## 1. Autowiring

Create a class called Passport that has the below attributes:

|  |  |
| --- | --- |
| passNum | int |
| dateOfIssue | LocalDate |
| dateOfExpiry | LocalDate |

Create another class called Employee that has the below attributes

|  |  |
| --- | --- |
| empId | int |
| empName | String |
| passObj | Passport |

All attributes, Getters and Setters are provided as a part of code skeleton.

The Passport object should be autowired above the property via annotations. Create a class called ApplicationConfig that has the required annotations for scanning and registering the bean definitions.

Create a class called Driver with the main method and write the logic to verify the correctness of your code. (Note: UI will not be tested).

2. Susan created a  java application for currency converter. Later she got familiar with  spring core using Maven and wanted to use the features of it to make her application loosely coupled. Create a class called CurrencyConverter that has the following attribute:

|  |  |
| --- | --- |
| mObj | Map<String,String> |

Map that holds currencyName and the conversion rate. The map needs to be configured in the xml and injected into the CurrencyConverter via setter based injection. Write a method called getTotalCurrencyValue that accepts a String. The string contains the total currency and the currencyName.  For ex: “7Dollar”. The method needs to parse the string and depending on the user input fetch the appropriate currency value from the xml file calculate the total rate and return the result back to the user.

Suppose if the user has entered a value “5Dollar”,  then the output should be 325.

The map values in the xml file for various currencies are as follows:

|  |  |
| --- | --- |
| DOLLAR | 65 |
| EURO | 80 |
| DINAR | 218 |

3. Create a class called Product  with the below attributes

|  |  |
| --- | --- |
| productId | String |
| productName | String |
| price | double |

* Include getters and setters for all the above-mentioned attributes.
* Product  class should be registered as a bean with the spring container via XML file
* The Product details are prepopulated in the xml file. [Refer code template]
* ArrayList class should be registered as a bean with the spring container via XML file and inject each Product to the List through constructor-based injection.

Create a class called Cart  with the below attributes

|  |  |
| --- | --- |
| Attribute Name | Value |
| productList | ArrayList<Product> |

* Include getter/setter for the above attribute
* Cart  class should be registered as a bean with the spring container java based configuration
* The values for these attribute productList should be injected via setter based injection.
* Cart should be registered as a bean with the spring container via java based configuration,  productList should be injected using setter based injection into the Cart class
* Create a class called Driver with the main method. Retrieve the Cart object , calculate the total price based on the productList and display the output.

**Assumption:**

**Assume the bean id for ArrayList bean in the beans.xml is prodList**

**Design Constraints:**

* Product class should be present in com.spring.app package
* Cart class should be present in com.spring.app package
* Main class should be present in com.app.driver package
* Product and Cart class should have the necessary getters and setters.
* The className/Attribute Name/PackageName should be same as specified in the problem statement. Do not create any new packages.
* The XML configuration should be done in the file beans.xml. This XML should be available under the resources folder of the created maven project.

**Sample Input/Output:**

Sample Output:

Total Price  Rs: 4500.0