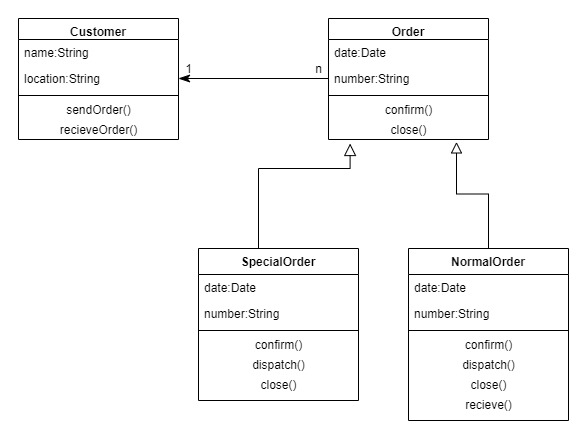
**1. Class Diagram**



Mainly, we have 2 elements or objects :

1. Customer
2. Order

One customer can create multiple orders, so the **cardinality ratio is one-to-many.**

**Customer** class having 2 attributes :

* name(datatype : String)
* location(datatype : String)

And 2 methods or operations :

* sendOrder()
* receiveOrder()

**Order** (super class) class having 2 attributes :

* date(datatype : Date)
* number(datatype : String)

And 2 methods or operations :

* confirm()
* close()

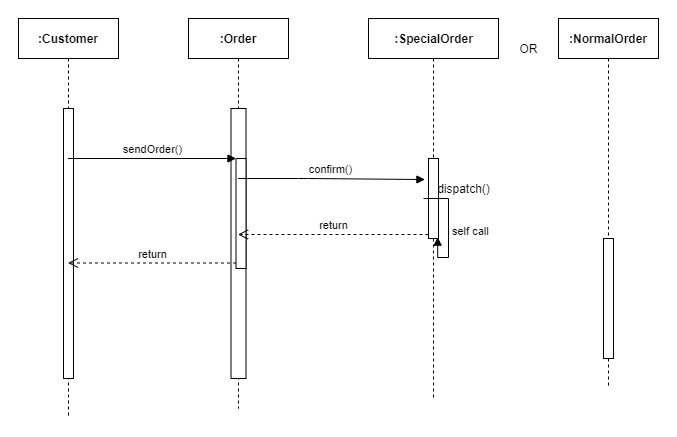
Order is an abstract class(because its functions’ definition are not defined in this class) and has two sub classes :

* SpecialOrder
* NormalOrder

Here, generalization modelling approach is defined, we have super class Order and 2 lines coming into, each with a non filled triangle.

These two subclasses or derived classes are implementing attributes and methods of the Order class along with their additional un-implementing methods like dispatch() and receive().

**2. Sequence Diagram:**



It has 4 objects, participating in the interaction:

* Customer
* Order
* SpecialOrder
* NormalOrder

**It is showing the message sequence .**

1. The first call is sendOrder() passing to Order Object or an method of Order object.
2. The second call is confirm() which is a method of SpecialOrder() object.
3. The third and last call is dispatch() which is also a method of SpecialOrder object.

**Return/Reply message** is defined using dotted line arrow.

**Dotted lines** show lifeline.

**Rectangular box or vertical line** represents the time needed for an object to complete a task which is known as Activation Box or execution.

**Overlapping rectangles** represent overlapping execution specifications- callback message or use for constructor call (order object creates another object SpecialOrder invoking constructor).

**"U shaped arrow**" shows self message/reflexive message, where the object is sending a message to itself.