/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Workshop # 5**

**Course:** *JAC444 - Winter 2021*

**Last Name:** *Trokoz*

**First Name:** *Liubov*

**ID:** *139578199*

**Section:** *NBB*

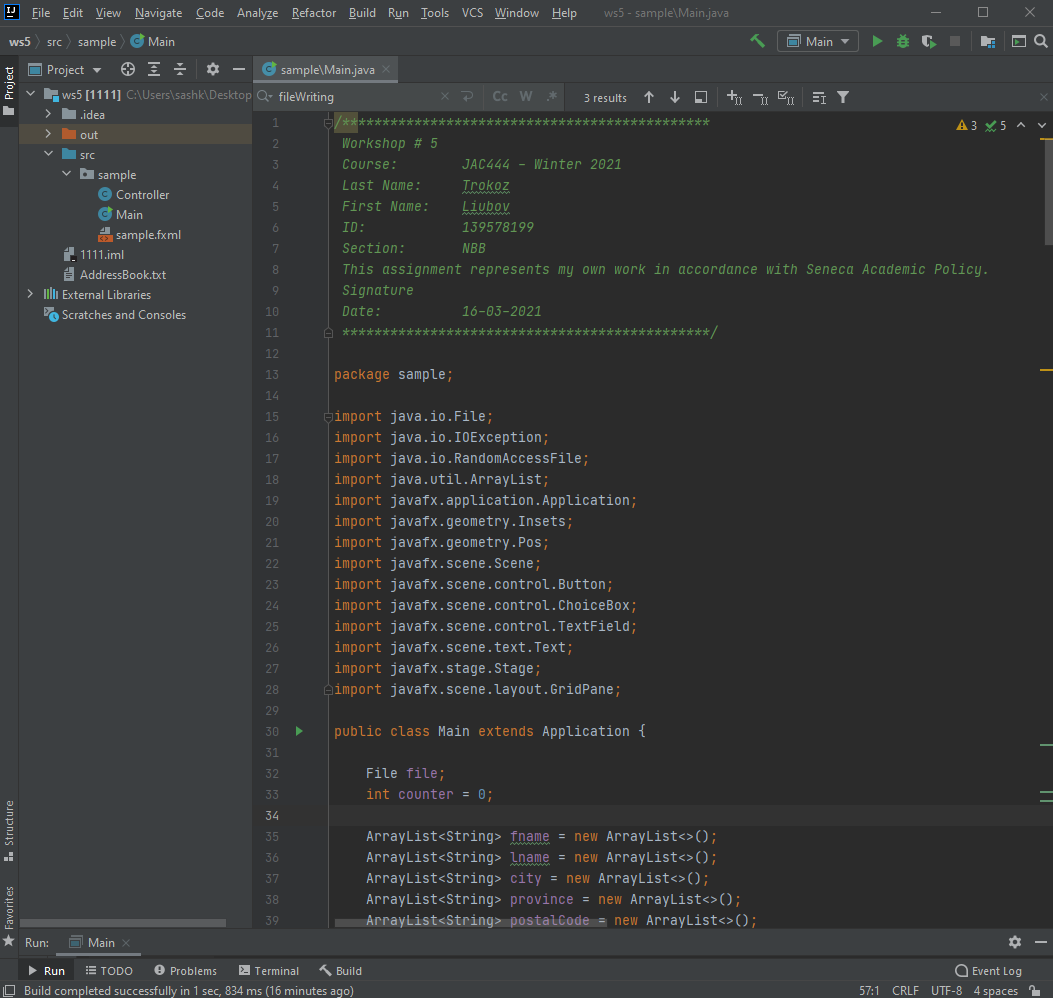
*This assignment represents my own work in accordance with Seneca Academic Policy.*

