**CORE JAVA ASSIGNMENT 5 – GENERICS**

**Assignment on Generics**

1. Use a HashSet to hold Employee Objects. Upon running the application, the details of the

employees added to the HashSet should be displayed.

Employee <<class>>

|-- id

|-- name

|-- salary

|-- department

|-- displayDetails()

Feel free to add properties and methods to Employee Class

Note: if we try to store any object other than Employee Object in HashSet, we should not be allowed to.

**Solution:**

**Employee.java**package com.GenericsAssignment;  
  
public class Employee {  
 private long id;  
 private String name, department;  
 private double salary;  
  
 public Employee(long id, String name, String department, double salary) {  
 this.id = id;  
 this.name = name;  
 this.department = department;  
 this.salary = salary;  
 }  
  
 public void displayDetails()  
 {  
 System.*out*.println("Emp\_id : " + id);  
 System.*out*.println("Name " + name);  
 System.*out*.println("Salary " + salary);  
 System.*out*.println("Department " + department);  
 }  
  
}

**GenericHashSet.java**package com.GenericsAssignment;  
import java.util.HashSet;  
  
public class GenericHashSet <T extends Employee> {  
 private HashSet<T> employeeHashSet = new HashSet<>();  
 public void setEmployeeHashSet(T e)  
 {  
 employeeHashSet.add(e);  
 System.*out*.println("New Employee is Added! ");  
 }  
 public void getEmployeeDetails()  
 {  
 if(employeeHashSet.isEmpty())  
 {  
 System.*out*.println("Employees Not Found");  
 }  
 else {  
 for (T itr : employeeHashSet) {  
 System.*out*.println("------------------------");  
 itr.displayDetails();  
 }}}}

