Day 1 (24th July 2024)

Set 1 :

1) Write a java program to print the below series of numbers 1,3,6,9,12,……..

2) Write a java program to accept number of days as integer and display total number of years, months and days out of it.

3) Identify the classes, attributes, methods and the relationships between the classes for the below use case. You can show the relationship by using class diagrams.

   Electrical equipment manufacturing company wants to store details of their products. There are 2 types of equipments: Bulbs and Tubelights. Both have some common properties like device id, illumination power and warranty period in years. For Tubelights the application should store additional details like length, width, color. For Bulbs the application should store additional details like color, holder type, gas used in the bulb. Application also maintains details of refrigerator like storage capacity, color, bulb used in the refrigerator.

Day 2(25th July 2024)

Set 1 :

(Note: In all the below given programs, create a separate class with main method and call the operations)

1) Write a java program which will accept elements for an array from user. Create a separate class which will have methods to perform following operations:
   1. Sort the array using bubble sort.
   2. Accept an element and return position of that element in the array. If element is not present, then return -1.
   3. Accept a position and remove element from that position in the array.
   4. Accept an element and check how many times the element has been repeated in the array.
   5. Display elements by removing the duplicates . if the elements of the array are [9,4,4,7,10,7] then method should return [9,4,7,10]

2) Create a class MathOperation with 2 data members num1 and num2 to store operands and 3rd data member result to store result of operation. Write following methods in the class:
   1. init – to accept values for num1 and num2 from user.
   2. add – to add num1 and num2 and store in result.
   3. multiply – to multiply num1 and num2 and store in result.
   4. power – to calculate num1 ^ num2 and store in result.
   5. display – to show value of result.

3) Write a program to test the below given methods of String class:
    a. charAt
    b. contains
    c. length
    d. indexOf
    e. stripLeading
    f. equalsIgnoreCase

Day 3 (26th July 2024)

Set 1

Note: All the classes should be part of packages. Create proper package hierarchy

1) Create a class Movie with the attributes movie name, produced by, directed by, duration, year, category (comedy/action/..) . Write necessary methods to accept and display the information. Create the constructors based on the below rules.
    a. Movie name and produced by are mandatory fields and should be supplied at the time of creating a class
    b. Compiler should raise an error when we try to create Movie object with out passing any parameters.
    c. Write a constructor which accepts all the attributes as parameters while creating the object. From this constructor call the constructor (mentioned at point a) to initialize mandatory field
2) In the above class create a static variable **moviesCount.** Write necessary methods to get the values. Every time an object of Movie is created, increment the value of movesCount variable.
3) Create a movieId field. Make this variable as a readOnly method (make it private and write only getter method). Generate movieId value by using the below formula
   movieId=”movieName”+”_”+moviesCount
   eg. if the Movie name is “Hello” and the value of moviesCount variable is 31, then store Hello_31

4) Class SpecialMovie contains all the attributes of Move class and other attributes to store the technology used for soundEffects and visualEffects. Class InternationalMovie contains all attributes of Move class and other attributes to store country and language.
    a. Write necessary classes, constructors and methods for storing and displaying additional information.
       Hint: use super keyword to call the methods/constructor of super class.
5) In the Movie class write a method called showDetails() which returns a String by concatenating all the field values.

Day 4 (29th July 2024)

Set 1:

6) Override showDetails() method in the sub classes to include the details of additional fields. Use super key word to call the methods of super class if required.
7) Create an array of Move class and store Objects of Move, SpecialMovie and InternationalMovie classes in the array.

8) Using a single for loop try traverse the above array and call the method showDetails() on all the objects of the array. Understand the concept of runtime polymorphism.
9) Check is it possible to call all the methods of SpecialMovie and InternationalMove while traversing the array. If not use typecasting to achieve the above task.
10) Complete the below tasks
    a. Create Interfaces
    b. Create inheritance by extending other interfaces
    c. Creating class by extending another class and implementing more than 1 interface
    d. Create a reference variable of an interface.
    e. Create a class implementing above interface.
    f. Store the object created in step e in the reference variable created in step d.
    g. Call the methods by using interface reference

Day 5 (30th July 2024)

11) A layered application is responsible for storing the details of Employee objects. View layer is responsible for user interaction and is only responsible for accepting and displaying details from/to user. View layer is will communicate with storage layer for all CRUD operations. Develop the applications based on the below class structure. Class structures, method information and responsibilities of each class are given below. Create a main class to test your application.

