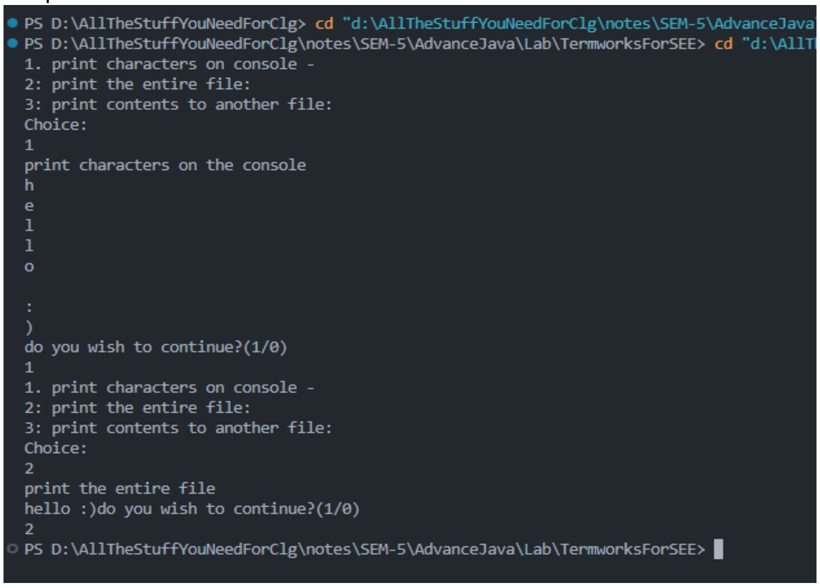
[1]Write a menu driven Java program to read contents of a file and: a) print characters on the console – one character at a time b) print the entire file c) print contents to another file. Read both source & target file names & check for their existence/ non – existence to take appropriate actions.

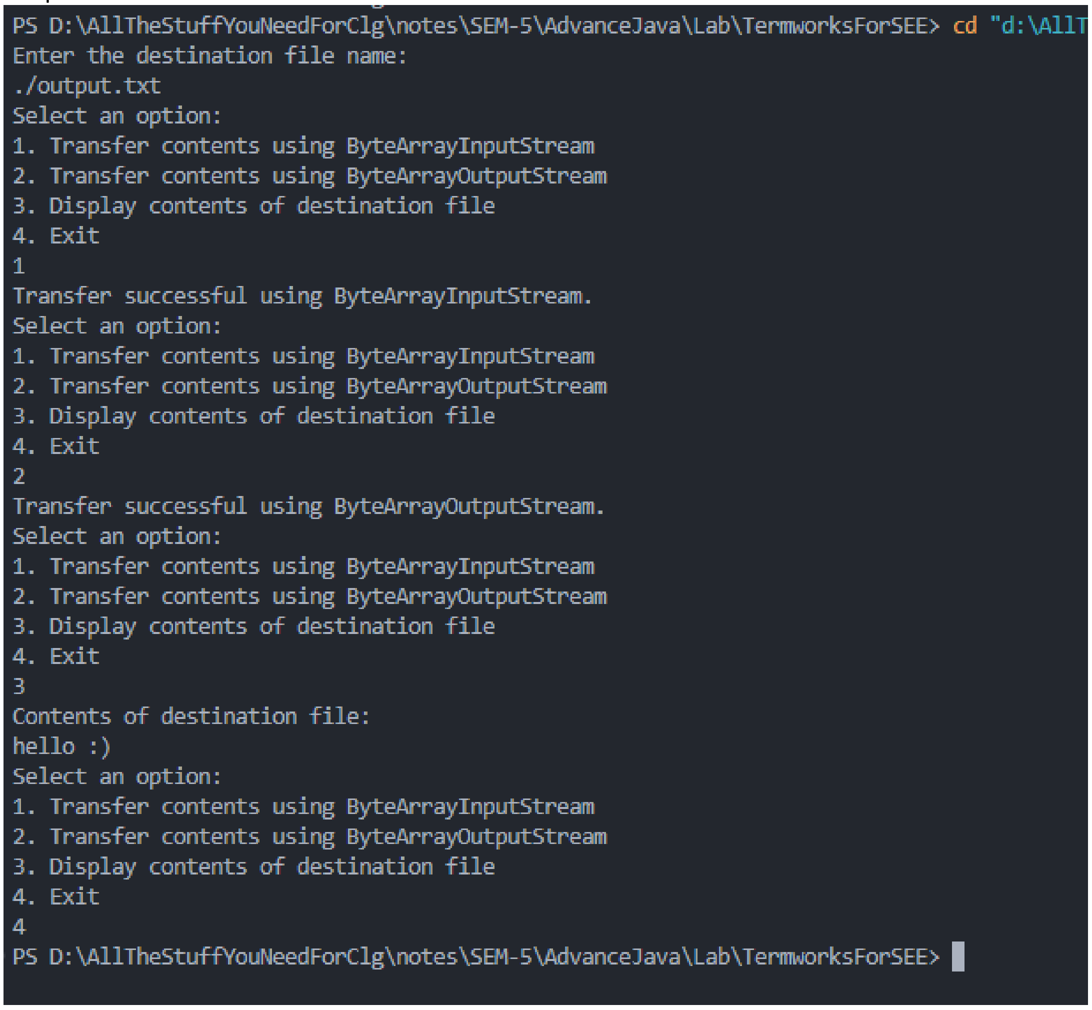
```java
//please make some legit changes and dont get caught if two or more programs are same
import java.io.*;
import java.util.*;
public class First{
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int flag = 1;
        while(flag == 1){
            System.out.println("1. print characters on console - ");
            System.out.println("2: print the entire file: ");
            System.out.println("3: print contents to another file: ");
            System.out.println("Choice: ");
            int c = s.nextInt();
            switch(c){
                case 1: System.out.println("print characters on the console");
                        try{
                            FileInputStream fip = new FileInputStream("input.txt");
                            int ch = fip.read();
                            //System.out.println(ch);
                            while(ch != -1){
                                System.out.println((char)ch);
                                ch = fip.read();
                            }
                            fip.close();
                        }
                        catch(Exception e){
                            System.out.println(e);
                        }
                        break;
                case 2: System.out.println("print the entire file");
                        try{
                            FileInputStream fip = new FileInputStream("input.txt");
                            int ch = fip.read();
                            //System.out.println(ch);
                            while(ch != -1){
                                System.out.print((char)ch);
                                ch = fip.read();
                            }
                            fip.close();
                        }
                        catch(Exception e){
                            System.out.println(e);
                        }
                        break;
                case 3: System.out.println("print contents onto other file");
                        try{
                            FileInputStream fip = new FileInputStream("input.txt");
                            FileOutputStream fop = new FileOutputStream("output.txt");
                            byte[] buf = new byte[100];
                            fip.read(buf);
                            fop.write(buf);
                            System.out.println("copied to another file: output.txt");
                            fip.close();
                            fop.close();