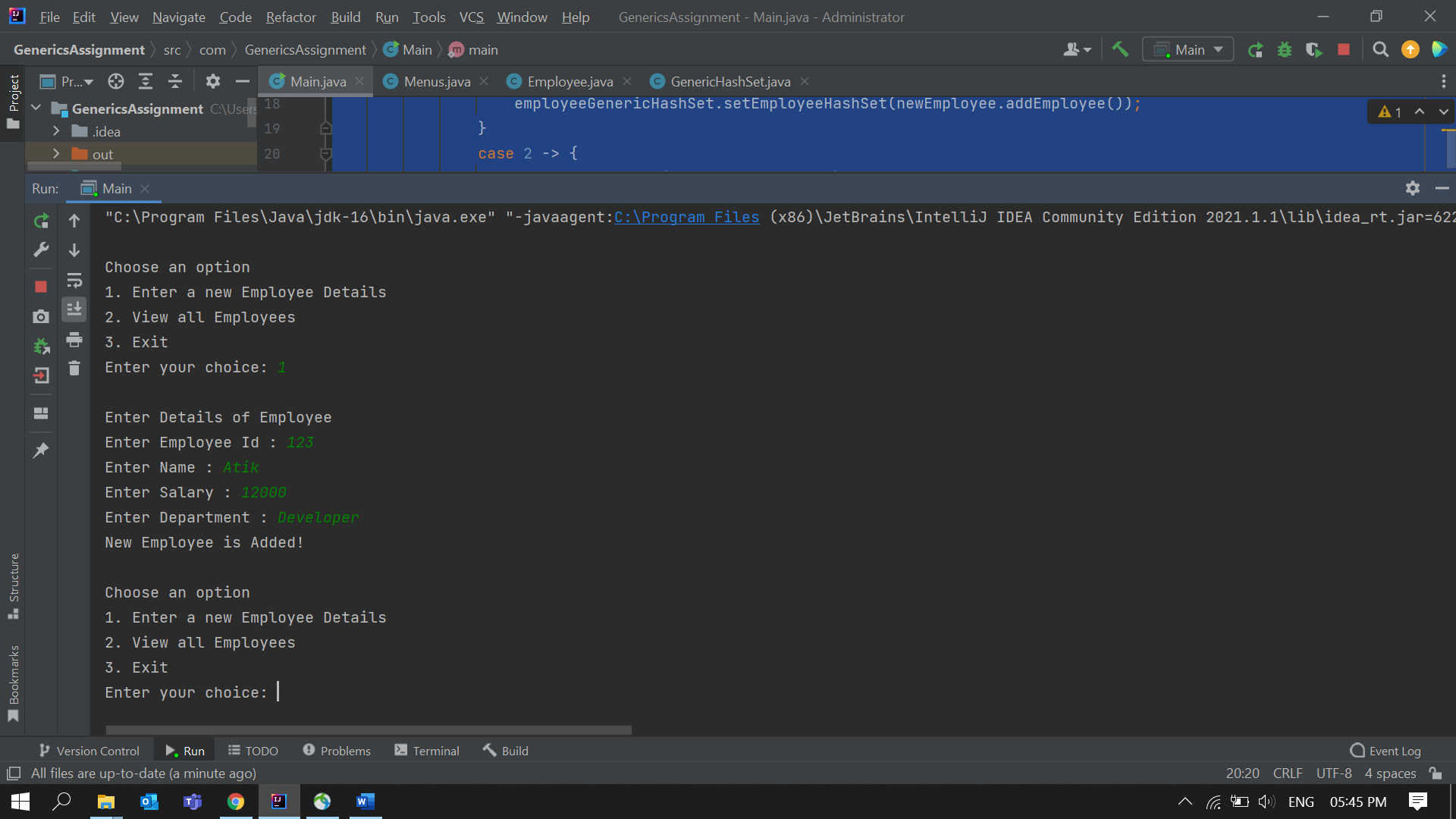