Entity class Employee contains empno, first name, last name, city & salary attributes.

View Layer

UserUI (Class)

acceptEmpDetailsAndStore()
displayEmpByEmpno()
Responsibility :
This class accepts the details from the user and calls the methods on Storage layer.
Call StorageFactory.getStoraget() to get the Storage implementation object

Storage layer

Storage (Interface)

void addEmployee(Employee e)
Employee getEmployee(int empno))

StorageImpl (Class)

Responsibility:
This class implements Storage Interface
Create an array to store the employee objects and provide the proper implementation to the methods in the interface

StorageFactory (Class)

public static Storage getStorage()

Responsibility:
This method is used to get the object of Storage implementation. View layer should call this method to get

Day 6 (31st July 2024)

1) Test the below concepts by writing necessary classes and methods
   a. Single try .. catch .. finally block
   b. Multiple catch block
   c. Nested try catch blocks
   d. Usage of throw and throws keywords.
2) Create a checked exception InvalidAgeException. Override all the constructors from the super class
3) Create a RuntimeException WrongInputException. Override all the constructors from the super class

4) Create the below two classes. LearningModule class is dependent on AgeValidator class to check whether the age is valid or not.

LearningModule

public String getLearningCode(String Name, age) throws WrongInputException.

Responsibility:
This method accepts Name and age of he person and returns generates a learning code by concatenating name and ageCode. (call getAgeCode from AgeValidator to get age code)
In case of InvalidAgeException throw WrongInputException by wrapping InvalidAgeException.

AgeValidator

public String getAgeCode(int age) throws InvalidAgeException.

Responsibility:
This method accepts age and returns "Adult" if the age is greater than 18 and returns "Kid" if the age is between 0 to 18. If the age is less than 0 then throw InvalidAgeException

Create the Main class for accepting the data from the user and call the methods on LearningModule object. In case of any exceptions catch the exceptions and display the appropriate message to the user.

5) Modify the Storage interface given the Day 5 as given below.

Storage (Interface)

void addEmployee(Employee e) throws EmployeeAlreadyExistsException

Employee getEmployee(int empno)) throws EmployeeNotFoundException

- While adding the employee if the employee with the same id already exists in the array Raise EmployeeAlreadyExistsException

- While Searching if the Employee with the given empno is not found in the array then raise EmployeeNotFoundException.

Modify other classes accordingly.

Day 7 : (1st Aug 2024)

Set 1 :

Note : use employee.txt file for solving the below problems

1) Create an implementation class StorageFileImpl implementing **Storage interface** given in day 5 assignment.
2) Create an array of type Employee like Employee[] employees ;
3) Initialize the array in StorageFileImpl Class Constructor with size as the number of lines in employee.txt
4) Implement the unimplemented methods of Storage Interface as given below.

void addEmployee(Employee e) throws EmployeeAlreadyExistsException	Read each line from employee.txt file and create an employee object with each line and add to employees array.
Employee getEmployee(int empno)) throws EmployeeNotFoundException	Check if an employee is present or not . return Employee if present else throw an EmployeeNotFoundException

5) Add new method in **StorageFileImpl** class to return an array of all Employees
6) Test all methods by calling from Main method.

Day 8 (2nd Aug 2024)

Set 1

1) In the previous question make the below modifications to the Employee class.
    a) add Date of Joining field of type Date
    b) Override hashCode and equals method

   Implement comparable interface. While comparing two objects consider empno filed.
2) Create different implementation classes to Storage interface as mentioned below

a. StorageListImpl : Use Array list for storing the employee objects
b. StorageSortedImpl : Use TreeSet for storing the employee objects

c. StorageMapImpl : Use HashMap for storing the employee objects as values and empnos as keys.

