Spring Fundamentals:

* Spring will create the objects of the class. It would be responsible for any updates. If there is any change in the class name spring will handle this. We won’t be using the new keyword to create an object. In the configuration file of spring, we will not only mention the class name but also initialize.
* All the objects created in each layer (like we create Dao object in business layer, business layer object in service layer) all this would be done by the spring framework. This is dependency injection. One class is dependent on the other class for its services. These dependencies should be automatically resolved.
* Spring is responsible for dependency injection. Spring would be responsible to create objects of all the layers
* Laws of demerit.
* Non Invasive approach – This approach gives the benefit to avoid sharing the data from the developer. In this approach the class names will come from some xml file.
* Context.xml – It is the main configuration file for spring
* Every object which you needs to be created you need to mention the class name in context.xml
* Frameworks try to handle unchecked exception
* Autowiring is done on constructor, field
* Resource annotation given for dependency injection.
* Interception – It can be used for security, transaction handling, logging
* AOP – Aspect Oriented Programming depends on Interception
  + Advice/Aspect – Before/After
  + Join Points – on which method call Aspect is called.
  + PointCut – Defined on advices to decide the join points
  + Special Compiler – When we need a special compiler to call the before and after methods.
  + Load Time weaving – When the special class loader is used to compile the code in the sequence of before->join points -> after
  + Proxy Objects :
    - Before()
    - Abc()
    - After()
* Aspects types
  + Try{
    - Method()
    - afterReturning.
  + }catch(){
  + afterThrowing
  + }finally {
  + After
  + }