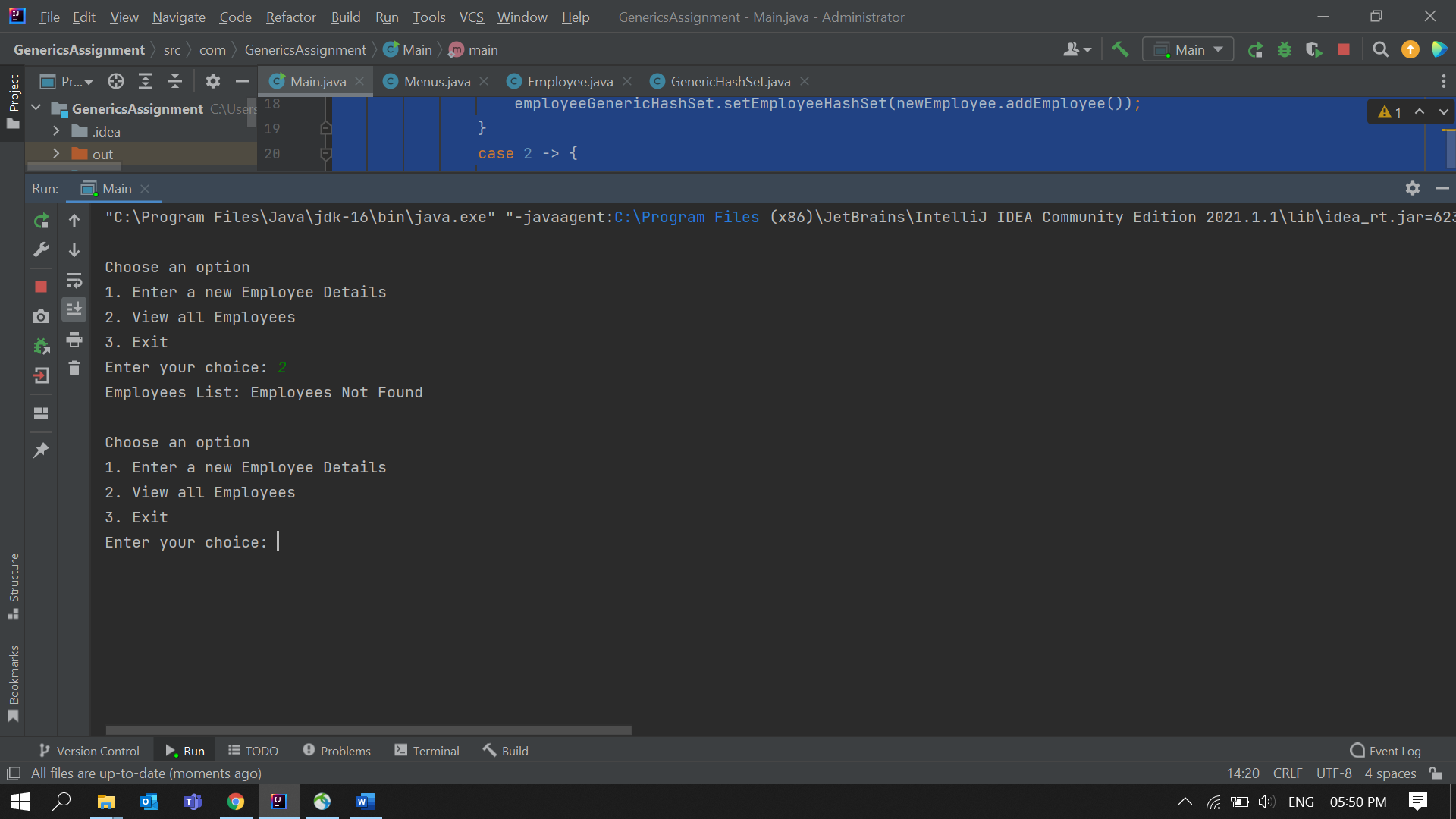
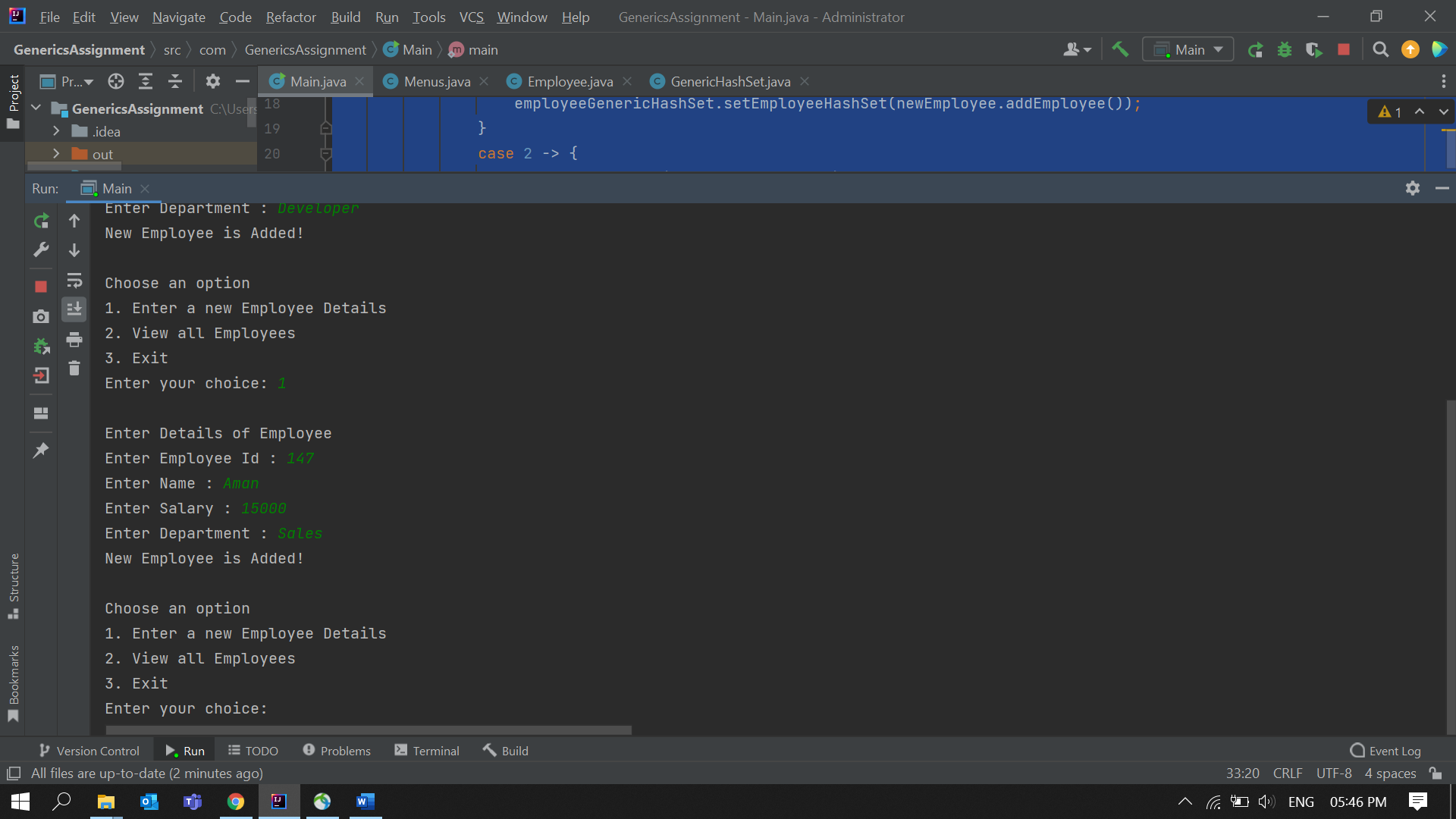
