**WEEK 1**

**ALGORITHMS\_DATA STRUCTURE**

**Exercise 2: E-commerce Platform Search Function**

import java.util.\*;

class Product {

    int productId;

    String productName;

    String category;

    public Product(int productId, String productName, String category) {

        this.productId = productId;

        this.productName = productName;

        this.category = category;

    }

    @Override

    public String toString() {

        return "Product ID: " + productId + ", Name: " + productName + ", Category: " + category;

    }

}

class SearchFunctions {

    public static int searchByName(Product[] products, String name) {

        for (int i = 0; i < products.length; i++) {

            if (products[i].productName.equalsIgnoreCase(name)) {

                return i;

            }

        }

        return -1;

    }

    public static int searchById(Product[] products, int id) {

        for (int i = 0; i < products.length; i++) {

            if (products[i].productId == id) {

                return i;

            }

        }

        return -1;

    }

    public static List<Product> searchByCategory(Product[] products, String category) {

        List<Product> results = new ArrayList<>();

        for (Product product : products) {

            if (product.category.equalsIgnoreCase(category)) {

                results.add(product);

            }

        }

        return results;

    }

}

public class ECommerceSearch {

    public static void main(String[] args) {

        Product[] products = {

            new Product(101, "Laptop", "Electronics"),

            new Product(102, "Shoes", "Footwear"),

            new Product(103, "Mobile", "Electronics"),

            new Product(104, "Shirt", "Clothing"),

            new Product(105, "Watch", "Accessories")

        };

        Scanner scanner = new Scanner(System.in);

        while (true) {

            System.out.println("\n--- Search Menu ---");

            System.out.println("1. Search by Name");

            System.out.println("2. Search by Product ID");

            System.out.println("3. Search by Category");

            System.out.println("4. Exit");

            System.out.print("Enter choice: ");

            int choice = -1;

            try {

                choice = Integer.parseInt(scanner.nextLine());

            } catch (Exception e) {

                System.out.println("Invalid input. Try again.");

                continue;

            }

            switch (choice) {

                case 1:

                    System.out.print("Enter product name: ");

                    String name = scanner.nextLine();

                    int nameIndex = SearchFunctions.searchByName(products, name);

                    if (nameIndex != -1) {

                        System.out.println("Product found → " + products[nameIndex]);

                    } else {

                        System.out.println("Product not found.");

                    }

                    break;

                case 2:

                    System.out.print("Enter product ID: ");

                    int id;

                    try {

                        id = Integer.parseInt(scanner.nextLine());

                    } catch (Exception e) {

                        System.out.println("Invalid ID. Please enter a number.");

                        continue;

                    }

                    int idIndex = SearchFunctions.searchById(products, id);

                    if (idIndex != -1) {

                        System.out.println("Product found → " + products[idIndex]);

                    } else {

                        System.out.println("Product ID not found.");

                    }

                    break;

                case 3:

                    System.out.print("Enter category: ");

                    String category = scanner.nextLine();

                    List<Product> matches = SearchFunctions.searchByCategory(products, category);

                    if (matches.isEmpty()) {

                        System.out.println("No products found in this category.");

                    } else {

                        System.out.println("Products in category '" + category + "':");

                        for (Product p : matches) {

                            System.out.println("→ " + p);

                        }

                    }

                    break;

                case 4:

                    System.out.println("Exiting. Thank you!");

                    scanner.close();

                    return;

                default:

                    System.out.println("Invalid choice. Try again.");

