benefit of linear constructor chaining:

1. provide flexibility, able to create object based on the different amount of data (parameters/ properties) on hand

2. avoid code duplication, create a separate constructor for each task and make links or chains among them, so as to increase the readability of the code.

3. If do not perform chaining among constructors and they require a specific parameter, then we will need to initialize that parameter twice in each constructor.

4. perform multiple tasks in a single constructor, rather than creating a code for each task in a single constructor we create a separate constructor for each task and make their chain which makes the program more readable.

5. ease of maintenance, when adding a new property, the downstream constructor will automatically be modified

The benefit of using this technique is that it can be used to reuse code, reduce duplication, and ensure that objects are created in a consistent and controlled state. By calling a constructor from another constructor, you can set default values, ensure that objects are created with the correct state, and simplify the code required to create an object. This can improve the maintainability and readability of your code.

difficult to add new property:

1. need to add new property for each constructor one by one

2. re-examine the chain to confirm the constructor calling the right constructor inside

3. easy to make mistake that forget to modify constructor when adding new property

Adding a new property to the EventSchedule class would have complicated the implementation in the following ways:

Initialization: When a new property is added, it would need to be initialized in each of the constructors in the linear chain. This could result in code duplication and could be prone to errors, such as forgetting to initialize the new property in one of the constructors.

Maintenance: If the new property requires a specific value to be set during initialization, this value would need to be passed to each of the constructors in the linear chain. This could result in complex constructor signatures and make the code difficult to maintain.

Updating existing code: If the new property requires a specific value to be set during initialization, any existing code that creates instances of the EventSchedule class would need to be updated to pass this value to the constructors. This could be a time-consuming task and could result in errors if not all instances are updated.

In terms of a non-linear chain, it can be easier to update when new features are added to the class. With a non-linear chain, the constructors can be updated individually, reducing the risk of introducing errors into the code. This also allows for greater flexibility and reduces the complexity of the constructor signatures.