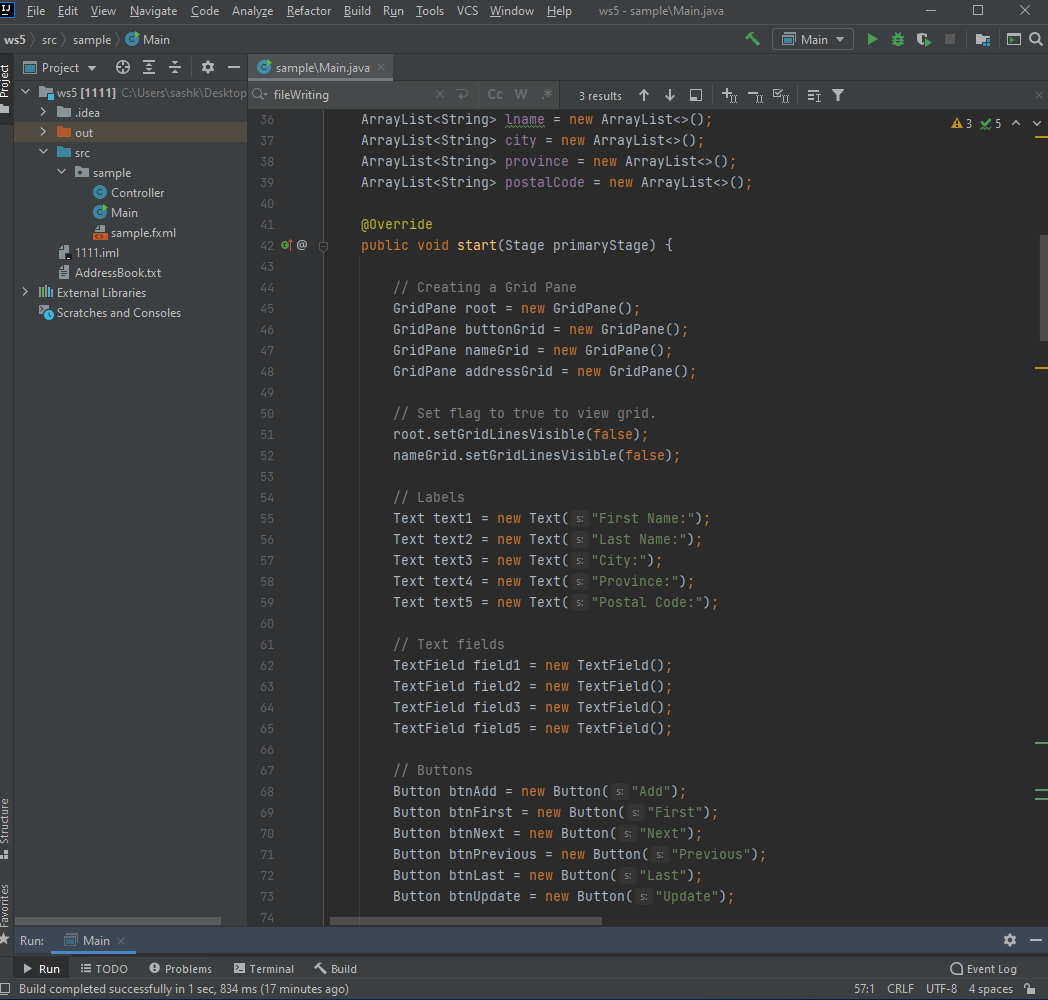
*Signature*

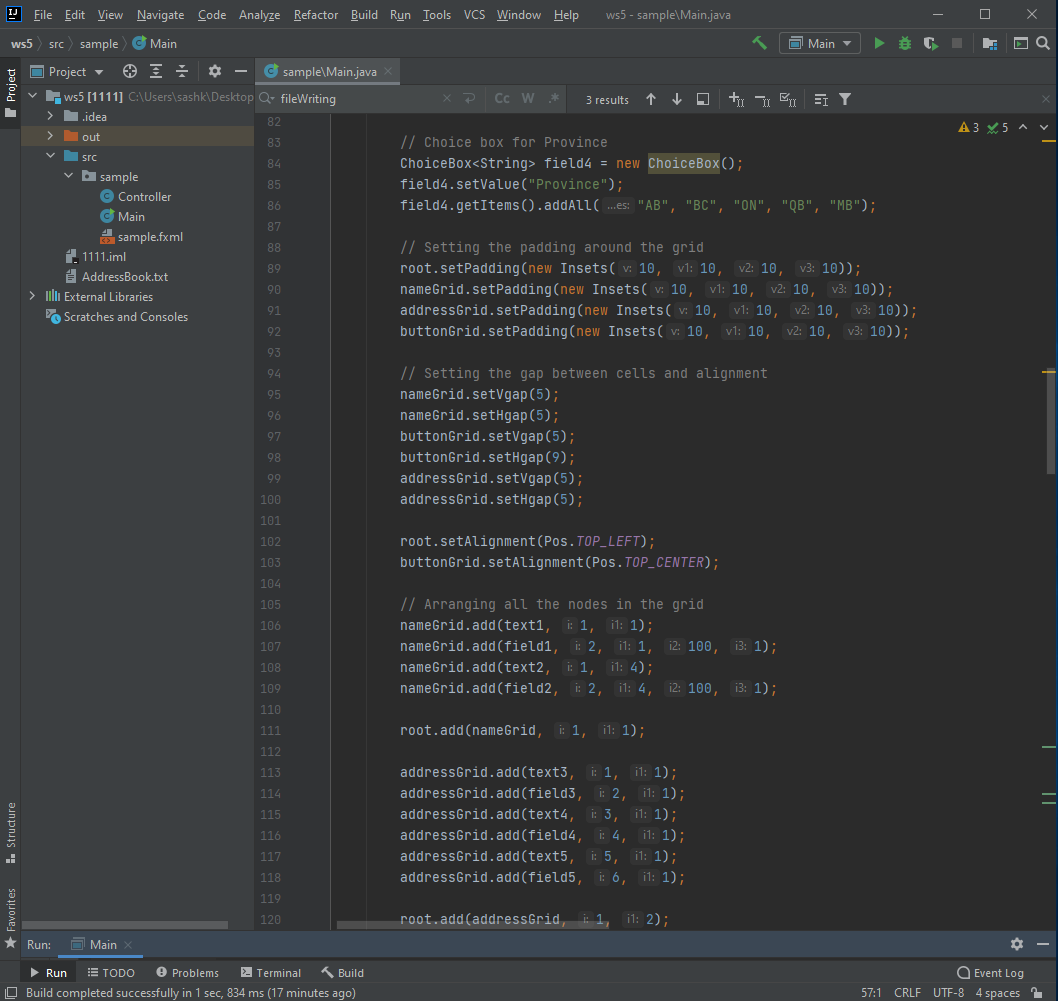
**Date:** *16-03-2021*

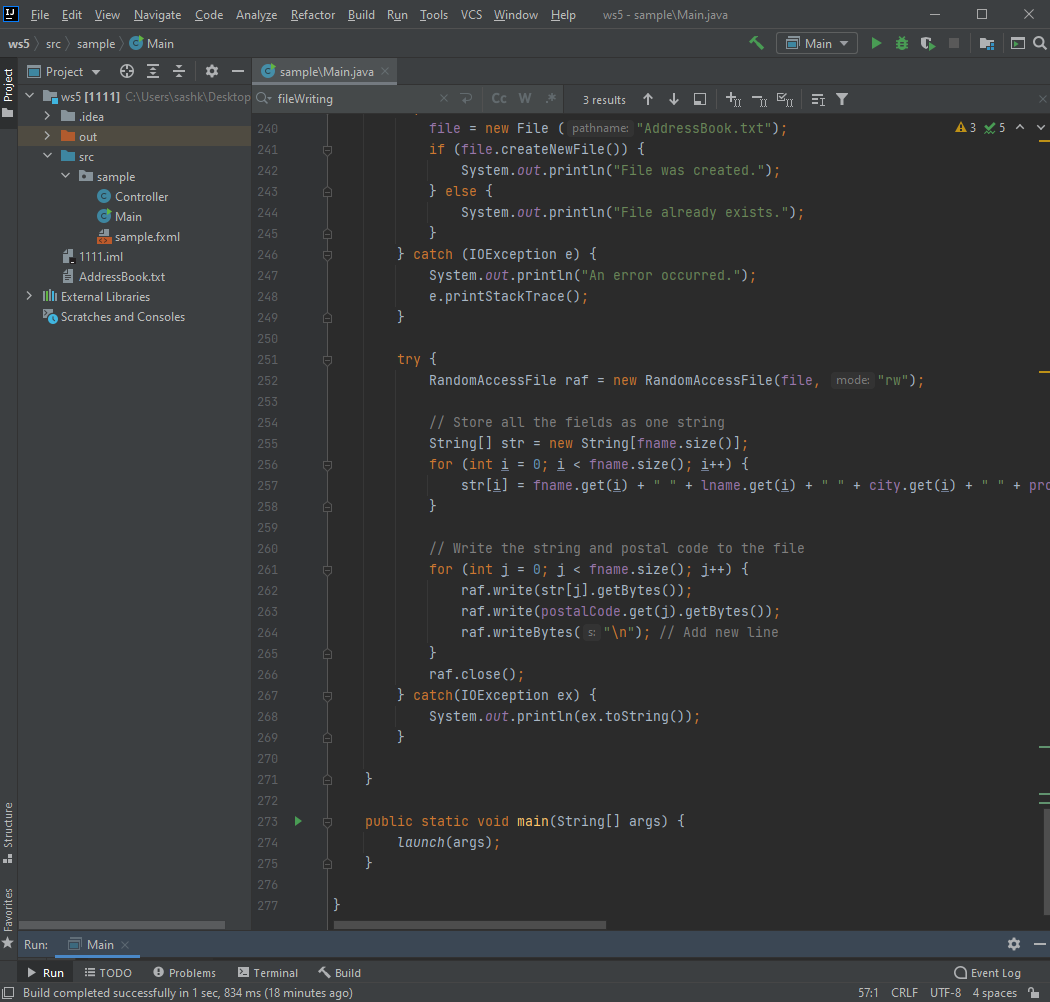
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

* 1. Detailed document includes screen shots of output and answers about **Random Access Files**.
* Main.java

