**Day 20**

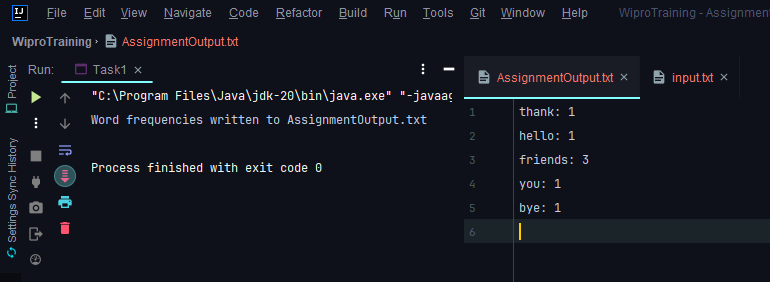
**Task 1: Java IO Basics**

Write a program that reads a text file and counts the frequency of each word using FileReader and FileWriter.

**Program:**

***package* Assignments.Day20;  
  
*import* java.io.\*;  
*import* java.util.\*;  
  
*public class* Task1 {  
 *public static void* main(String[] args) {  
 String inputFilePath = "input.txt"; *// input text file* String outputFilePath = "AssignmentOutput.txt"; *// output file  
  
 try* (BufferedReader reader = *new* BufferedReader(*new* FileReader(inputFilePath));  
 BufferedWriter writer = *new* BufferedWriter(*new* FileWriter(outputFilePath))) {  
  
 *Map*<String, Integer> wordFrequencyMap = *new* HashMap<>();  
  
 String line;  
 *while* ((line = reader.readLine()) != *null*) {  
 String[] words = line.split("\\s+"); *// Split by whitespace  
 for* (String word : words) {  
 word = word.toLowerCase(); *// Convert to lowercase* wordFrequencyMap.put(word, wordFrequencyMap.getOrDefault(word, 0) + 1);  
 }  
 }  
  
 *for* (*Map*.*Entry*<String, Integer> entry : wordFrequencyMap.entrySet()) {  
 writer.write(entry.getKey() + ": " + entry.getValue());  
 writer.newLine();  
 }  
  
 System.*out*.println("Word frequencies written to " + outputFilePath);  
 } *catch* (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}**

**Output:**



**Task 2: Serialization and Deserialization**

Serialize a custom object to a file and then deserialize it back to recover the object state.

**Program**

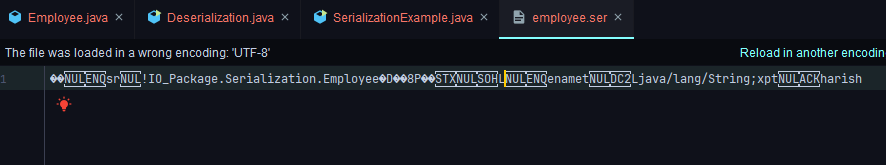
**Step 1: Create a class called Employee**

***package* IO\_Package.Serialization;  
  
*import* java.io.*Serializable*;  
  
*public class* Employee *implements Serializable* {  
 *transient private int* eid;  
 *private* String ename;  
 *public* Employee(*int* eid, String ename){  
 *this*.eid = eid;  
 *this*.ename = ename;  
 }  
  
 @Override  
 *public* String toString() {  
 *return* "EName = "+*this*.ename+" EID = "+*this*.eid;  
 }  
}**

**Step 2:** Create a new class called Serialization which create a “employee.ser” file.

***package* IO\_Package.Serialization;  
  
  
*import* java.io.FileOutputStream;  
*import* java.io.IOException;  
*import* java.io.ObjectOutputStream;  
  
*public class* SerializationExample {  
 *public static void* main(String[] args) *throws* IOException {  
 Employee emp = *new* Employee(101,"harish");  
 ObjectOutputStream oos = *new* ObjectOutputStream(*new* FileOutputStream("employee.ser"));  
 oos.writeObject(emp);  
 System.*out*.println("Employee is Serialized");  
 }  
}**

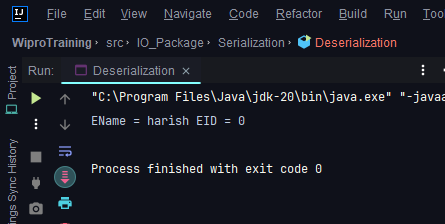
**Step 3:** Check the employee.ser file which is not readable or understandable.

****

**Step 4:** Create a new class called Deserialization which helps in reading the “employee.ser” file

*package* IO\_Package.Serialization;  
 ***import* java.io.FileInputStream;  
*import* java.io.IOException;  
*import* java.io.ObjectInputStream;  
  
*public class* Deserialization {  
 *public static void* main(String[] args) *throws* IOException, ClassNotFoundException {  
 FileInputStream fileInputStream = *new* FileInputStream("Employee.ser");  
 ObjectInputStream objectInputStream = *new* ObjectInputStream(fileInputStream);  
 Object o = objectInputStream.readObject();  
 Employee e1 = (Employee) o;  
 System.*out*.println(e1);  
 }  
}**

**Output:**



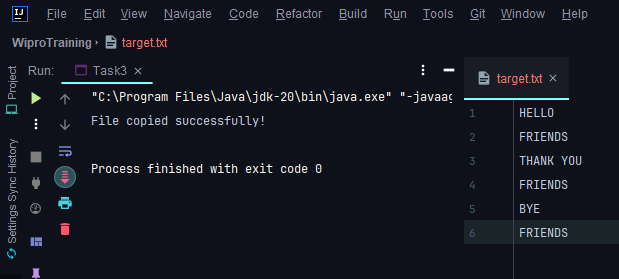
**Task 3: New IO (NIO)**

Use NIO Channels and Buffers to read content from a file and write to another file.

