**For all of the programs below submit all of your java code and the output of your program to show that it works.**

1. **ResizingArrayQueueOfStrings.** Develop a class ResizingArrayQueueOfStrings that implements the queue abstraction with an array. Your implementation should use array resizing to grow and shrink the available storage when appropriate. Write a client program that can be used to test your class. The client program should present a list of the possible operations and enable the user to choose (as many times as they like) an operation to execute, prompting the user for input when needed.
2. **MagicSet.**  Write a class that represents a set. Recall that a set has the following properties it does not contain duplicates and order is not important. If you are unable to make a class that can use any type (generics), make your set handle whole numbers. You must implement your set using an array. The amount of storage used should grow and shrink as needed: it should be halved if the array is 25% full and doubled when full. Your set should support the following operations:
   1. Add a value
   2. Does the set contain a value
   3. Delete a value
   4. Union: return a new set containing all your values and another set
   5. toString

Here is a UML diagram for a MagicSet that only handles whole numbers. Write a client program that can be used to test your class. The client program should present a list of the possible operations and enable the user to choose (as many times as they like) an operation to execute, prompting the user for input when needed.

|  |
| --- |
| **MagicSet** |
| -set:int[]  -next:int |
| +MagicSet()  +MagicSet(other:MagicSet)  +add(value:int):void  +contains(target:int):Boolean  +delete(value:int):Boolean  +union(other:MagicSet):MagicSet  +toString():String |