1. Write a java program

***Hi, Hearty Welcome to 24GB!!***

1. Find the total number of Lower case letters
2. Find the total number of Upper case letters
3. Find the total number of total vowels exist (upper and lower case letters are same)
4. Print individual count of each vowels (upper and lower case letters are same)
5. Find the total number of spaces
6. Find the total number of special characters
7. Find the totalnumber of “Numbers”
8. Print like :***iH, ytraeHemocleWot BG42 !!***(Just reverse of each word)
9. Print in following order Lower Case letters, Upper case letters, Numbers and Special Characters

NOTE: Input Sentence should be dynamic (ie., works for any sentences)

1. Create n\*n *(‘n’ should be odd number)*integer 2D array calculate the totals of each rows, columns & diagonals and store it 1D integer array *(Condition: Use only 2 ‘for’ loops both for getting input as well as for calculation)*

NOTE: Order of storing the results in 1D resultant array is up to your choice

1. Write a java program to get up to 10 digit number and find the following*(Condition: Use only one ‘while’ loop)*
   1. Find the total number of Odd digits
   2. Find the totalnumber of Evendigits
   3. Find the total number of one digit Prime numbers (Check the digit prime or not)
   4. Find the total number of digits divisible by ‘2’ except ‘4’
   5. Find the total number of digits
2. Create two HashTableswith the following data (Key: Integer, Values: Arraylist)

|  |  |  |  |
| --- | --- | --- | --- |
| *Employee\_Details* | | | |
| *Emp\_ID* | *Emp\_Name* | *Salary* | *Dept* |
| 251 | MasoodAzhar | 10000 | Finance |
| 355 | Abu Bakr | 20000 | Logistics |
| 754 | MuhammedRasul | 5000 | Sales |
| 384 | SubhanQuershi | 2500 | Sales |
| 463 | Hafiz Saeed | 25000 | Purchase |
| 835 | Shekau | 14500 | Purchase |

|  |  |  |
| --- | --- | --- |
| *Grade\_Details* | | |
| *Grade* | *Low\_Salary* | *High\_Salary* |
| A | 1000 | 4999 |
| B | 5000 | 14999 |
| C | 15000 | 29999 |

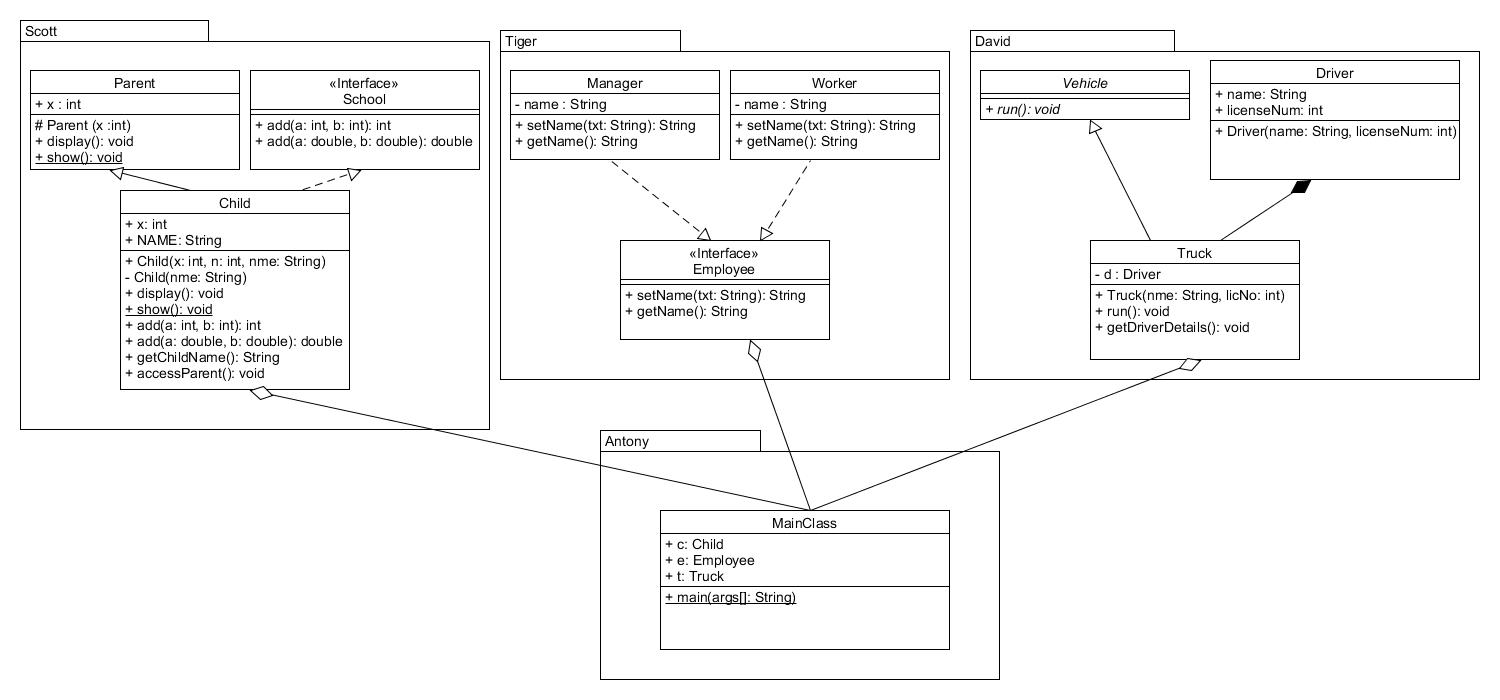
1. Find the employees whosesalary is above 7500
2. Find the employees works in “Sales” department
3. Find the largest and smallest salaried employees details including their salary grades
4. Find the Grade “B” salaried employees