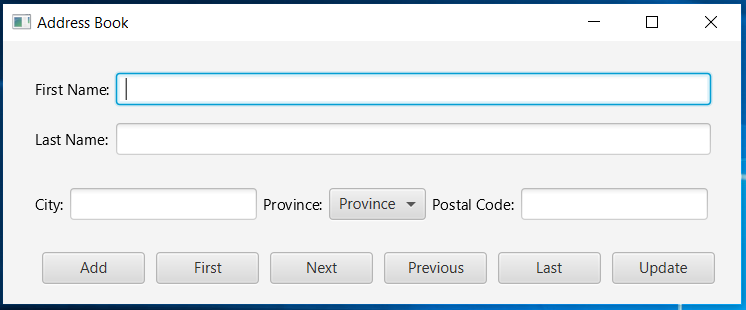


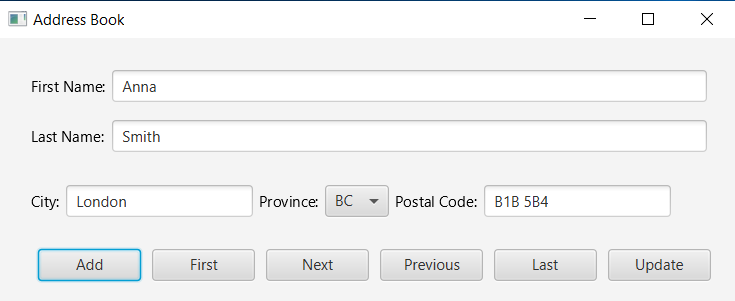


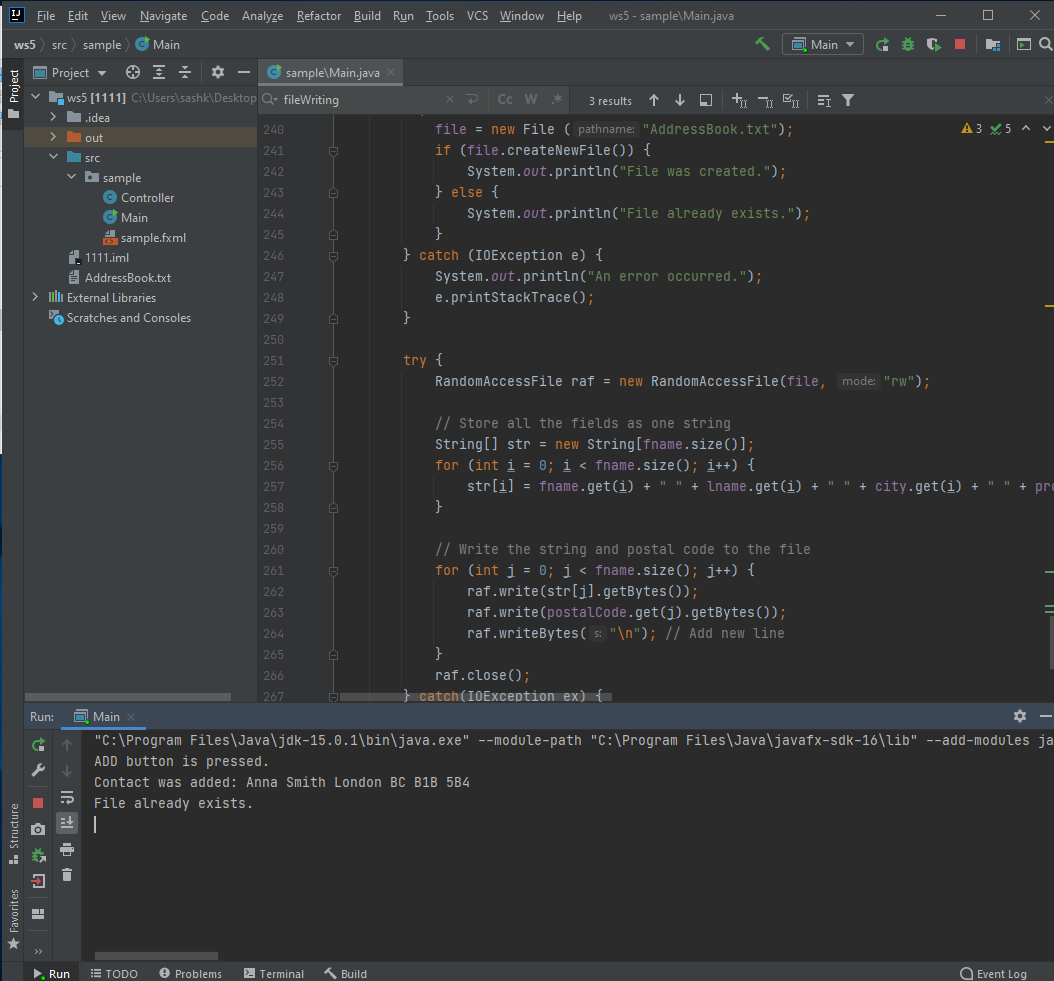


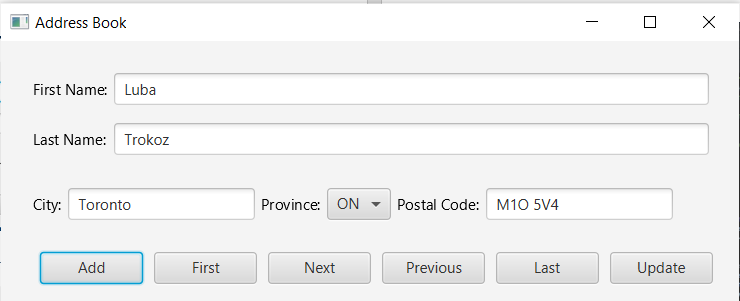


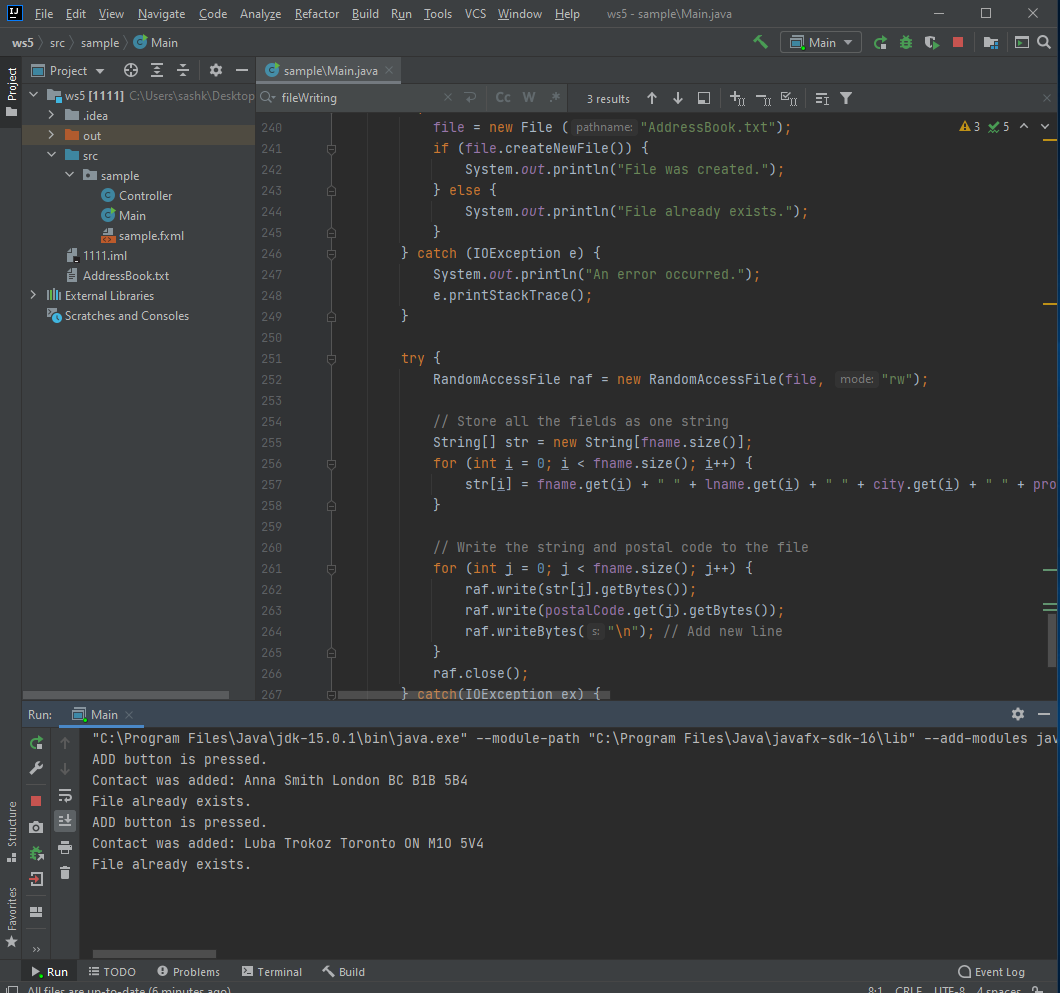
* Output:

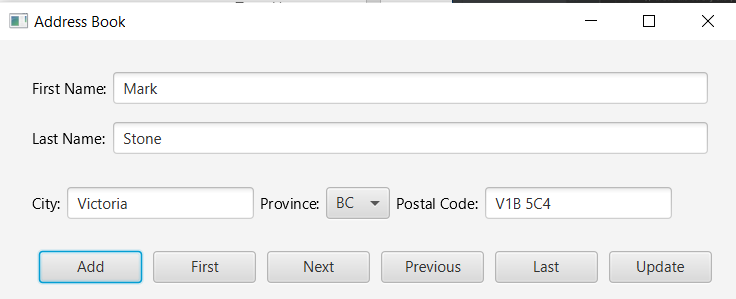


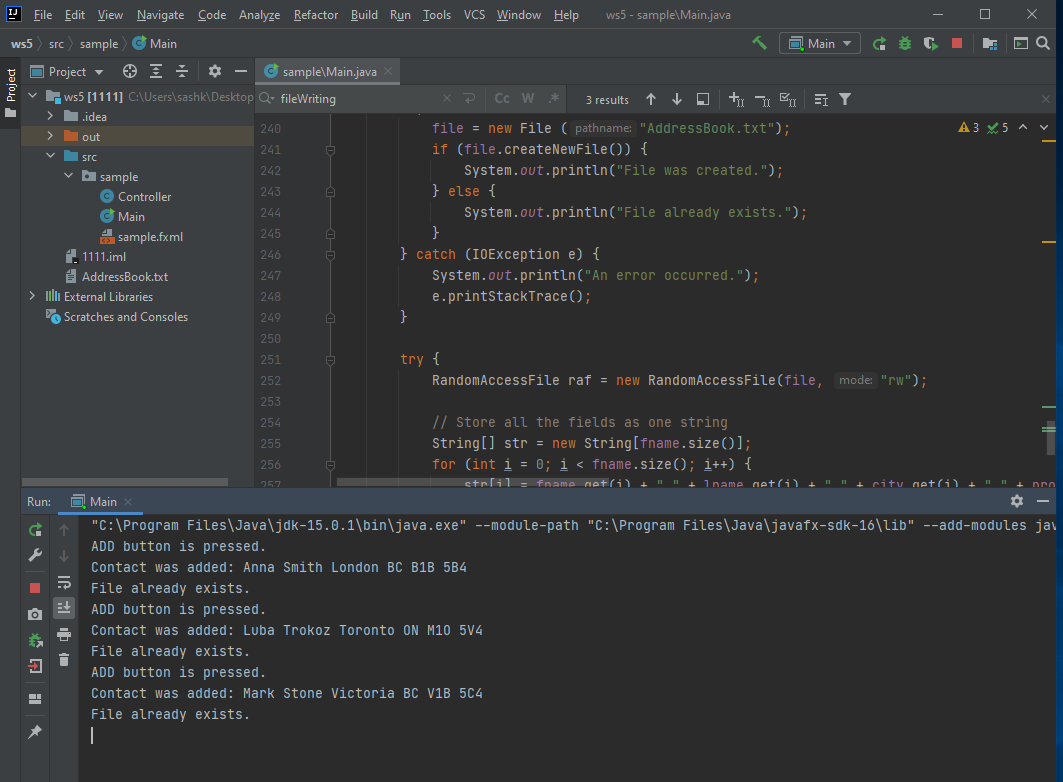


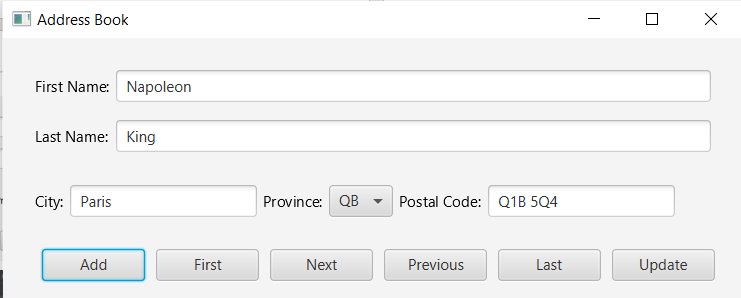


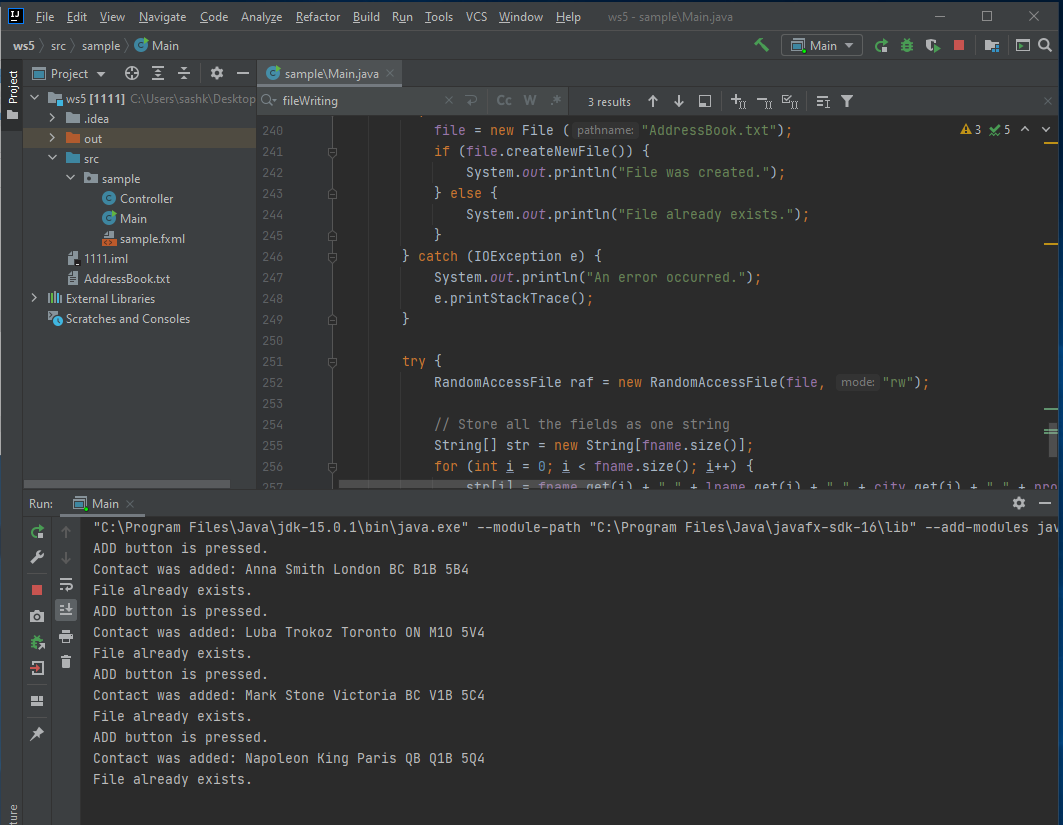