3) Modify getStorage() as mentioned below. Method should accept integer as a parameter
   a. Storage getStorage(int code) method
   b. If code =1 then return StorageImpl object
   c. If code = 2 then return StorageListImpl
   d. If code=3 then return StorageSortedImpl
   e. For any other value return StorageMapImpl

Day 9: (5th Aug 2024)

Set 1 :

1)

Create a class Movie

```
Class Movie {
  Private int id;
Private String name;
Private String hero;
Private int yearOfRelease;
Private Double rating;

 // create getter and setter methods
// create all argument constructor
// create no argument constructor
// create toString method

}
```

a) Create a functional interface MovieFinder

   Public boolean findMovie(Movie movie);

b) Create a class MovieUtils

```
Class MovieUtils{

Public static void getMovies(List<Movie> movies , MovieFinder movieFinder){

    For(Movie movie:movies){
       If(movieFinder(movie)){
      System.out.println(movie);
```

```
}

}

}
}
```

c) Create main class

```
Class MoviesDemo{
Public static void main(String[] args){

// create List of 10 movies


// 1  print all the movies whose names starts with “a”
// 2. Print all movies  whose rating is greater then equals to 3.5
// 3. Print all movies a specific hero name
// 4 Print all movies which are released in a specific year

}
}
```

// Call MoviesUtil getMovie method passing Lambda of MovieFinder

2) Print all movies details using forEach method of List
   Hint : Use Consumer interface for question 3s

Day 10 : (6th Aug 2024)

Set 1 :

1) Write the proper SQL instructions for the below actions
   e. Create a table Product (productid, productname, description, price). Product id is a primary key
   f. Create a table Purchases (purchaseid, productid, number of items, date of purchase,). Purchase id is a primary key and product id is foreign key .
   g. Insert, update and delete the rows in the above table.
   h. Write SQL queries for
      i. List all products
      ii. List all products with price less than 100
      iii. List all products with productname starts with “P”

iv. List all Purchaseid, productid, product name, price and number of items
v. List all purchaseids where price is less than 100
vi. List all purchaseids where product name is “Pen”

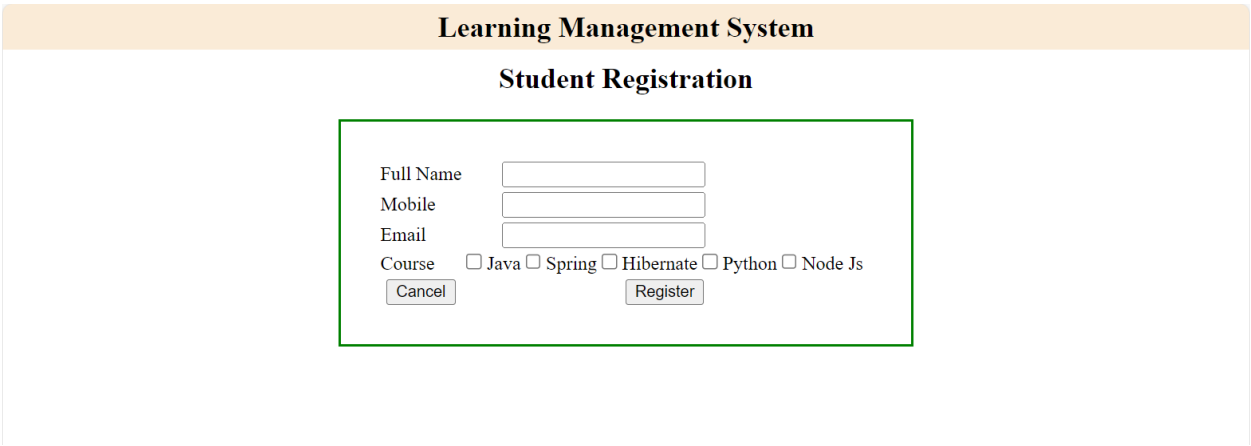
2) Write a class for ProductStore and write all the necessary methods for below operations.
Note : Use appropriate functional interfaces available in stream api
   a. Public void addProduct(Product p) throws ProductAlreadyExistsException (Raise this exception the data base throws Record already exists exception)
   b. Public void updateProduct(Product p)
   c. Public List<Product> getAllProducts()
   d. Public List<Product> getProductsByPrice(int price)
   e. Public void addTwoProducts (Product p1, Product p2) throws ProductsNotAddedException (Raise this exception even if we are not able to add any one product. Use transactions for this operation)

Day 11 : No Assignment on this day(7th Aug 2024)

Day 12: (8th Aug 2024)

Set 1 :

12) Write the HTML file to create the below web page



- When the user click on cancel button it should clear the all the entries .
- When the user clicks on register button, it should validate the email and mobile fields as per the given description.
  - Mobile numbers should be 10 digit number.
  - Mobile should start with only 6 or 7 or 8 or 9

- Email should be in email id format like abc@xyz.com . write custom logic for validating the email.
- User should select at least one course.
- Full name should not empty.