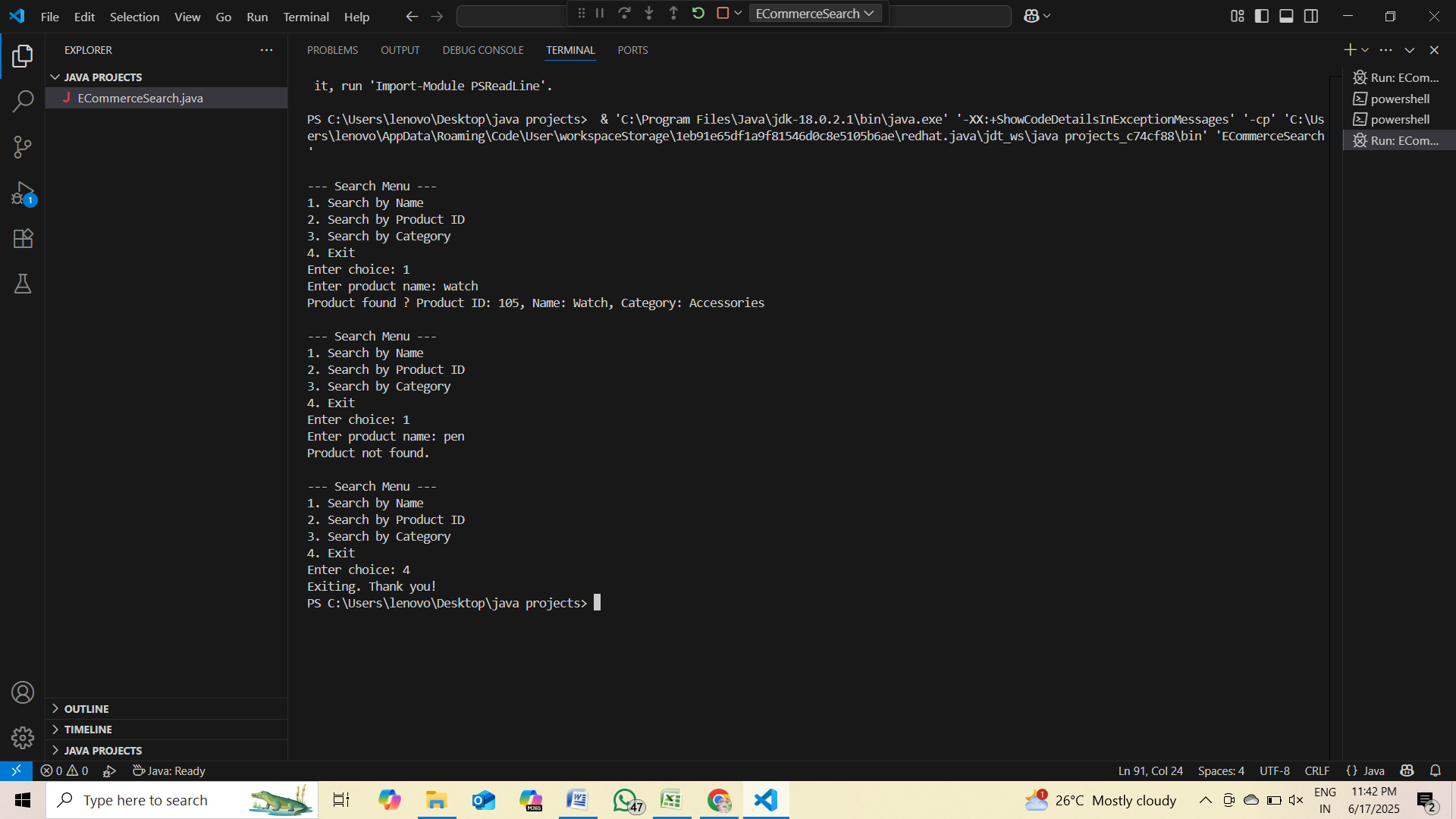
            }

        }

    }

}

**OUTPUT**

****

**Exercise 7: Financial Forecasting**

public class FinancialForecast {

    // Recursive method to calculate future value

    public static double futureValue(double principal, double rate, int years) {

        if (years == 0)

            return principal;

        return futureValue(principal, rate, years - 1) \* (1 + rate);