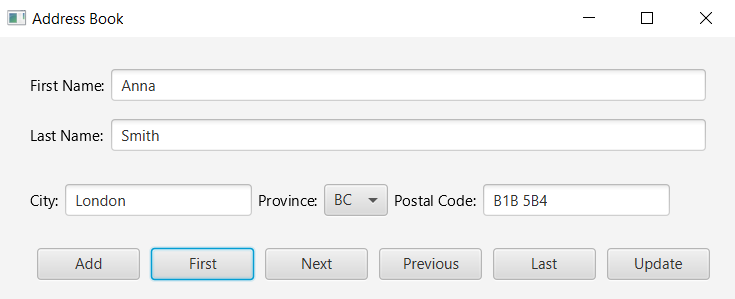


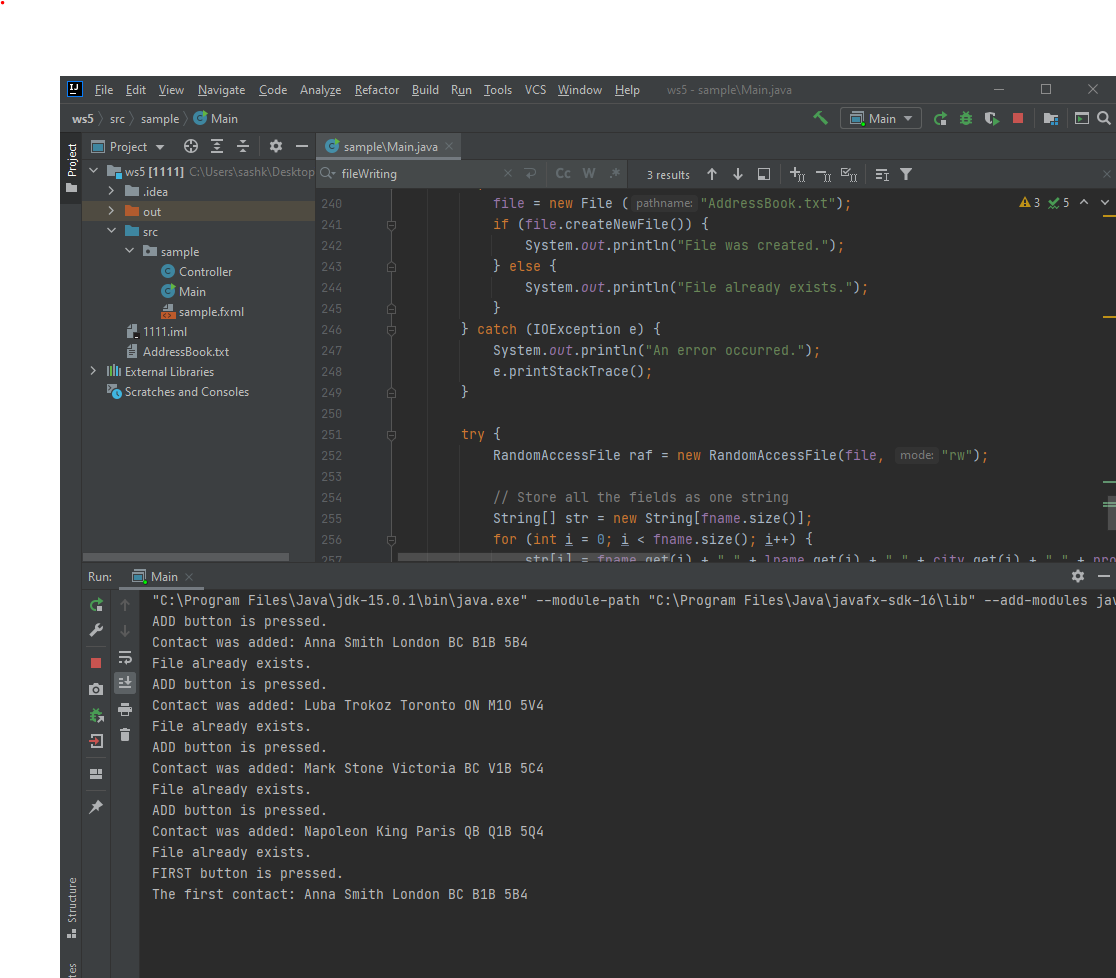


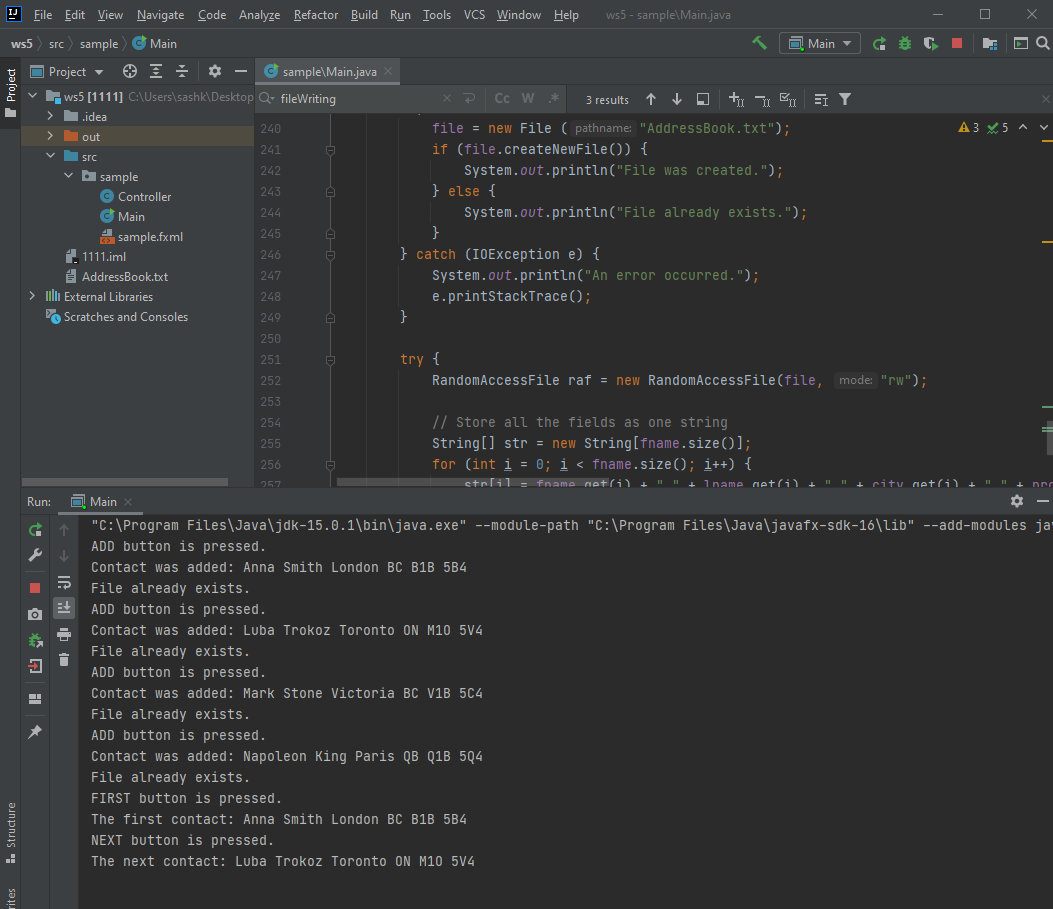