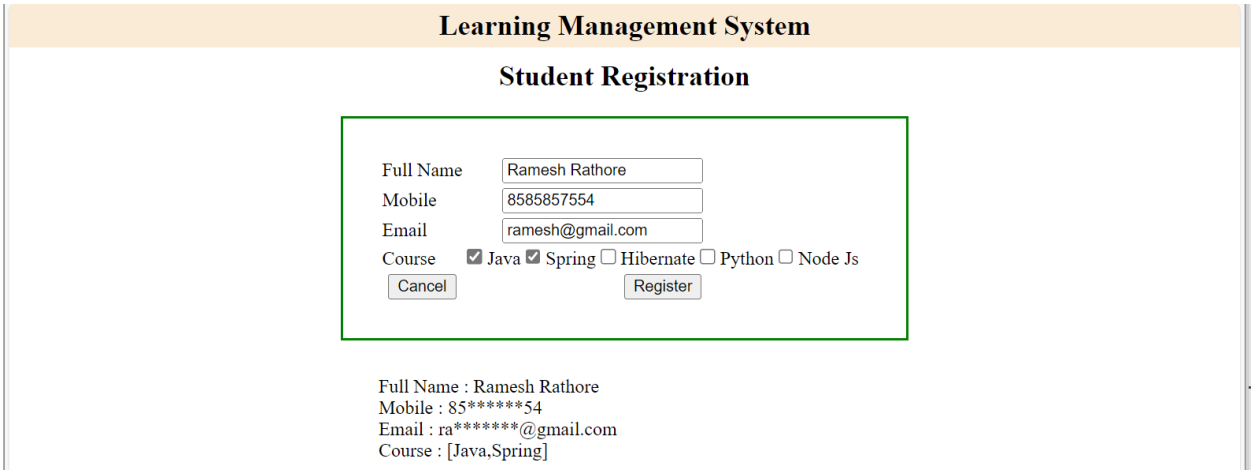
Note: Write a separate java script function for each above validation rules.

Day 13 (9th Aug 2024)

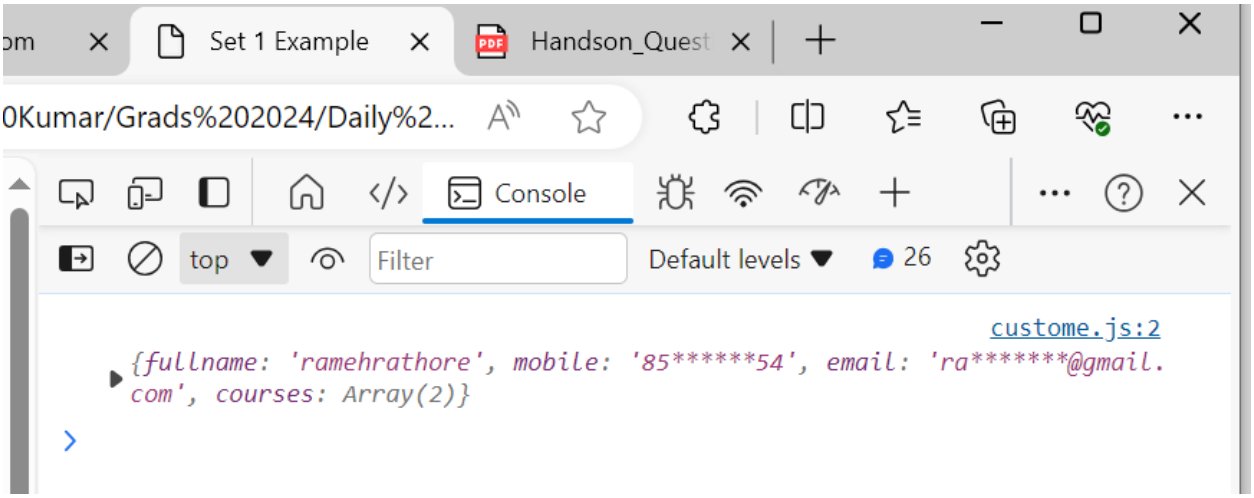
Set 1

Note: Consider the same web page given in day 12 assignment for solving these assignments.

1. When the user click on Register button , display the details in same web page as shown below if all details are validated to true



2. Construct a JSON object with data given in the form and print to browser console as shown below .



Day 14: (14th Aug 2024)

Set 1:

1. Write the Junit Test cases for methods implemented in day 4 , day 6 and day 7 assignment