NOTE: For finding ‘Salary Grade’, compare ‘Salary’ in Employee\_Detailshashtable with ‘Low\_Salary’ and ‘High\_Salary’ in Grade\_Detailshashtable

1. Write a ArrayList program that only accepts Integer and
   1. Print elements in ArrayList in reverse order using ListIterator
   2. Sort the elements

NOTE: Sorting should be done by rearranging of elements in same ArrayList

1. Write a java program that raises any of the runtime exception and do the following
   1. Handle it the same runtime exception
   2. Handle it with “Exception” class
   3. Try without ‘catch’ block
2. Write a java program to implement below ‘UML Class Diagram’*(Refer: UML Class Diagram Guide)*

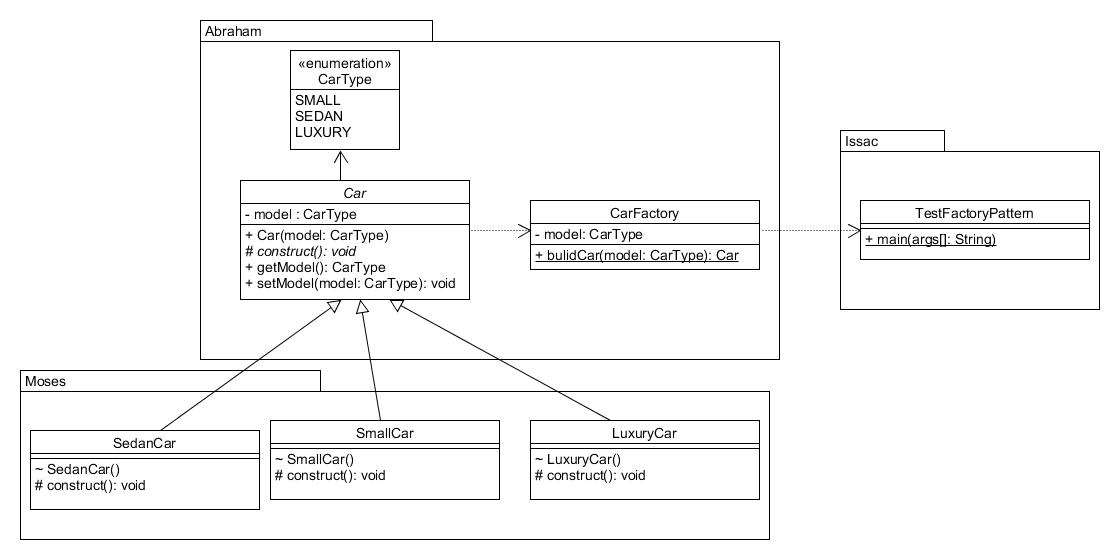


Do the following in ‘main method’ in MainClass.java

* 1. Initialize object as ‘c’ for Child class through its public constructor and do the following
     1. Call ‘display()’ method
     2. Call ‘show()’ method
     3. Call 2 ‘add()’ methods
     4. Call ‘getChlidName()’ method
     5. Print value of ‘x’
     6. Call ‘accessParent()’ method
  2. Initialize objects for Manager and Worker class through its parent reference ‘e’ and access its methods
  3. Initialize object for Truck class as ‘t’ and call its methods
     1. Call ‘run()’ method
     2. Call ‘getDriverDetails()’ method

