CS 111, Programming Fundamentals II
Lab 8: Random Access Files

Computer Science

This lab is meant to further your understanding of binary files, and give you hands-on experience random access files.

Introduction

We've discussed in lecture, what binary files are and how to work with them in Java. For this lab you will be provided a binary file which contains some information about a few house listed for sale. For each house there will be an address, number of bedrooms, square footage, price, and a short description about the house. To get started download the following file from Canvas:

- ***HouseData.bin***

HouseData.bin is a binary file. You can try opening it in a Notepad but you will only see some of the information stored in the file. It is not a text file, but rather a binary file. Some of the information stored in the binary file is text, even though it is stored in binary format, you will be able to see the characters. This is because when a character is being stored it is converted to a decimal equivalent Unicode number, which is then encoded in binary, Text editors to the reverse of it to show us characters.

The file has the following structure for each entry (house record):

- Address - **44 bytes** total
- Num bedrooms - **1 byte**
- Square footage - **2 bytes**
- Price - **4 bytes**
- Description – **112 bytes**

Address and description were written with method ***writeUTF()***. Address is originally 42 bytes long, but the method added 2 additionally bytes in the begging to specify how many bytes belong to this string. If you use the method ***readUTF()*** to read the address information, you don't have to worry about those two bytes.

In this lab you will read all bytes from the file and print out the information. There are several distinct house "entries" in the file. You will also write a method, which will update the price for each house by a certain percentage.

Write a java class named **ReadHouseData** This class should have the main method and two additional methods. A method named ***printOutHouseData()***, which takes a single parameter of type String representing the file name. This method will read the bytes from the file and print out house records. The second method should be named ***increasePrice()***, which also takes in a file name and an additional int value. The class **RandomAccessFile** allows us to move the file pointer to any location in the file. Using the class **RandomAccessFile** in the method ***increasePrice(),*** you will update the price for each house record. More information about each method is available below.

I. printOutHouseData()

Using **DataInputStream** with **FileInputStream** read the binary file. Have a look at the lecture slides on how to set this up, if you don’t remember. There are several methods in the **DataInputStream** which can be used to read the file. Have a look at the API.

To read the address, you can use the method ***readUTF()***. To read the square footage, you can use the method ***readShort().*** The numbers of bedrooms is stored as a single byte. Check which method can be used to read a single byte and represent a number. Also check which method can read 4 bytes and represent a number for the price of the house.

It is crucial to read the information from the file in the correct order. Otherwise you will not get the desired output. Have a look on the previous page at the file structure. As you are reading the information and printing it out, add brief text to each print out. For example when you print out the number of bedrooms, you can add text like “Number of bedrooms : ”. Have a look at the sample output in **Figure 1**.

II. increasePrice()

The method ***increasePrice()*** should take in two parameters, a file name and an integer, to specify a percent value by which the price should be increased. In this method open the binary file using the **RandomAccessFile** class, in “**rw**” mode. Refer to the lecture slides for examples if needed.

RandomAccessFile class has the same methods as **DataInputStream** and **DataOutputStream**. Additionally it also has a few other methods, one of them being ***seek().*** The **seek()** method takes in an integer value, and places the file pointer at the location of the integer value. When reading/writing is being performed, it is always done at the location of the file pointer.

By looking at the file structure (number of bytes for each piece of information) you can calculate at which byte location the price for the first house is stored. Do the following inside a loop:

1. Using the method **seek()**, you can move the file pointer to the location of the first byte for the price of the first house.

2. Read the price, and increase it by the percent value getting passed into the method.

3. Move the file pointer back to the beginning of the price you just read.

4. Write to the file the updated price.

5. Move the pointer to the next house entry in the file. (You will need to calculate the number of bytes to move the file pointer.)

If the steps mentioned above are done properly, the prices should be updated without corrupting the file.