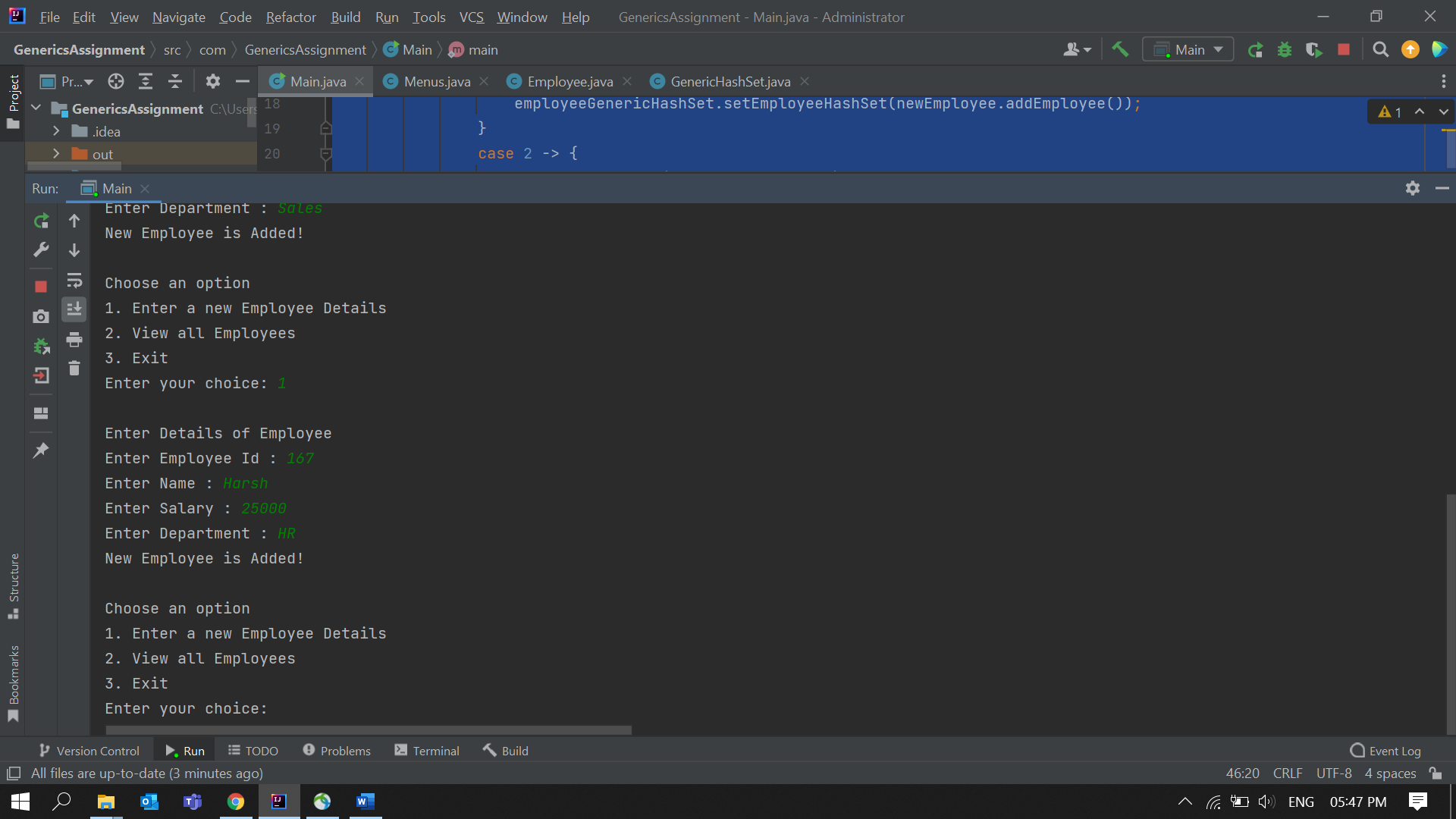
**Main.java**