                        }
                        catch(Exception e){
                            System.out.println(e);
                        }
                        break;
            }
            System.out.println("do you wish to continue?(1/0)");
            flag=s.nextInt();
        }
        s.close();
    }
}
```
Output:


```
PS D:\AllTheStuffYouNeedForClg> cd "d:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE> cd "d:\AllT
1. print characters on console -
2: print the entire file:
3: print contents to another file:
Choice:
1
print characters on the console
h
e
l
l
o

:
)
do you wish to continue?(1/0)
1
1. print characters on console -
2: print the entire file:
3: print contents to another file:
Choice:
2
print the entire file
hello :)do you wish to continue?(1/0)
2
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE>
```

[2] Write a Java Program to demonstrate the implementation of stream classes in Java. Assume that an input file named "input.txt" already exists with few lines of random text. Accept a filename from the user, this will be the destination file. Write a menu driven program to do the following: 1) Transfer the contents of the input file to the destination file using the ByteArrayInputStream /ByteArrayInputStream class 3) Display the contents of the destination file

```
//please cancel out the comments, edit the input.txt file and delete contents from
output.txt file if you get this program
// and also change the menu options
import java.io.*;

public class Second {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter the destination file name:");
            String destFile = reader.readLine(); //specify output file as ./output.txt
creating a folder containing all the files in one place
            int choice;
            do {
                System.out.println("Select an option:");
                System.out.println("1. Transfer contents using ByteArrayInputStream");
                System.out.println("2. Transfer contents using ByteArrayOutputStream");
                System.out.println("3. Display contents of destination file");
                System.out.println("4. Exit");
                choice = Integer.parseInt(reader.readLine());
                switch (choice) {
                    case 1:
                        transferWithByteArrayInputStream(destFile);
                        break;
                    case 2:
                        transferWithByteArrayOutputStream(destFile);
                        break;
                    case 3:
                        displayDestinationFile(destFile);
                        break;
                    case 4:
                        break;
                    default:
                        System.out.println("Invalid choice, please try again.");
                }
            } while (choice != 4);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

