1. **Introduce an integrated solution using Abstract Factory Pattern, Builder Pattern and Singleton Pattern for the above descriptions and explain the advantages and trade-off of each folder in terms of SOLID principles, cohesion and coupling.**

S - Single Responsibility Principle:
except the Single Responsibility in the Master Control file as the Master Control file is modified to call four different behaviours(input,sorter,shifter) except that we can see that the code modules have only one responsibility to perform.
We can see that every module or class have only one responsibility to perform if it is taking input then only on put behaviour from that class is reflected, same goes for sorter, shifter and output classes. We are making use of Abstract Factory for that.

O - Open close Principle:
Every module the code need not to be modified or changed as per behaviour requirements rather we can extend the code to reach the desired functionality. As we can observe from the code that if we ned to add any more functionality we need to extend the code rather than commenting out or modifying the code.

L - Liskov substitution principle:
We can use either the derived class or the base class for the code to call the base class objects without the need of any modifications.

I - Interface segregation principle:
There is no model that forcefully is required to use the extra implemented behaviours from the interface or the abstract classes.

D - Dependency inversion principle :
As we can see from code, we can see that everything is hidden and dependant on the interfaces and abstract classes rather than other modules.

Advantages :

1 - Using Abstract factory Pattern we have the following advantages :
A : Loose Coupling between modules.
B : Easy to create objects.
C : Use of Interfaces thus leading to good information hiding and encapsulation.

2 - Using Builder Pattern we have the following Advantages:
A - Control over Object Creation: It allows us to build the line object which is a complex implementation and is used in input module.
B - Encapsulation: The code constructor implementation is encapsulated by hiding the construction process from client code.

Disadvantages:

1 - For both Builder Pattern and the Abstract Factory Pattern we can see that initial code was quite easy to understand but as we have applied both pattern we can see that the code is quite complex now, with additional interfaces and abstract classes it become quote difficult to understand the flow of the data.

Cohesion and Coupling:

1 - Abstract Factory pattern increases the coupling between the modules to the formation of concrete factories to call out the objects.
2 - Builder pattern applied in line class increases cohesion by separating the construction of objects from their representation.