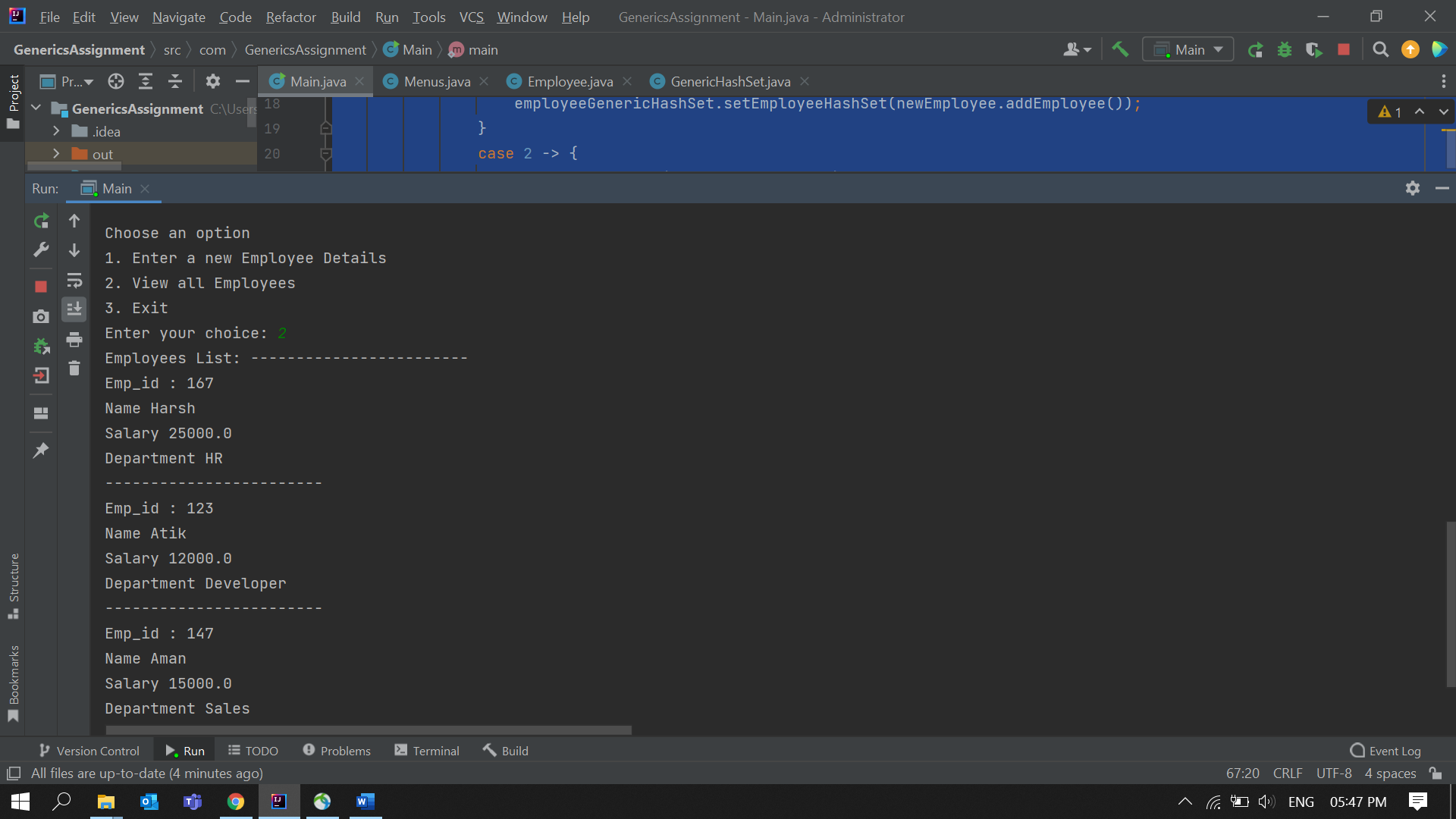
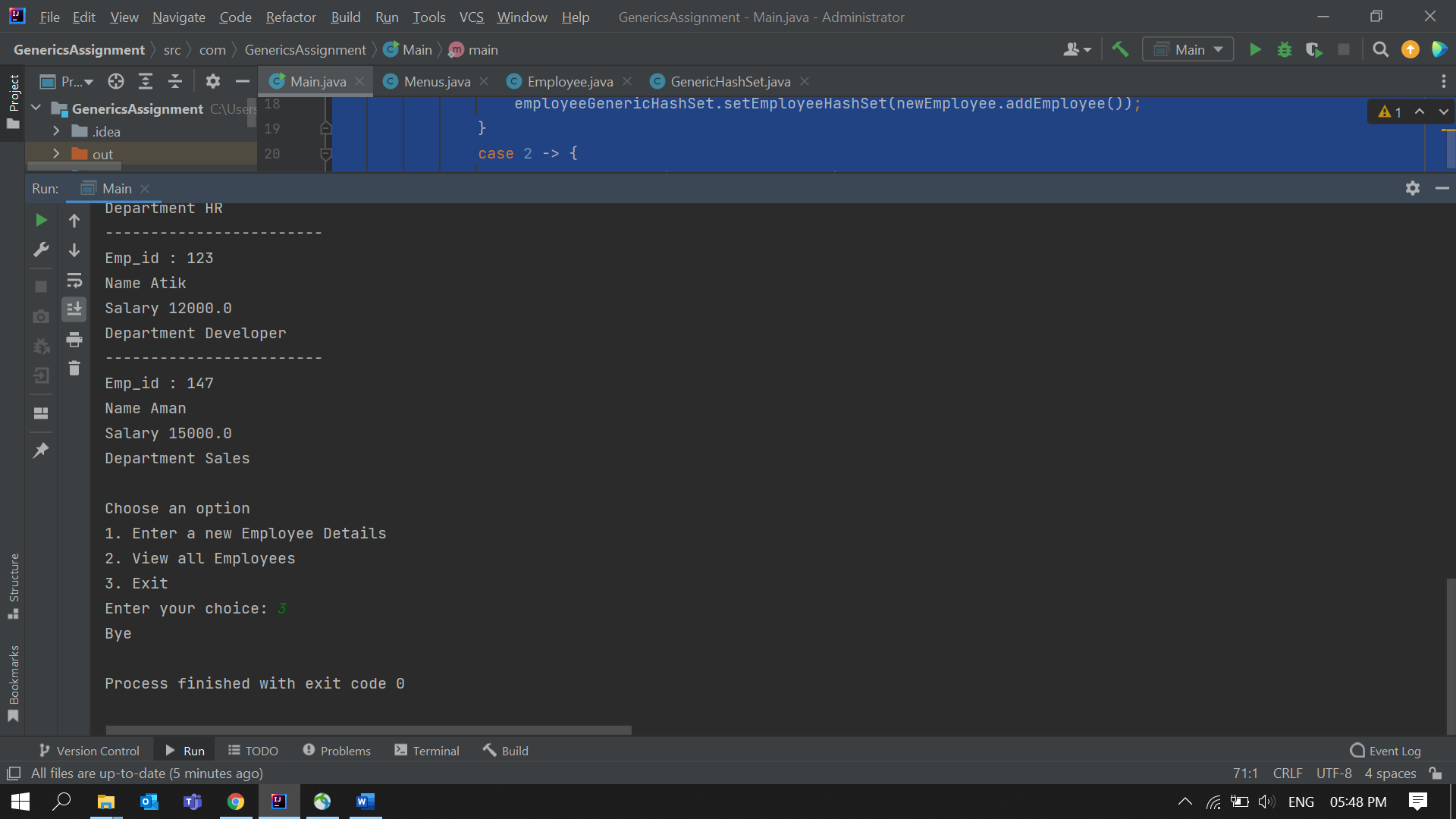
package com.GenericsAssignment;  
import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 GenericHashSet<Employee> employeeGenericHashSet = new GenericHashSet<>();  
 Scanner input = new Scanner(System.*in*);  
 int choice;  
 do{  
 Menus.*displayMenu*();  
 choice = input.nextInt();  
 switch (choice) {  
 case 1 -> {  
 Menus newEmployee = new Menus();  
 employeeGenericHashSet.setEmployeeHashSet(newEmployee.addEmployee());  
 }  
 case 2 -> {  
 System.*out*.print("Employees List: ");  
 employeeGenericHashSet.getEmployeeDetails();  
 }  
 case 3 -> System.*out*.println("Bye");  
 default -> System.*out*.println("Invalid input");  
 }  
 }while(choice != 3);  
 }  
}