```java
private static void transferWithByteArrayInputStream(String destFile) {
    try {
        FileInputStream inputStream = new FileInputStream("input.txt");
        ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = inputStream.read(buffer)) != -1) {
            outputStream.write(buffer, 0, bytesRead);
        }
        byte[] data = outputStream.toByteArray();
        ByteArrayInputStream byteArrayInputStream = new ByteArrayInputStream(data);
        FileOutputStream fileOutputStream = new FileOutputStream(destFile);
        while ((bytesRead = byteArrayInputStream.read(buffer)) != -1) {
            fileOutputStream.write(buffer, 0, bytesRead);
        }
        System.out.println("Transfer successful using ByteArrayInputStream.");
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private static void transferWithByteArrayOutputStream(String destFile) {
    try {
        FileInputStream inputStream = new FileInputStream("input.txt");
        ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = inputStream.read(buffer)) != -1) {
            outputStream.write(buffer, 0, bytesRead);
        }
        byte[] data = outputStream.toByteArray();
        FileOutputStream fileOutputStream = new FileOutputStream(destFile);
        fileOutputStream.write(data);
        System.out.println("Transfer successful using ByteArrayOutputStream.");
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private static void displayDestinationFile(String destFile) {
    try {
        FileInputStream inputStream = new FileInputStream(destFile);
        BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));
        String line;
        System.out.println("Contents of destination file:");
        while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```
Output:



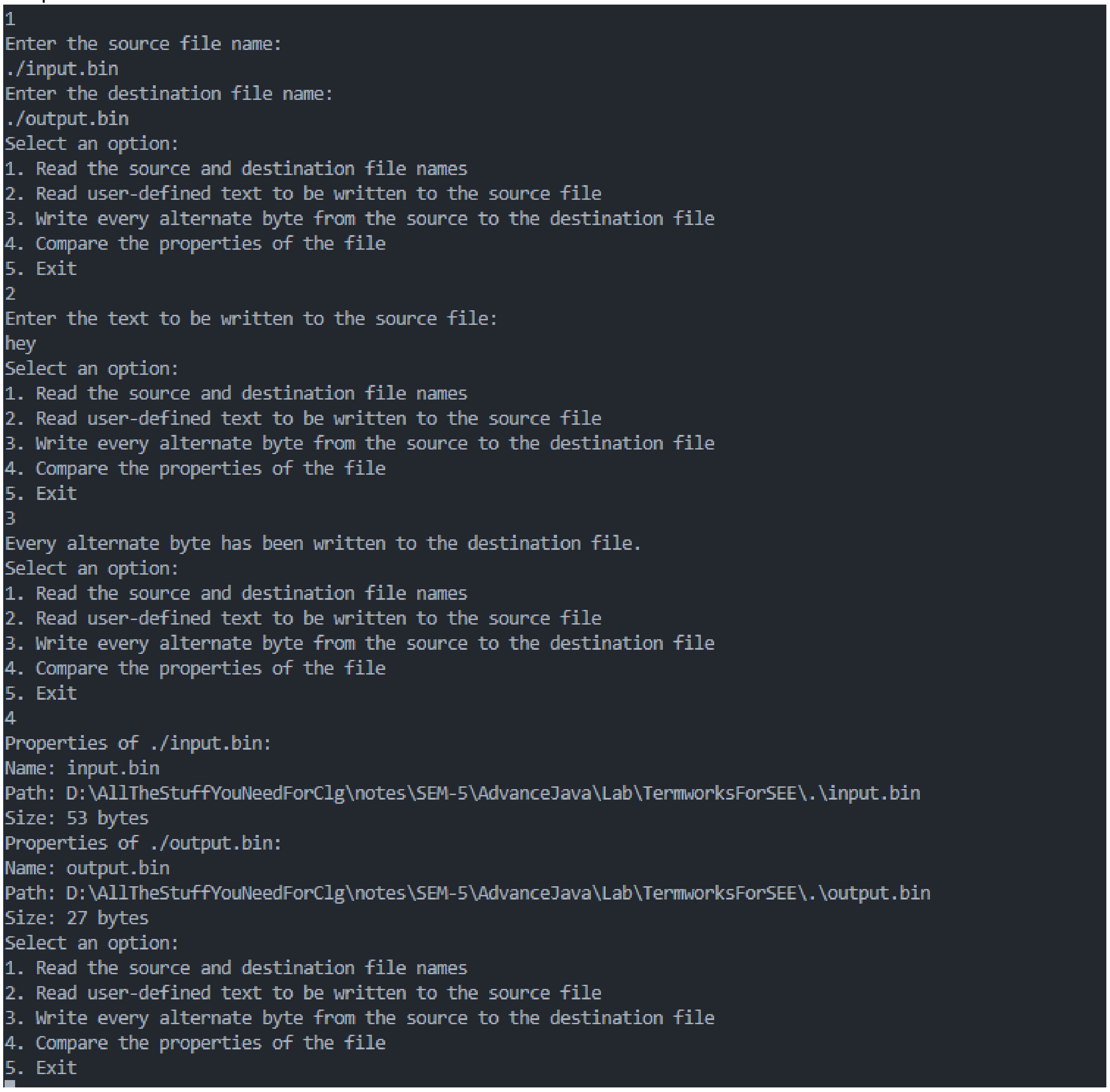
```
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE> cd "d:\AllT
Enter the destination file name:
./output.txt
Select an option:
1. Transfer contents using ByteArrayInputStream
2. Transfer contents using ByteArrayOutputStream
3. Display contents of destination file
4. Exit
1
Transfer successful using ByteArrayInputStream.
Select an option:
1. Transfer contents using ByteArrayInputStream
2. Transfer contents using ByteArrayOutputStream
3. Display contents of destination file
4. Exit
2
Transfer successful using ByteArrayOutputStream.
Select an option:
1. Transfer contents using ByteArrayInputStream
2. Transfer contents using ByteArrayOutputStream
3. Display contents of destination file
4. Exit
3
Contents of destination file:
hello :)
Select an option:
1. Transfer contents using ByteArrayInputStream
2. Transfer contents using ByteArrayOutputStream
3. Display contents of destination file
4. Exit
4
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE>
```

