int | To determine size of 1st array |
| size2 | int | To determine size of 2nd array |
| j | int | To transverse and merge the arrays |
| k | int | To transverse and merge the arrays |