**Menus.java**

package com.GenericsAssignment;  
import java.util.Scanner;  
public class Menus {  
  
 Employee newEmployee;  
 Scanner input = new Scanner(System.*in*);  
 long id;  
 String name, department;  
 double salary;  
 public static void displayMenu()  
 {  
 System.*out*.println("\nChoose an option");  
 System.*out*.println("1. Enter a new Employee Details");  
 System.*out*.println("2. View all Employees");  
 System.*out*.println("3. Exit");  
 System.*out*.print("Enter your choice: ");  
 }  
 public Employee addEmployee()  
 {  
 System.*out*.print("\nEnter Details of Employee");  
 System.*out*.print("\nEnter Employee Id : ");  
 id = input.nextLong();  
 System.*out*.print("Enter Name : ");  
 name = input.next();  
 System.*out*.print("Enter Salary : ");  
 salary = input.nextDouble();  
 System.*out*.print("Enter Department : ");  
 department = input.next();  
 newEmployee = new Employee(id,name,department,salary);  
 return newEmployee;  
 }  
  
}

**Output:**



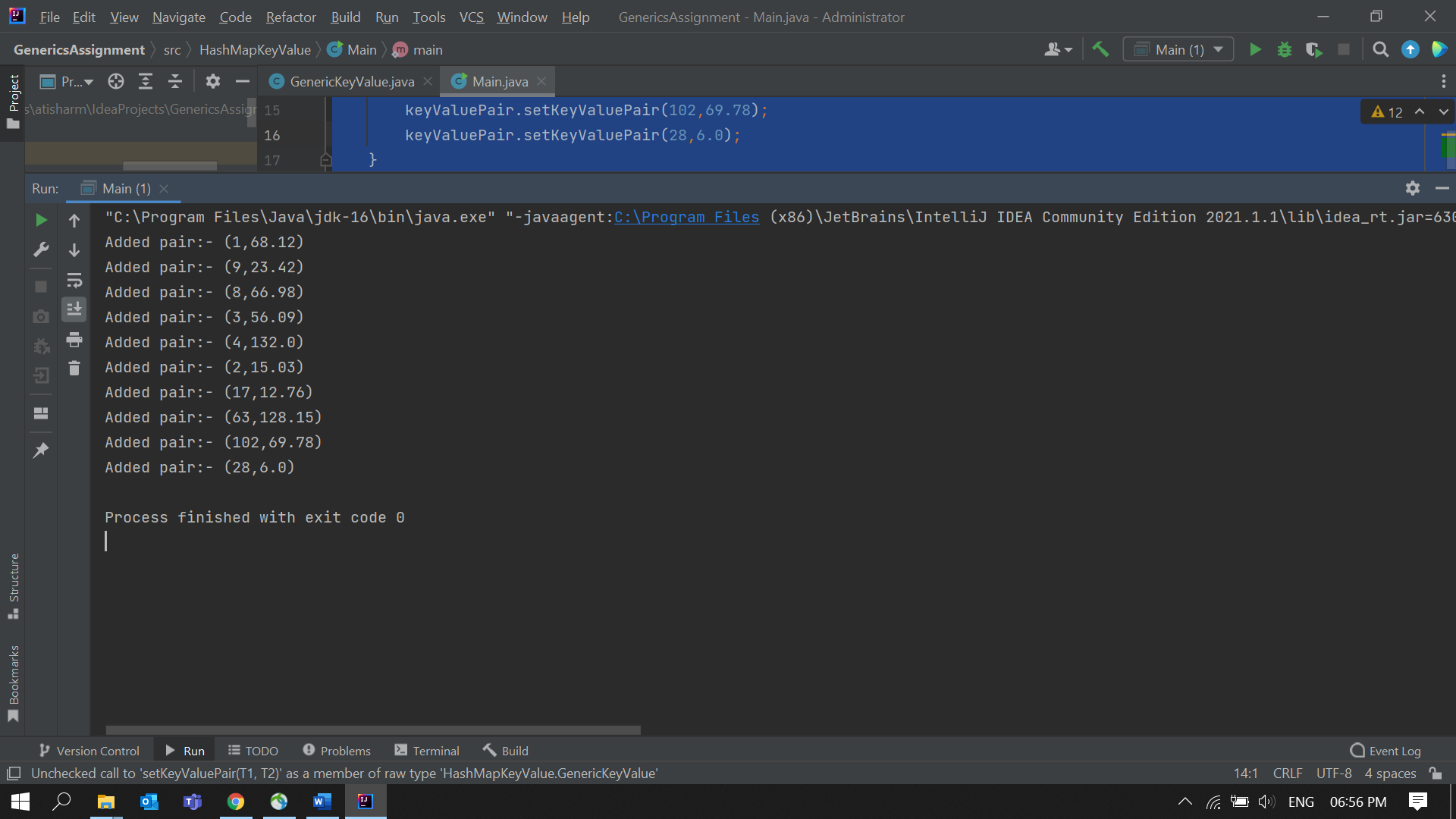
2. Write an application to hold 10 random int values as keys and 10 random double values as values for a HashMap. Print the data store in the HashMap. Note: Keys can only be int and values double.

**Solution:**

**GenericKeyValue.java**package HashMapKeyValue;  
  
import java.util.HashMap;  
  
public class GenericKeyValue <T1 extends Integer , T2 extends Double> {  
 HashMap<T1,T2> keyValuePair = new HashMap<>();  
  
 public void setKeyValuePair(T1 key,T2 value) {  
 System.*out*.println("Added pair:- "+ "(" + key + "," + value + ")" );  
 keyValuePair.put(key,value);  
 }  
}

**Main.java**package HashMapKeyValue;  
  
public class Main {  
 public static void main(String[] args) {  
 GenericKeyValue keyValuePair = new GenericKeyValue();  
  
 keyValuePair.setKeyValuePair(1,68.12);  
 keyValuePair.setKeyValuePair(9,23.42);  
 keyValuePair.setKeyValuePair(8,66.98);  
 keyValuePair.setKeyValuePair(3,56.09);  
 keyValuePair.setKeyValuePair(4,132.0);  
 keyValuePair.setKeyValuePair(2,15.03);  
 keyValuePair.setKeyValuePair(17,12.76);  
 keyValuePair.setKeyValuePair(63,128.15);  
 keyValuePair.setKeyValuePair(102,69.78);  
 keyValuePair.setKeyValuePair(28,6.0);  
 }  
}

