CSE 215: Programming Language II Lab

Lab – 12

File I/O

Objective:

- Methods of File Class
- Read and Write operation

File Class

The File class contains the methods for obtaining the properties of a file/directory and for renaming and deleting a file/directory.

```
public class TestFileClass {
  public static void main(String[] args) {
    java.io.File file = new java.io.File("image/us.gif");                create a File
    System.out.println("Does it exist? " + file.exists());               exists()
    System.out.println("The file has " + file.length() + " bytes");       length()
    System.out.println("Can it be read? " + file.canRead());             canRead()
    System.out.println("Can it be written? " + file.canWrite());         canWrite()
    System.out.println("Is it a directory? " + file.isDirectory());      isDirectory()
    System.out.println("Is it a file? " + file.isFile());                isFile()
    System.out.println("Is it absolute? " + file.isAbsolute());          isAbsolute()
    System.out.println("Is it hidden? " + file.isHidden());              isHidden()
    System.out.println("Absolute path is " +
      file.getAbsolutePath());                                           getAbsolutePath()
    System.out.println("Last modified on " +
      new java.util.Date(file.lastModified()));                          lastModified()
  }
}
```

The **`lastModified()`** method returns the date and time when the file was last modified, measured in milliseconds since the beginning of UNIX time (00:00:00 GMT, January 1, 1970).

Writing Data

The **`java.io.PrintWriter`** class can be used to create a file and write data to a text file. First, you have to create a **`PrintWriter`** object for a text file as follows:

```
PrintWriter output = new PrintWriter(filename);
```

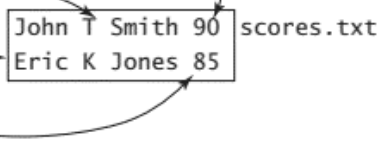
Then, invoke the `print`, `println`, and `printf` methods on the PrintWriter object to write data to a file.

```
public class WriteData {
  public static void main(String[] args) throws IOException {
    java.io.File file = new java.io.File("scores.txt");
    if (file.exists()) {
      System.out.println("File already exists");
      System.exit(1);
    }

    // Create a file
    java.io.PrintWriter output = new java.io.PrintWriter(file);

    // Write formatted output to the file
    output.print("John T Smith ");
    output.println(90);
    output.print("Eric K Jones ");
    output.println(85);

    // Close the file
    output.close();
  }
}
```



Reading Data Using Scanner Class

Scanner Class that we used before to read user inputs from console, it can also be used to read input from File.

To read from a file, create a Scanner for a file, as follows:

```
Scanner input = new Scanner(new File(filename));
```

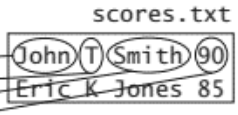
```
import java.util.Scanner;

public class ReadData {
  public static void main(String[] args) throws Exception {
    // Create a File instance
    java.io.File file = new java.io.File("scores.txt");

    // Create a Scanner for the file
    Scanner input = new Scanner(file);

    // Read data from a file
    while (input.hasNext()) {
      String firstName = input.next();
      String mi = input.next();
      String lastName = input.next();
      int score = input.nextInt();
      System.out.println(
        firstName + " " + mi + " " + lastName + " " + score);
    }

    // Close the file
    input.close();
  }
}
```



Lab Task

Create a Quiz class with id and mark. Now write a program that reads a file containing records of Quiz objects and initialize an array. The program should then print all the objects in the Quiz array and print the id of the student who obtained the highest mark in a separate file.

Sample File:
113098 20
115089 15
345678 12
234566 18
Program Output:
ID:113098 mark:20
ID:115089 mark:15
ID:345678 mark:12
ID:234566 mark:18
Highest mark obtained by ID:113098