**Program:**

***package* Assignments.Day20;  
  
*import* java.io.IOException;  
*import* java.nio.ByteBuffer;  
*import* java.nio.channels.FileChannel;  
*import* java.nio.file.*Path*;  
*import* java.nio.file.Paths;  
*import* java.nio.file.StandardOpenOption;  
  
*public class* Task3 {  
 *public static void* main(String[] args) {  
 *Path* sourceFilePath = Paths.*get*("input.txt"); *// source file  
 Path* targetFilePath = Paths.*get*("target.txt"); *// target file  
  
 try* (FileChannel sourceChannel = FileChannel.*open*(sourceFilePath, StandardOpenOption.*READ*);  
 FileChannel targetChannel = FileChannel.*open*(targetFilePath, StandardOpenOption.*CREATE*,  
 StandardOpenOption.*WRITE*)) {  
  
 ByteBuffer buffer = ByteBuffer.*allocate*(1024);  
  
 *while* (sourceChannel.read(buffer) != -1) {  
 buffer.flip(); *// Prepare buffer for writing* targetChannel.write(buffer);  
 buffer.clear(); *// Clear buffer for next read* }  
  
 System.*out*.println("File copied successfully!");  
 } *catch* (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}**

**Output:**

****

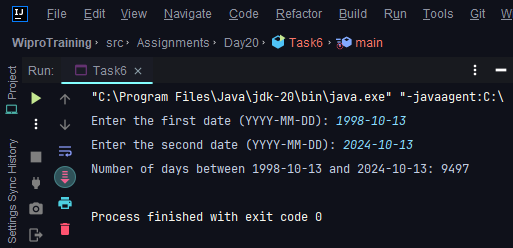
**Task 6: Java 8 Date and Time API**

Write a program that calculates the number of days between two dates input by the user.

**Program:**

***package* Assignments.Day20;  
  
*import* java.time.LocalDate;  
*import* java.time.temporal.ChronoUnit;  
*import* java.util.Scanner;  
  
*public class* Task6 {  
 *public static void* main(String[] args) {  
 Scanner scanner = *new* Scanner(System.*in*);  
  
 System.*out*.print("Enter the first date (YYYY-MM-DD): ");  
 String dateString1 = scanner.nextLine();  
 LocalDate date1 = LocalDate.*parse*(dateString1);  
  
 System.*out*.print("Enter the second date (YYYY-MM-DD): ");  
 String dateString2 = scanner.nextLine();  
 LocalDate date2 = LocalDate.*parse*(dateString2);  
  
 *long* daysBetween = ChronoUnit.*DAYS*.between(date1, date2);  
  
 System.*out*.println("Number of days between " + date1 + " and " + date2 + ": " + daysBetween);  
 }  
}**

**Output:**

****

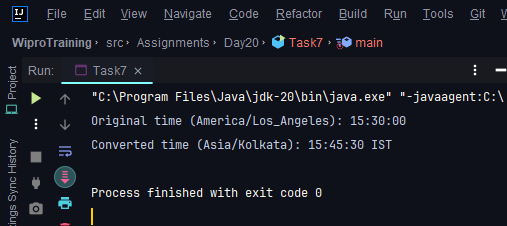
**Task 7: Timezone**

Create a timezone converter that takes a time in one timezone and converts it to another timezone.

**Program:**

***package* Assignments.Day20;  
  
*import* java.time.Instant;  
*import* java.time.ZoneId;  
*import* java.time.ZonedDateTime;  
*import* java.time.format.DateTimeFormatter;  
  
*public class* Task7 {  
  
 *public static void* main(String[] args) {  
   
 String time = "15:30:00";   
 String fromZone = "America/Los\_Angeles";   
 String toZone = "Asia/Kolkata";   
  
 Instant instant = Instant.*parse*("2024-06-09T10:15:30.00Z");  
   
 ZonedDateTime sourceDateTime = ZonedDateTime.*ofInstant*(instant, ZoneId.*of*(fromZone));  
   
 ZonedDateTime targetDateTime = sourceDateTime.withZoneSameInstant(ZoneId.*of*(toZone));  
  
 DateTimeFormatter formatter = DateTimeFormatter.*ofPattern*("HH:mm:ss z");  
   
 String formattedTargetTime = formatter.format(targetDateTime);  
  
 System.*out*.println("Original time (" + fromZone + "): " + time);  
 System.*out*.println("Converted time (" + toZone + "): " + formattedTargetTime);  
 }  
}**

**Output:**

****