**Output:**

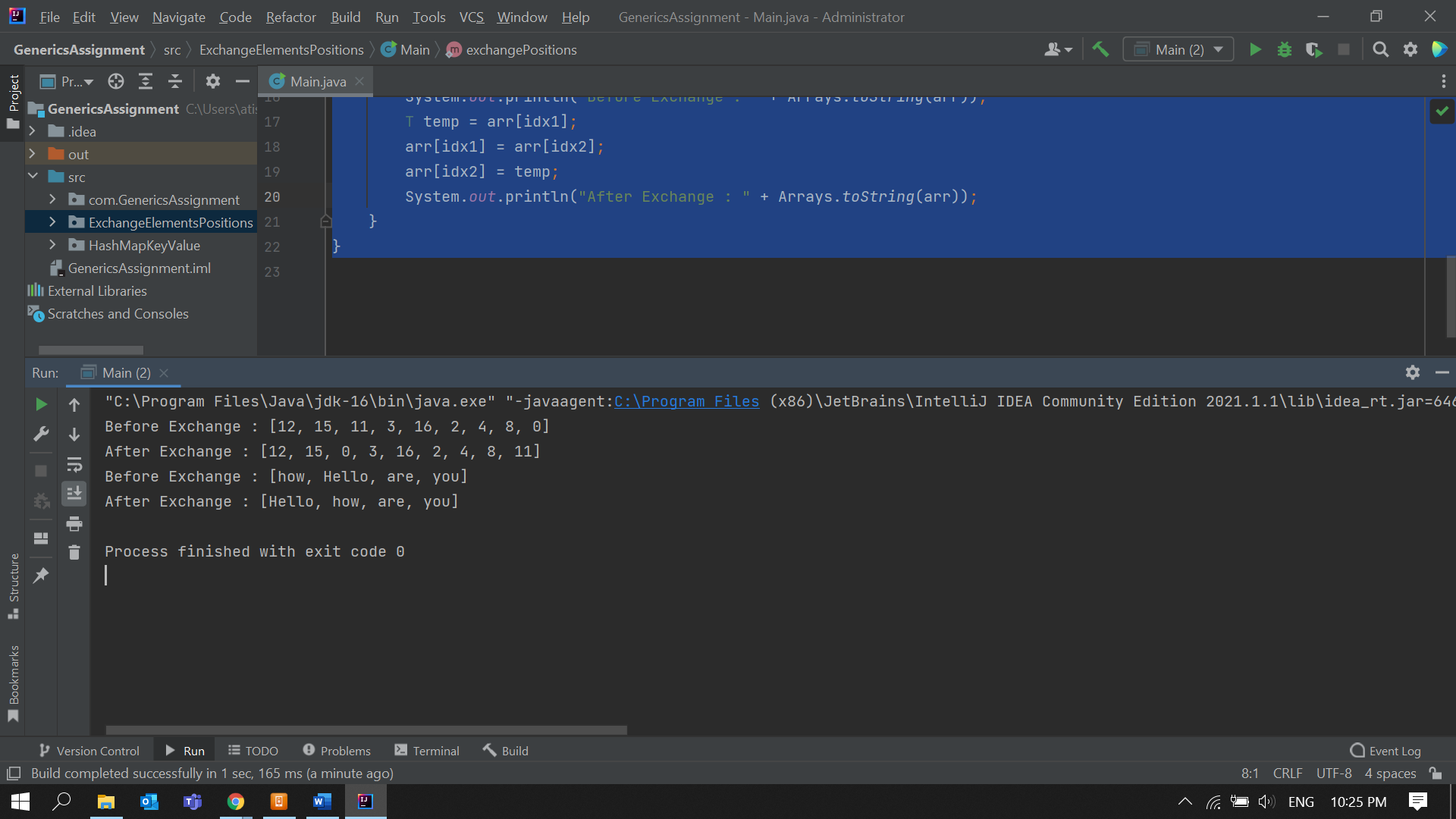


3. Write a generic method to exchange the positions of two different elements in an array.

**Solution:  
  
Main.java**

package ExchangeElementsPositions;  
import java.util.Arrays;  
  
public class Main {  
 public static void main(String[] args) {  
 Integer[] arr = {12 ,15,11, 3,16,2,4,8,0};  
 *exchangePositions*(arr,2,arr.length-1);  
  
 String[] str = {"how" , "Hello", "are" , "you" };  
 *exchangePositions*(str, 1, 0 );  
  
 }  
 public static <T> void exchangePositions(T[] arr, int idx1 , int idx2)  
 {  
 System.*out*.println("Before Exchange : " + Arrays.*toString*(arr));  
 T temp = arr[idx1];  
 arr[idx1] = arr[idx2];  
 arr[idx2] = temp;  
 System.*out*.println("After Exchange : " + Arrays.*toString*(arr));  
 }  
}

**Output:**



4) Design a class named Pair which has two properties. The name of the first property is key and that of the second property is value. When designing the class take case of the following scenarios:

a. Create an Object of Pair class to store String value for the property key and String value for the property value. Restriction Apart from String type no other types should be acceptable as key or value input

e.g.

myObj.setKey("1");  
myObj.setValue("Hello");

b. Create an object of the class Pair to store String value for the property key and java.util.Date as value for the property value

myObj.setKey("Today is");  
myObj.setValue(new java.util.Date());