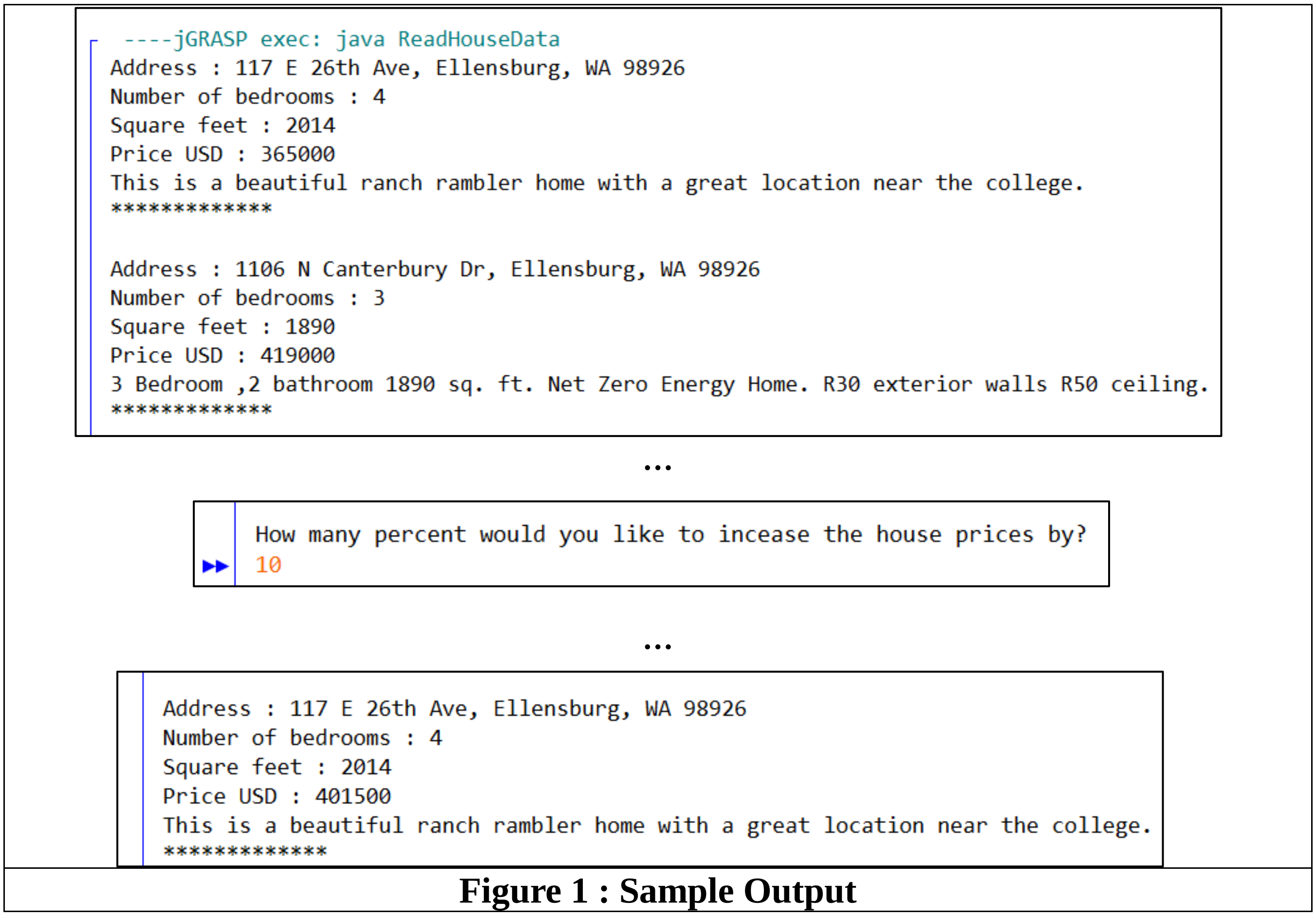
III. main()

This method is very short and can be written in 7 lines of code, not including comments or line breaks. In the main method do the following:

1. Call the method ***printOutHouseData()*** passing to it the name of the file.
2. Prompt the user and retrieve a value for percent by which the house prices should be increased.
3. Call the method ***increasePrice()*** passing to it the file name and the percent value.
4. Print out some text to separate the next set of print statements, could be something like "updating prices" or just a few line breaks.
5. Call the method ***printOutHouseData()*** once more.