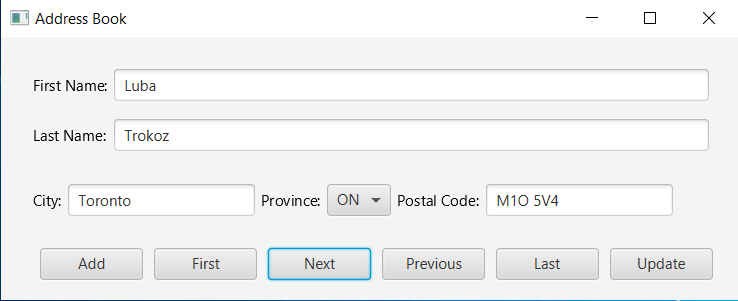


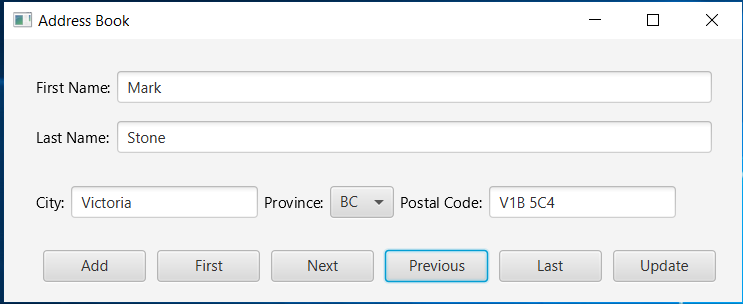
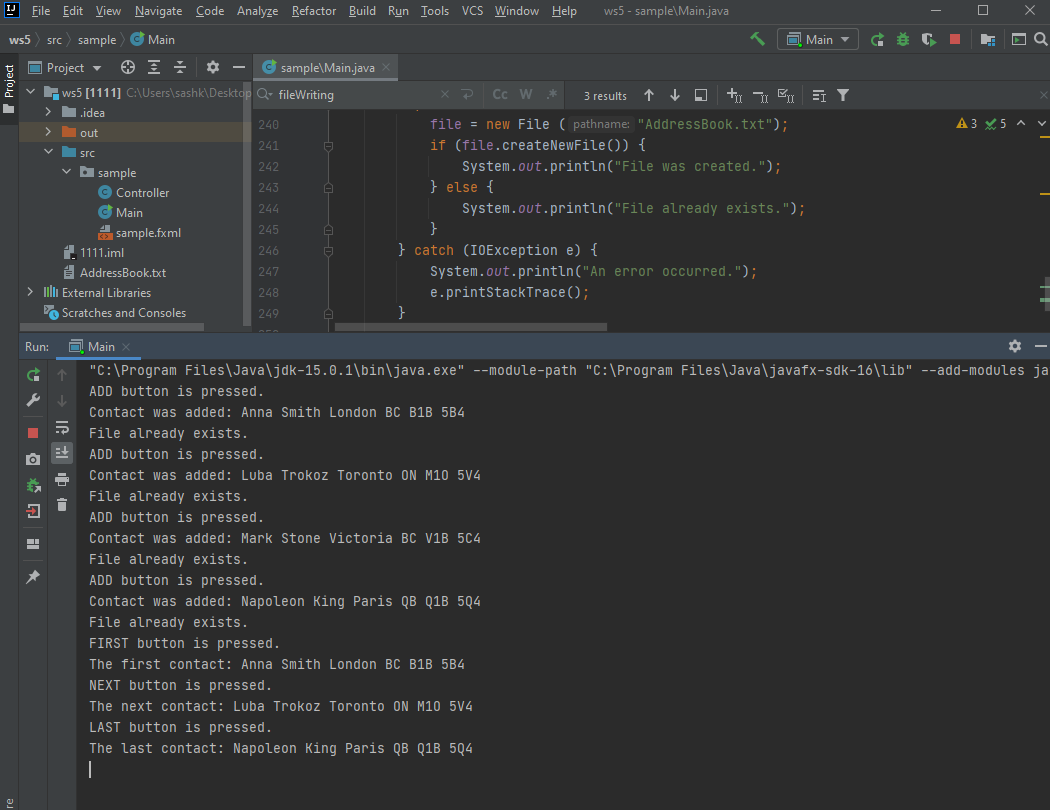
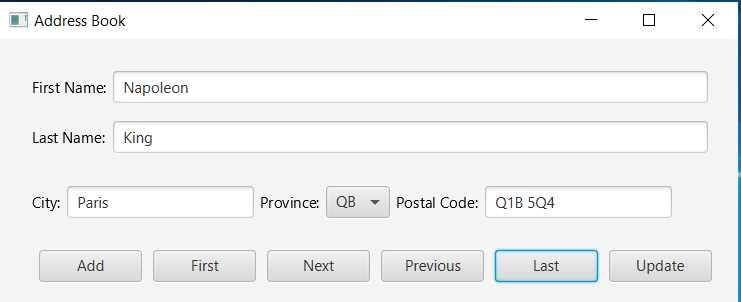