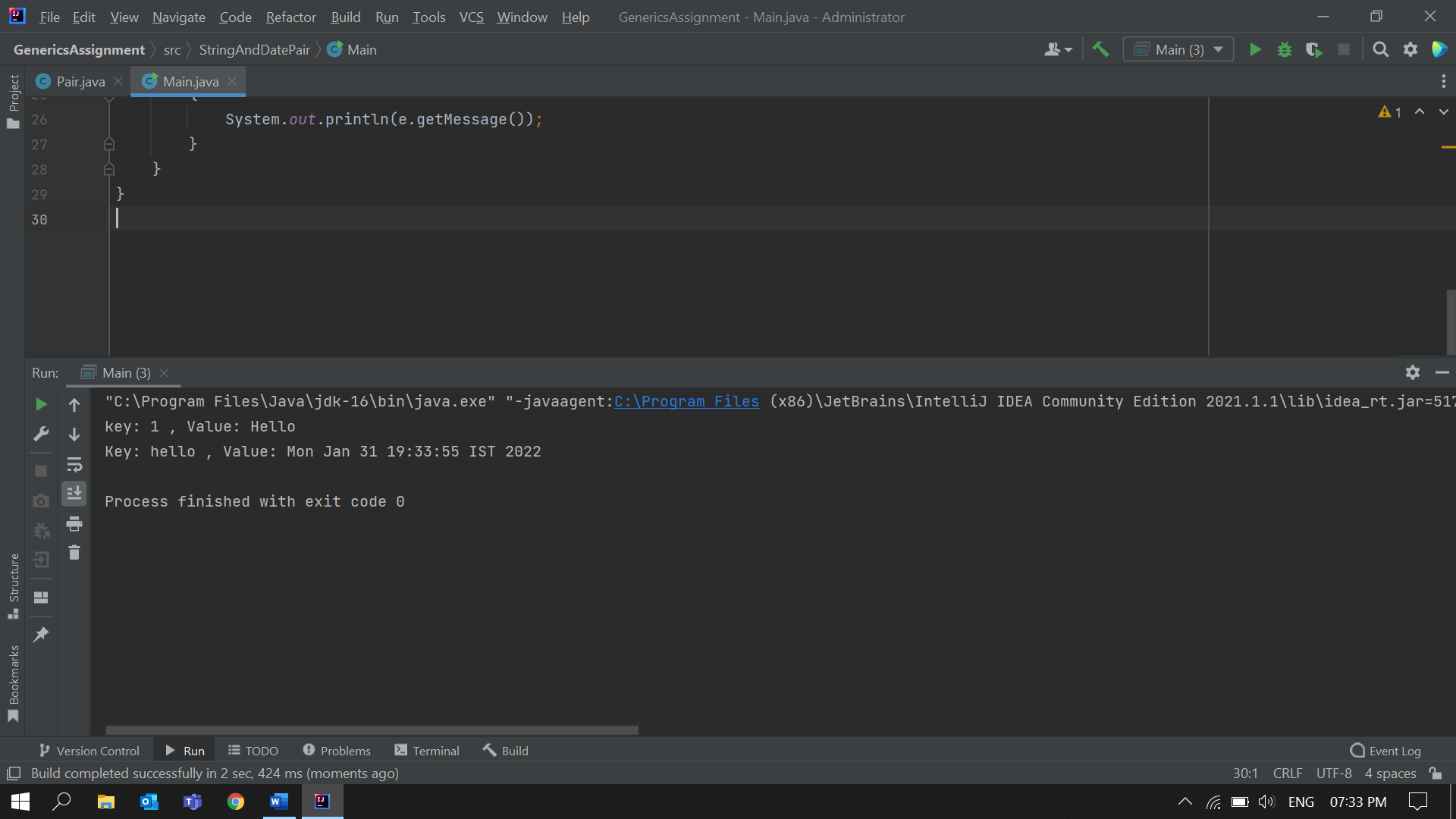
Note: In scenario a. no data apart from String should be used for key and value, in scenario b. no data apart from String for key and java.util.Date should be allowed

**Solution:  
  
Main.java**

package StringAndDatePair;  
import java.util.Date;  
public class Main {  
 public static void main(String[] args) {  
  
 try {  
 Pair<String> obj1 = new Pair<>();  
 obj1.setKey("1");  
 obj1.setValue("Hello");  
 System.*out*.println("key: " + obj1.getKey()+ " , Value: " + obj1.getValue());  
  
 Pair<Date> obj2 = new Pair<>();  
 obj2.setKey("hello");  
 obj2.setValue(new java.util.Date());  
 System.*out*.println("Key: " + obj2.getKey()+ " , Value: " + obj2.getValue());  
   
 }  
 catch(Exception e)  
 {  
 System.*out*.println(e.getMessage());  
 }  
 }  
}

**Pair.java**  
package StringAndDatePair;  
  
import com.sun.jdi.InvalidTypeException;  
  
import java.util.Date;  
  
public class Pair <T extends Comparable<T>>{  
 String key;  
 T value;  
  
 public String getKey() {  
 return key;  
 }  
  
 public void setKey(String key) {  
 this.key = key;  
 }  
  
 public T getValue() {  
 return value;  
 }  
  
 public void setValue(T value) throws Exception {  
 if(value.getClass() == String.class || value.getClass() == java.util.Date.class)  
 {  
 this.value = value;  
 }  
 else{  
 throw new InvalidTypeException("Only accept String or Date");  
 }  
 }  
}

**Output:**

 **Trying to add non String and Date data in pair**

Pair<Integer> obj3 = new Pair<>();  
obj3.setKey("text");  
obj3.setValue(123); // exception will be thrown   
System.*out*.println("Key: " + obj3.getKey()+ " , Value: " + obj3.getValue());

**Output:**