IV. Sample Output

Sample output of the program can be seen below on **Figure 1** and **Figure 2**.



```
 ----jGRASP exec: java ReadHouseData
Address : 117 E 26th Ave, Ellensburg, WA 98926
Number of bedrooms : 4
Square feet : 2014
Price USD : 365000
This is a beautiful ranch rambler home with a great location near the college.
*************

Address : 1106 N Canterbury Dr, Ellensburg, WA 98926
Number of bedrooms : 3
Square feet : 1890
Price USD : 419000
3 Bedroom ,2 bathroom 1890 sq. ft. Net Zero Energy Home. R30 exterior walls R50 ceiling.
*************
```

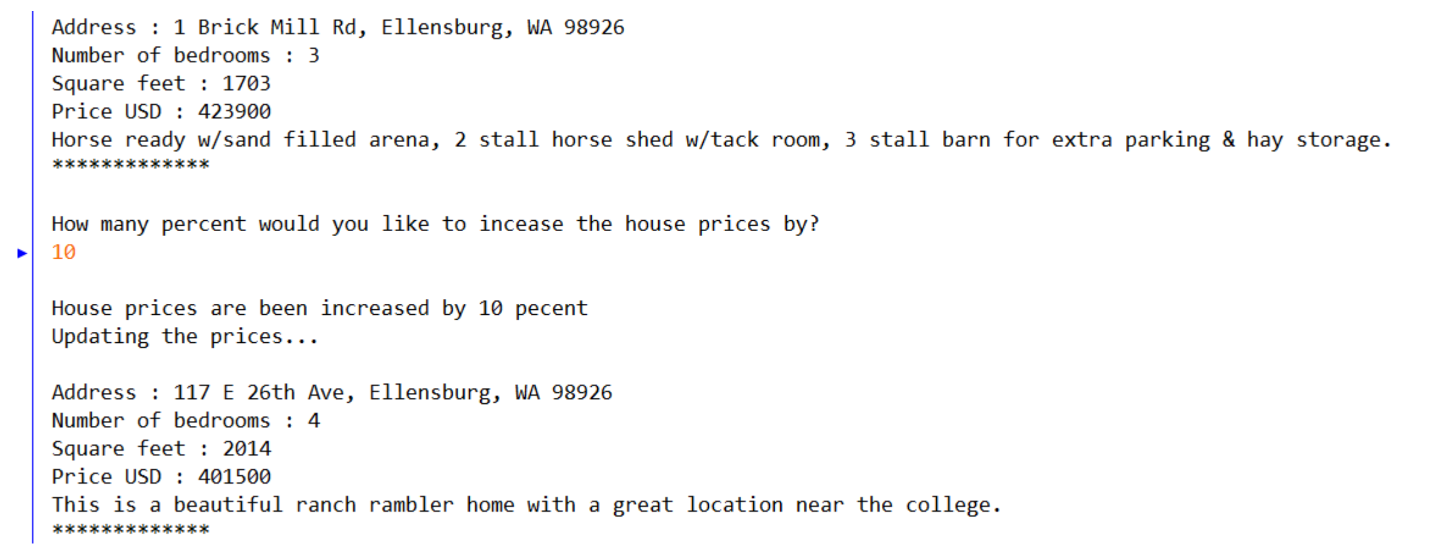
…

```
How many percent would you like to incease the house prices by?
10
```

…

```
Address : 117 E 26th Ave, Ellensburg, WA 98926
Number of bedrooms : 4
Square feet : 2014
Price USD : 401500
This is a beautiful ranch rambler home with a great location near the college.
*************
```

Figure 1 : Sample Output



```
Address : 1 Brick Mill Rd, Ellensburg, WA 98926
Number of bedrooms : 3
Square feet : 1703
Price USD : 423900
Horse ready w/sand filled arena, 2 stall horse shed w/tack room, 3 stall barn for extra parking & hay storage.
*************

How many percent would you like to incease the house prices by?
10

House prices are been increased by 10 pecent
Updating the prices...

Address : 117 E 26th Ave, Ellensburg, WA 98926
Number of bedrooms : 4
Square feet : 2014
Price USD : 401500
This is a beautiful ranch rambler home with a great location near the college.
*************
```

Figure 2 : Another Sample Output

V. Upload your work to Canvas

For this lab, make sure that you upload the following file to the Lab 8 assignment in your Canvas account:

ReadHouseData.java

Rubric

File / Lab	Points
Method *ReadHouseData* reads and prints out the house records	30
Method *increasePrice* correctly increases the price and overwrites it in the file	40
The *main* method calls the methods correctly and prompts the user	10
Code compiles and runs as expected	15
Comments, format and a screenshot of the output	5
Total	100