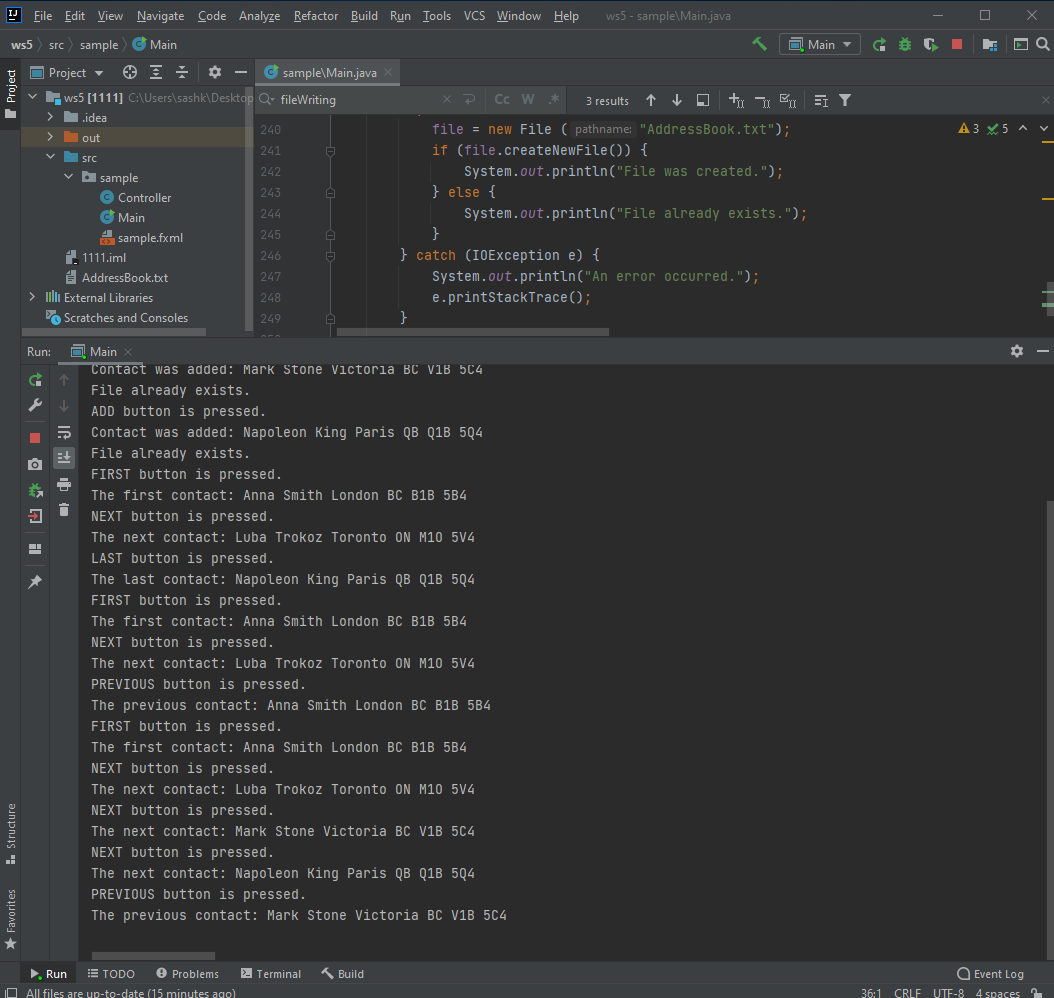




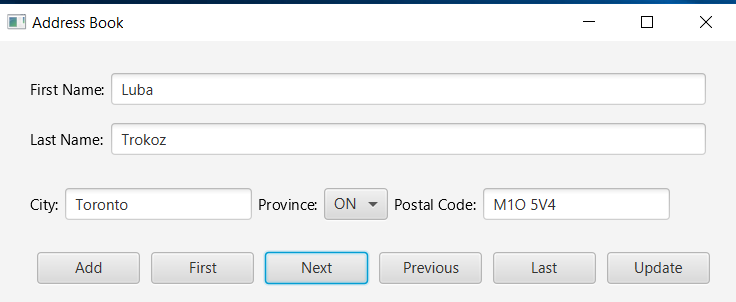


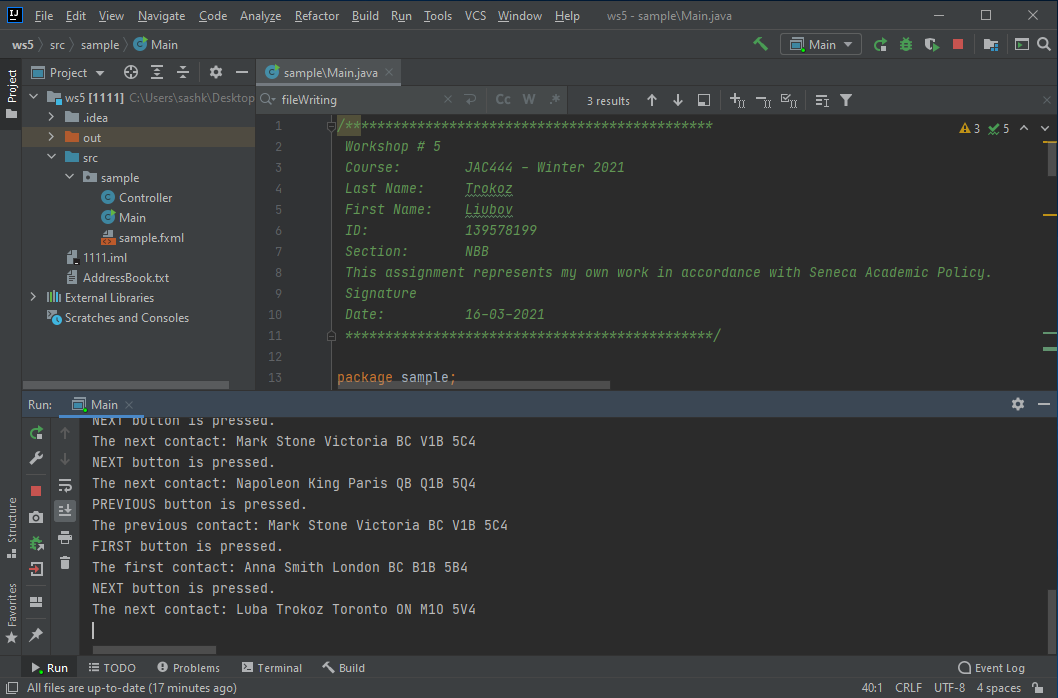
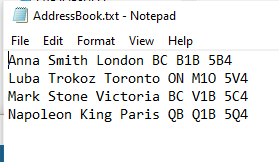


