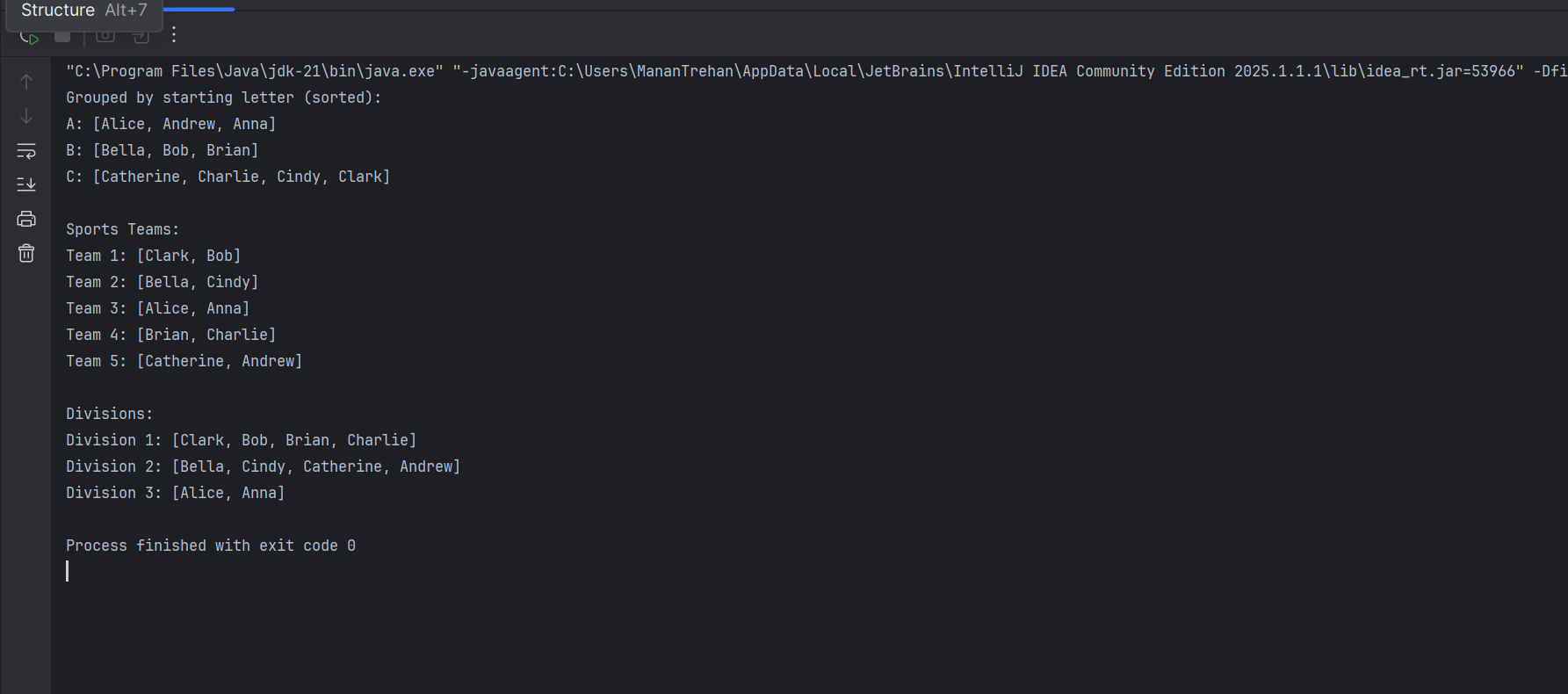
**Java Assignment 3: Collections**

Q1.)

import java.util.\*;  
import java.util.stream.Collectors;  
  
public class Employee {  
 public static void main(String[] args) {  
 // Sample data: Map of departments with list of employee names  
 Map<String, List<String>> departments = new HashMap<>();  
 departments.put("HR", Arrays.*asList*("Alice", "Andrew", "Anna"));  
 departments.put("Engineering", Arrays.*asList*("Brian", "Bella", "Bob"));  
 departments.put("Marketing", Arrays.*asList*("Charlie", "Catherine", "Cindy", "Clark"));  
  
 // Step 1: Combine all employees into one list  
 List<String> allEmployees = departments.values().stream()  
 .flatMap(Collection::stream)  
 .collect(Collectors.*toList*());  
  
 // Step 2-4: Group by first letter, filter and sort  
 Map<Character, List<String>> groupedByInitial = allEmployees.stream()  
 .sorted()  
 .collect(Collectors.*groupingBy*(name -> name.charAt(0), TreeMap::new, Collectors.*toList*()));  
  
 System.*out*.println("Grouped by starting letter (sorted):");  
 groupedByInitial.forEach((ch, names) -> System.*out*.println(ch + ": " + names));  
  
 // Step 5: Create 5 sports teams with randomized employees  
 Collections.*shuffle*(allEmployees);  
 int teamSize = (int) Math.*ceil*(allEmployees.size() / 5.0);  
  
