We want to have to make use of polymorphism to perform operations on sums, counts, and averages.

1) Create a class named ***Aggregator***.
   a) The ***Aggregator*** class has a data member named ***Numbers***, consisting of a list of integers (should be protected readonly) using ***List<int>.***
   b) A constructor initializes that array of integers to the list, but if the given array is **null**, throw an ***ArrayNullArgumentException***.
   c) A **public abstract get-property** named ***Value*** that simply returns an integer value.
   d) A public virtual method named ***Append*** that takes an ***Aggregator*** instance as an argument and returns an integer. This method will append all the numbers from the instance's list to the current list of numbers. If the aggregator instance argument is null, throw ***ArgumentNullException***.
   e) The ***To String*** override method will print all the numbers from the list, separated by a space.
2) Create a class named Sum that extends the ***Aggregator*** class.
   a) The ***Sum*** initializes a list of integers via the ***base*** constructor.
   b) The ***Value*** simply returns the sum of all the integers in the list.
   c) The ***Append*** override method does the following:
      i) If the instance is the instance of the **Sum**, it will add each number to the list, and if there are leftovers, those numbers will be appended to the list.
      ii) Otherwise, it will call the base implementation.
3) Create another class named ***Count*** that extends the Aggregator class.
   a) The ***Count*** class initializes a list of integers via the base constructor.
   b) The ***Value*** simply returns the count of all the integers in the list.
4) Create another class named ***Average*** that extends the ***Sum*** class.
   a) Create a ***readonly*** private field named _count for instances of ***Count.***
   b) The constructor will also initialize both instances of ***Sum*** with the given numbers. In addition, if the array is empty, throw an ***ArgumentOutOfRangeException***.
   c) Create a ***property*** named ***DecimalValue*** that simply returns the decimal value of the average of these numbers by using both instances of ***Sum*** and ***Count***.
   d) Override the ***Value*** property that returns the integer version of ***DecimalValue***.
5) Test your classes.