CPSC 3200 Object-Oriented Development

Programming Assignment #3: Due Friday April 29, 2022 before MIDNIGHT

P3 exercises your understanding of inheritance and Dependency Injection

For an acceptable P3 submission:

1. Design using inheritance and Dependency Injection
2. Using C#, fulfill requirements as specified in steps 1-7 from P1

Part I: Class Design

Design an inheritance hierarchy of *dataExtractors*, where each *dataExtractor* object encapsulates two integer arrays x and y that individually contain only unique values. They may be of different lengths but each must be of some minimum (which varies by object) non-zero length. x is acquired via Dependency Injection; hence, a defined error response is expected. Invalid client requests cause a state change.

A *dataExtractor* object supports client requests to retrieve data as follows.

1) *any()* -- returns some composite from array x and/or array y
2) *target(z)* – returns z values selected according to state
   a. from either array x or array y
   b. where all values are either odd or even
3) *sum(z)* – returns the sum of z values according to state
   a. from either array x or array y
   b. where all values are either odd or even

Define two descendant classes, where

dataHalf object is-a *dataExtractor* and thus operates like a *dataExtractor* object, except that:

a. every number in x is divided by 2
b. when the number of failed client requests exceeds a bound (which varies from object to object), the object is deactivated
c. *any()* – returns the same composite for two successive requests
   the 1st and 2nd any() request return the same composite
   the 3rd and 4th any() request return the same composite

dataPlus object is-a *dataExtractor* and thus operates like a *dataExtractor* object, except that:

a. y is initially concatenated with a, where a is not the same number across all objects
   e.g. if y is [3, 44, 7, 56, 2] then ya could be [3, 44, 7, 56, 2, 871]
b. upon every n == j*kth client request, where j starts at 1 and increments with every kth request, and k varies from object to object, n is concatenated to the end of current array y.
c. *target(z)* – returns z odd values from array x concatenated with z even values from array y

Many details are missing. ***You MUST make and DOCUMENT your design decisions!!***
Do NOT tie your type definition to the Console.

Use Unit Testing to verify functionality of each class => 3 test files.

Part II: Driver (P3.cs) -- External Perspective of Client – tests inheritance hierarchy design

The P3 driver must test the use of all 3 types together.

Unit Testing ensures the functionality of each type, separately.

Thus, the driver will differ from the unit tests which test each type separately.

1) Use at least one heterogeneous collection for testing collective functionality
2) Instantiate a variety of objects
3) Trigger a variety of mode changes