 List<List<String>> sportsTeams = new ArrayList<>();  
 for (int i = 0; i < 5; i++) {  
 int start = i \* teamSize;  
 int end = Math.*min*(start + teamSize, allEmployees.size());  
 sportsTeams.add(new ArrayList<>(allEmployees.subList(start, end)));  
 }  
  
 System.*out*.println("\nSports Teams:");  
 for (int i = 0; i < sportsTeams.size(); i++) {  
 System.*out*.println("Team " + (i + 1) + ": " + sportsTeams.get(i));  
 }  
  
 // Step 6: Merge the 5 sports teams into 3 divisions  
 List<List<String>> divisions = Arrays.*asList*(new ArrayList<>(), new ArrayList<>(), new ArrayList<>());  
  
 for (int i = 0; i < sportsTeams.size(); i++) {  
 divisions.get(i % 3).addAll(sportsTeams.get(i));  
 }  
  
 System.*out*.println("\nDivisions:");  
 for (int i = 0; i < divisions.size(); i++) {  
 System.*out*.println("Division " + (i + 1) + ": " + divisions.get(i));  
 }  
 }  
}



Q2.)

import java.util.\*;  
  
// Product class  
class Product {  
 private final String productId;  
 private String name;  
 private String category;  
 private double price;  
  
 public Product(String productId, String name, String category, double price) {  
 this.productId = productId;  
 this.name = name;  
 this.category = category;  
 this.price = price;  
 }  
  
 public String getProductId() { return productId; }  
 public String getName() { return name; }  
 public String getCategory() { return category; }  
 public double getPrice() { return price; }  
  
 public void setName(String name) { this.name = name; }  
 public void setCategory(String category) { this.category = category; }  
 public void setPrice(double price) { this.price = price; }  
  
 // Use productId to determine uniqueness  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (!(o instanceof Product)) return false;  
 Product product = (Product) o;  
 return productId.equals(product.productId);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(productId);  
 }  
  
 @Override  
 public String toString() {  
 return "[" + productId + ", " + name + ", " + category + ", $" + price + "]";  
 }  
}  
  
// Catalogue Manager  
class ProductCatalogue {  
 private final Map<Product, Integer> catalogue = new HashMap<>();  
  
 // Add product  
 public boolean addProduct(Product product, int quantity) {  
 if (catalogue.containsKey(product)) {  
 System.*out*.println("Product already exists!");  
 return false;  
 }  
 catalogue.put(product, quantity);  
 return true;  
 }  
  
 // Retrieve quantity  
 public Integer getQuantity(Product product) {  
 return catalogue.get(product);  
 }  
  
 // Update product details  
 public boolean updateProduct(String productId, String name, String category, double price) {  
 for (Product p : catalogue.keySet()) {  
 if (p.getProductId().equals(productId)) {  
 p.setName(name);  
 p.setCategory(category);  
 p.setPrice(price);  
 return true;  
 }  
 }  
 return false;  
 }  
  
 // Delete product  
 public boolean deleteProduct(String productId) {  
 for (Product p : new HashSet<>(catalogue.keySet())) {  
 if (p.getProductId().equals(productId)) {  
 catalogue.remove(p);  
 return true;  
 }  
 }  
 return false;  
 }  
  
 // Sort by Product ID  
 public List<Map.Entry<Product, Integer>> sortByProductId() {  
 return catalogue.entrySet().stream()  
 .sorted(Comparator.*comparing*(e -> e.getKey().getProductId()))  
 .toList();  
 }  
  
 // Sort by Product Name  
 public List<Map.Entry<Product, Integer>> sortByProductName() {  
 return catalogue.entrySet().stream()  
 .sorted(Comparator.*comparing*(e -> e.getKey().getName()))  
 .toList();  
 }  
  
 // Display catalogue  
 public void displayCatalogue() {  
 catalogue.forEach((product, quantity) ->  
 System.*out*.println(product + " | Quantity: " + quantity));  
 }  
}

// Main Function

public class Main {  
 public static void main(String[] args) {  
 ProductCatalogue catalogue = new ProductCatalogue();  
  
 Product p1 = new Product("P101", "Laptop", "Electronics", 1200.00);  
 Product p2 = new Product("P102", "Mouse", "Accessories", 25.99);  
 Product p3 = new Product("P103", "Keyboard", "Accessories", 45.00);  
  
 catalogue.addProduct(p1, 10);  
 catalogue.addProduct(p2, 50);  
 catalogue.addProduct(p3, 30);  
  
 System.*out*.println("\n-- Sorted by Product ID --");  
 catalogue.sortByProductId().forEach(e -> System.*out*.println(e.getKey() + " | Qty: " + e.getValue()));  
  
 System.*out*.println("\n-- Sorted by Product Name --");  
 catalogue.sortByProductName().forEach(e -> System.*out*.println(e.getKey() + " | Qty: " + e.getValue()));  
  
 // Update example  
 catalogue.updateProduct("P102", "Wireless Mouse", "Accessories", 29.99);  
  
 // Delete example  
 catalogue.deleteProduct("P103");  
  
 System.*out*.println("\n-- Catalogue After Updates --");  
 catalogue.displayCatalogue();  
 }  
}