NOTE:

* 1. In Parent Class
     1. Initial class variable ‘x’ through its protected constructor
     2. Print “Parent Display” in ‘display()’ method
     3. Print “Parent Show” in ‘show()’ method
  2. In Child class
     1. In public constructor “*Child(x: int, n: int, nme: String)”* use ‘x’ to initialize class variable ‘x’ , ‘n’ to initialize parent class variable ‘x’ and ‘nme’ to call private constructor
     2. Initialize its final variable ‘NAME’ through private constructor
     3. Print “Child Display” in ‘display()’ method
     4. Call parent class ‘display()’ method
     5. Print “Child Show” in ‘show()’ method
     6. Return results in ‘add()’ method
     7. Return ‘NAME’ in ‘getChlidName()’ method
     8. Call ‘show()’ method and print value of ‘x’ of parent class in ‘accessParent()’ method
  3. In Manager & Worker class
     1. Initialize class variable ‘name’ through ‘setName()’ method
     2. Return ‘name’ in ‘getName()’ method
  4. In Vehicle class (abstract class)
     1. ‘run()’ is abstract method
  5. In Driver Class
     1. Initialize ‘name’ &licenseNum‘ through its constructor
  6. In Truck Class
     1. Initialize object ‘d’ for ‘Driver class’ in its public constructor
     2. Print “Truck Run” in ‘run()’ method
     3. Print ‘name’ & ‘licenseNum’ of ‘Driver’ class in ‘getDriverDetails()’ method

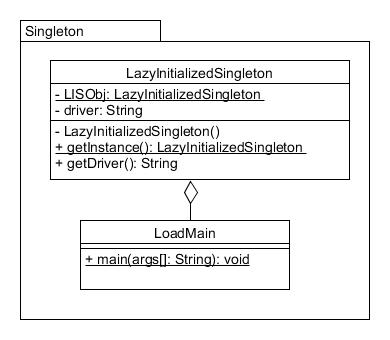
1. Implement the following Factory Pattern in javaDiagram’ *(Refer: UML Class Diagram Guide)*

Do the following in ‘main()’ method in TestFactoryPattern Class

* 1. Create an object for CarFactory Class
     1. Call buildCar(CarType.SMALL) method
     2. Call buildCar(CarType.SEDAN) method
     3. Call buildCar(CarType.LUXURY) method

NOTE:

1. In Car Class (Abstract Class)
   1. Initialize private class variable ‘model’ through public constructor
   2. ‘construct()’ is abstract method
   3. Return ‘model’ in ‘getModel()’ method
   4. Set ‘model’ through ‘setModel()’ method
2. In SedanCar Class
   1. Call parent constructor from constructor with ‘SEDAN’ CarType argument
   2. Call ‘construct()’ from constructor
   3. Print “This is SEDAN CAR” in ‘construct()’ method
3. In SmallCar Class
   1. Call parent constructor from constructor with ‘SMALL’ CarType argument
   2. Call ‘construct()’ from constructor
   3. Print “This is SMALL CAR” in ‘construct()’ method
4. In LuxuryCar Class
   1. Call parent constructor from constructor with ‘LUXURY’ CarType argument
   2. Call ‘construct()’ from constructor
   3. Print “This is LUXURY CAR” in ‘construct()’ method
5. In CarFactory Class
   1. Return ‘Car’ object based on CarType argument in ‘buildCar()’ method
6. Implement the following Singleton Pattern in JavaDiagram’ *(Refer: UML Class Diagram Guide)*



Do the following at ‘main()’ method in LoadMain class

1. Call ‘getInstance()’ method to create object for ‘LazyInitializedSingleton’ Class
2. Call ‘getDriver()’ method and print the string

NOTE:

1. In LazyInitializedSingleton Class
   1. Initialize class variable ‘LISObj’ & ‘driver’ to NULL
   2. In private constructor, initialize ‘driver’ to “Singleton Lazy Initialization”
   3. In ‘getInstance()’ method , if ‘LISObj’ is NULL then create an object for LazyInitializedSingleton class and return the object
   4. Return ‘driver’ value in ‘getDriver()’ method