**ASSIGNMENT**

**HOF and Functional Programming**

1. Reverse String. The problem is to create a program that uses the setTimeout() function to reverse a given string after a delay of 2 seconds. The program should use a variable “input” storing a string as input and then implement a delay of 2 seconds before reversing the string. The reversed string should then be printed as output.

import java.util.Timer;

import java.util.TimerTask;

public class ReverseString {

public static void main(String[] args) {

String input = "hello world";

Timer timer = new Timer();

timer.schedule(new TimerTask() {

@Override

public void run() {

String reversed = new StringBuilder(input).reverse().toString();

System.out.println(reversed);

}

}, 2000);

}

}

OUTPUT

dlroW olleH

1. Random Number Generator with Delay and Progress Indication:  The goal of this program is to generate a random number after a delay of 3 seconds, and store the delay in a variable so can be modified. The program displays a message every second indicating the time remaining until the random number is generated and then outputs the generated number.

import java.util.concurrent.TimeUnit;

public class RandomNumberGenerator {

public static void main(String[] args) throws InterruptedException {

int delay = 3; // delay in seconds

int progress = delay;

while (progress > 0) {

System.out.println("Generating random number in " + progress + " seconds...");

TimeUnit.SECONDS.sleep(1);

progress--;

}

int randomNumber = (int) (Math.random() \* 100);

System.out.println("Random number generated: " + randomNumber);

}

}

OUTPUT

Generating random number in 3 seconds...

Generating random number in 2 seconds...

Generating random number in 1 seconds...

Random number generated: 43

1. Build a feature for Store's Inventory. Suppose a store has a list of items and their prices in US Dollars stored as an object. Create a JavaScript program to convert the prices to Indian Rupees using an exchange rate of 1 USD to 80 INR. The program should use the map higher-order function to create a new object with the converted prices in Rupees

import java.util.HashMap;

import java.util.Map;

public class StoreInventory {

public static void main(String[] args) {

Map<String, Double> inventory = new HashMap<>();

inventory.put("item1", 10.50);

inventory.put("item2", 20.75);

inventory.put("item3", 5.99);

Map<String, Double> convertedInventory = new HashMap<>();

inventory.forEach((item, price) -> {

double convertedPrice = price \* 80;

convertedInventory.put(item, convertedPrice);

});

System.out.println("Original Inventory: " + inventory);

System.out.println("Converted Inventory: " + convertedInventory);

}

}

OUTPUT

Original Inventory: {item2=20.75, item1=10.5, item3=5.99}

Converted Inventory: {item2=1660.0, item1=840.0, item3=479.20000000000005}

1. Filtering and Capitalizing: Books Published After 2010 with Author Names. Write a program that takes a list of books, including their authors and publication years as input. The program should then filter out all books that were published before 2010 and create a new array with the remaining books, but with their author names capitalized

import java.util.ArrayList;

import java.util.List;

public class BookFilter {

public static void main(String[] args) {

List<Book> books = new ArrayList<>();

books.add(new Book("Author1", "Book1", 2009));

books.add(new Book("Author2", "Book2", 2015));

books.add(new Book("Author3", "Book3", 2011));

books.add(new Book("Author4", "Book4", 2008));

List<Book> filteredBooks = new ArrayList<>();

books.forEach(book -> {

if (book.getYear() >= 2010) {

book.setAuthor(book.getAuthor().toUpperCase());

filteredBooks.add(book);

}

});

System.out.println("Original Books: " + books);

System.out.println("Filtered Books: " + filteredBooks);

}

}

class Book {

private String author;

private String title;

private int year;

public Book(String author, String title, int year) {

this.author = author;

this.title = title;

this.year = year;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public String getTitle() {

return title;

}

public int getYear() {

return year;

}

@Override

public String toString() {

return title + " by " + author + " (" + year + ")";

}

}

OUTPUT

Original Books: [Book1 by Author1 (2009), Book2 by AUTHOR2 (2015), Book3 by AUTHOR3 (2011), Book4 by Author4 (2008)]

Filtered Books: [Book2 by AUTHOR2 (2015), Book3 by AUTHOR3 (2011)]

1. URL validation. Write a program using a regex that matches valid URLs. Valid URLs should start with either http:// or https://, followed by one or more letters, digits, or special characters, followed by a dot, followed by one or more letters. Print a message indicating if the input matches the conditions or not.

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class URLValidator {

public static void main(String[] args) {

String[] urls = {"http://www.google.com", "https://www.yahoo.com", "ftp://example.com",

"https://stackoverflow.com/questions/ask", "http://www.amazon.in?search=books"};

String regex = "^https?://([\\w\\d\\-]+\\.)+[\\w\\d]{2,}$";

Pattern pattern = Pattern.compile(regex);

for (String url : urls) {

Matcher matcher = pattern.matcher(url);

if (matcher.matches()) {

System.out.println(url + " is a valid URL.");

} else {

System.out.println(url + " is not a valid URL.");

}

}

}

}

OUTPUT

http://www.google.com is a valid URL.

https://www.yahoo.com is a valid URL.

ftp://example.com is not a valid URL.

https://stackoverflow.com/questions/ask is not a valid URL.

http://www.amazon.in?search=books is not a valid URL.

1. LinkedIn Profile URL Validator. As a developer, you want to create a program that validates LinkedIn profile URLs to ensure that they are formatted correctly and contain only valid characters. Valid LinkedIn profile URLs should start with https:// www.linkedin.com/in/ followed by a combination of one or more letters, digits, underscores, or hyphens, and end with a letter or digit. The length of the profile ID should be between 5 and 30 characters. The program should use a regular expression to match valid LinkedIn profile URLs, and should not match URLs that do not follow this format or contain invalid characters. The program should provide clear output messages indicating whether each input is a valid LinkedIn profile URL or not.

import java.util.Scanner;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class LinkedInProfileValidator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a LinkedIn profile URL: ");

String url = scanner.nextLine();

Pattern pattern = Pattern.compile("^https://www.linkedin.com/in/[a-zA-Z0-9\_-]{5,30}[a-zA-Z0-9]$");

Matcher matcher = pattern.matcher(url);

if(matcher.matches()) {

System.out.println("Valid LinkedIn profile URL");

} else {

System.out.println("Invalid LinkedIn profile URL");

}

scanner.close();

}

}

OUTPUT

Enter a LinkedIn profile URL: https://www.linkedin.com/in/piyush-kumar-705165250

Valid LinkedIn profile URL