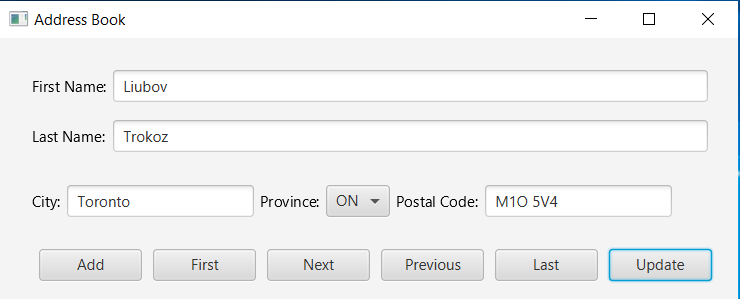


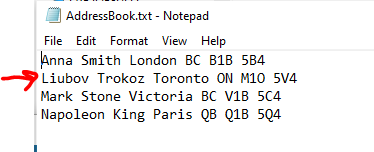


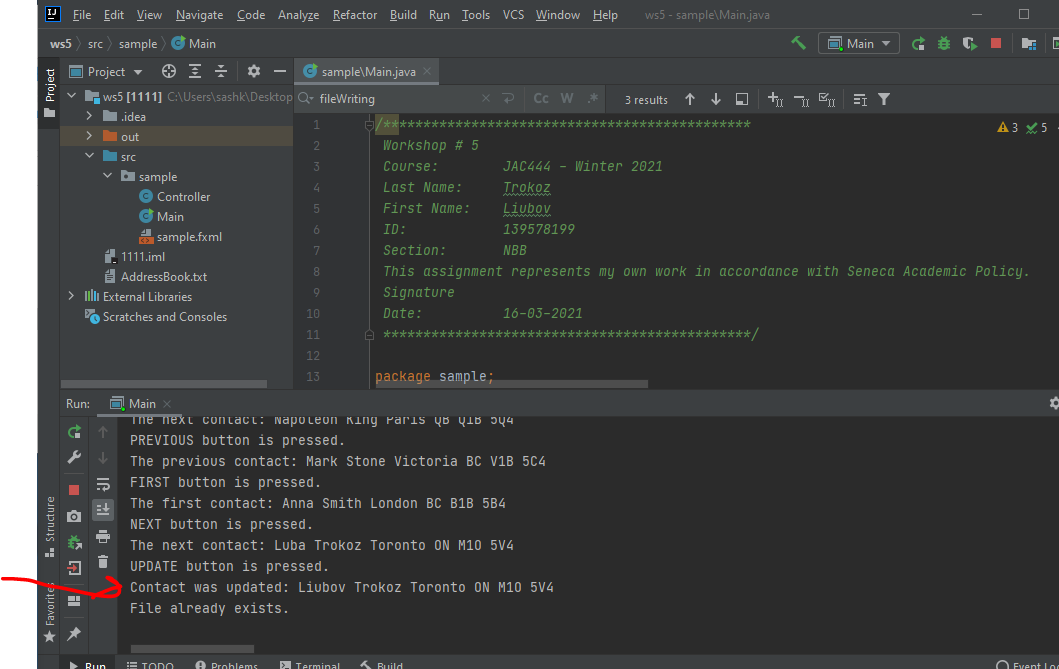
**For updating I found my contact and changed first name from Luba to Liubov:**











# Random Access File

## Advantages

The main advantage of Random Access Files is implied in their name where “every record in the file is available at any time” ([www.manmrk.net](http://www.manmrk.net)).

Also, random-access files allow to get access to any record directly, usually that files are smaller than sequential files, and their “streams can move data in both directions” (http://www-acad.sheridanc.on.ca).

## Disadvantages

The disadvantage of Random Access Files is “can be wasteful of disk space because space is allocated for the longest possible field in every record” ([www.manmrk.net](http://www.manmrk.net)).

## Usage.

Random Access File “provides the facility to read and write data to a file. RandomAccessFile works with file as large array of bytes stored in the file system and a cursor using which we can move the file pointer position.” (www.journaldev.com)

## Syntax.

At the beginning it will be written “RandomAccessFile(File file, String mode)”.

Where mode can be next: "r" - open for reading only. Invoking any of the write methods of the resulting object will cause an IOException to be thrown."rw" - open for reading and writing. If the file does not already exist then an attempt will be made to create it."rws" - open for reading and writing, as with "rw", and also require that every update to the file's content or metadata be written synchronously to the underlying storage device. "rwd" - open for reading and writing, as with "rw", and also require that every update to the file's content be written synchronously to the underlying storage device.” (<https://docs.oracle.com/javase/7/docs/api/java/io/RandomAccessFile.html>)

Random Access File must be close() in the end.

## Constructors

RandomAccessFile(File file, String mode) – for reading and optionally writing to the file with the specified name.

RandomAccessFile(String name, String mode) – for reading and optionally writing to the file with the specified in the file argument.

## Methods

There are a lot of RandomAccessFile methods. Here is a part of it (https://docs.oracle.com/javase/7/docs/api/java/io/RandomAccessFile.html):

getFD() - Returns the opaque file descriptor object associated with this stream.

getChannel() - Returns the unique FileChannel object associated with this file.

read() - Reads a byte of data from this file. The byte is returned as an integer in the range 0 to 255 (0x00-0x0ff). This method blocks if no input is yet available.

write() - Writes the specified byte to this file. The write starts at the current file pointer.

seek() - Sets the file-pointer offset, measured from the beginning of this file, at which the next read or write occurs. The offset may be set beyond the end of the file. Setting the offset beyond the end of the file does not change the file length. The file length will change only by writing after the offset has been set beyond the end of the file.

length() - Returns the length of this file.

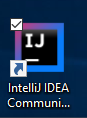
close() - Closes this random access file stream and releases any system resources associated with the stream. A closed random access file cannot perform input or output operations and cannot be reopened.

# References

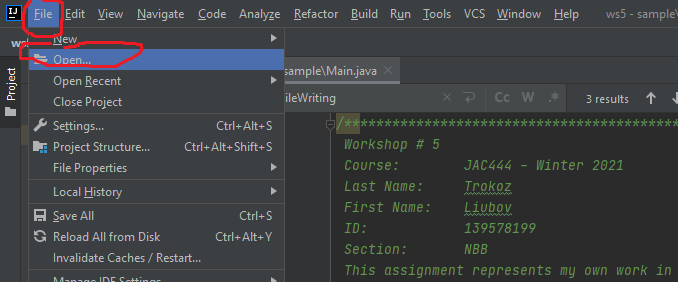
1. <http://www.manmrk.net/tutorials/basic/PowerBASIC/PBW10/html/Random_Access_Files.html#:~:text=However%2C%20random%20access%20files%20can,a%20thousand%20actually%20uses%20it>.
2. <http://www-acad.sheridanc.on.ca/~jollymor/prog24178/files5.html>
3. <https://www.journaldev.com/921/java-randomaccessfile-example>
4. https://docs.oracle.com/javase/7/docs/api/java/io/RandomAccessFile.html

**The instructions on how to run code**

1. Open IntelliJ and create project



1. Open project



1. Run code – Shift + F10 (Microsoft)