    }

    public static void main(String[] args) {

        double principal = 1000.0;  // Initial investment

        double rate = 0.05;         // 5% annual growth rate

        int years = 5;              // Forecasting for 5 years

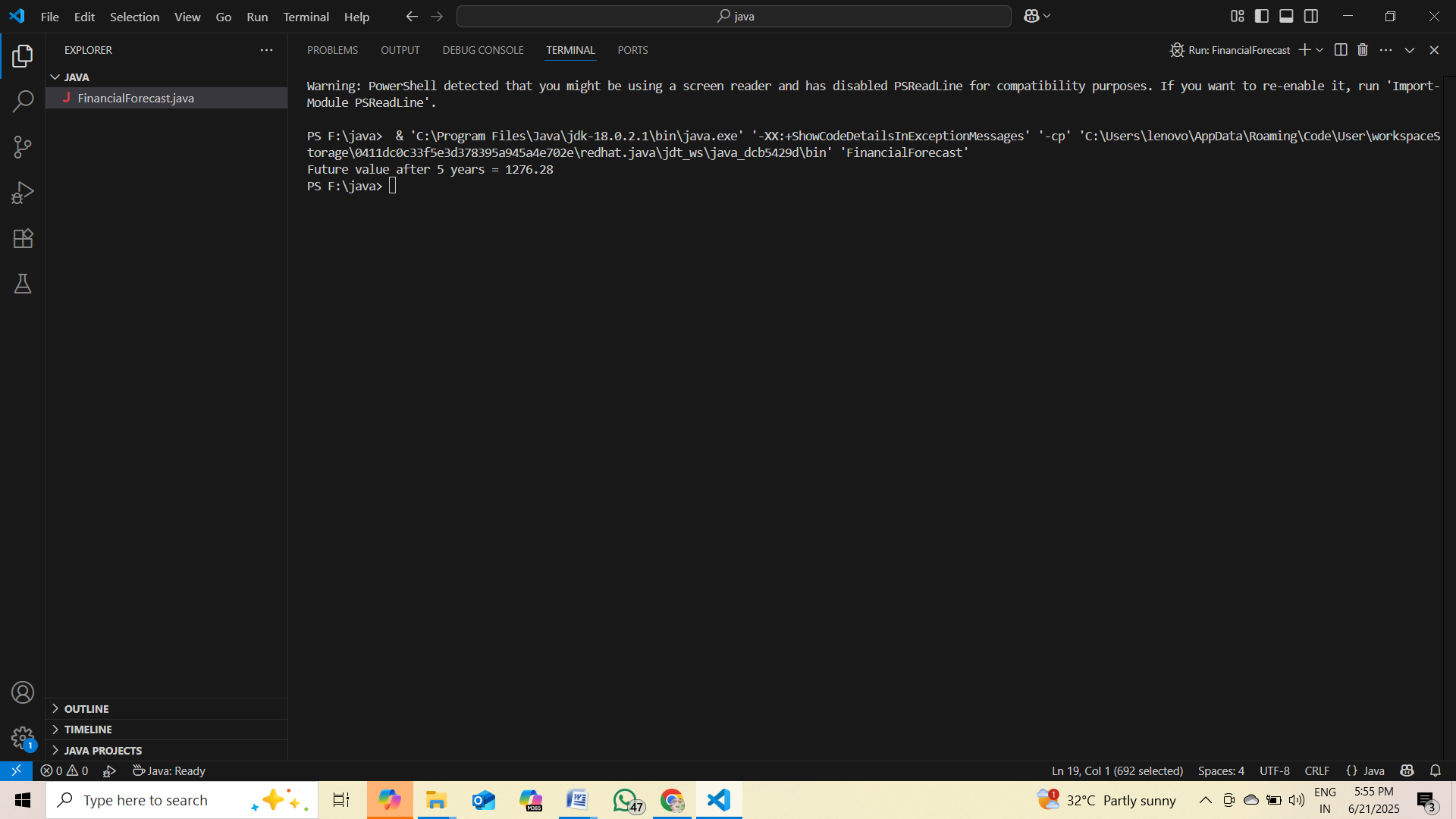
        double result = futureValue(principal, rate, years);

        System.out.printf("Future value after %d years = %.2f\n", years, result);

    }

}

**OUTPUT**

****