[3] Write a Java Program to demonstrate the implementation of reading and writing binary data in Java. 1) Read the source and destination file names. 2) Read user defined text to be written to the source file. 3) Write every alternate byte from the source to the destination file. 4) Compare the properties of the file.

```
//please cancel out the comments, edit the input.txt file and delete contents from
output.txt file if you get this program
// and also change the menu options
import java.io.*;
public class Third {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
            String sourceFile = null;
            String destFile = null;
            String text = null;
            int choice;
            do {
                System.out.println("Select an option:");
                System.out.println("1. Read the source and destination file names");
                System.out.println("2. Read user-defined text to be written to the source
file");
                System.out.println("3. Write every alternate byte from the source to the
destination file");
                System.out.println("4. Compare the properties of the file");
                System.out.println("5. Exit");
                choice = Integer.parseInt(reader.readLine());
                switch (choice) {
                    case 1:
                        System.out.println("Enter the source file name:");
                        sourceFile = reader.readLine(); //specify input file as ./input.bin
creating a folder containing all the files in one place
                        System.out.println("Enter the destination file name:");
                        destFile = reader.readLine(); //specify output file as ./output.bin
creating a folder containing all the files in one place
                        break;
                    case 2:
                        System.out.println("Enter the text written to the source file:");
                        text = reader.readLine();
                        break;
                    case 3:
                        writeAlternateBytes(sourceFile, destFile);
                        break;
                    case 4:
                        compareProperties(sourceFile, destFile);
                        break;
                    case 5:
```

```java
                        break;
                    default:
                        System.out.println("Invalid choice, please try again.");
                }
            } while (choice != 5);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private static void writeAlternateBytes(String sourceFile, String destFile) {
        try {
            FileInputStream inputStream = new FileInputStream(sourceFile);
            FileOutputStream outputStream = new FileOutputStream(destFile);
            int b;
            int count = 0;
            while ((b = inputStream.read()) != -1) {
                if (count % 2 == 0) {
                    outputStream.write(b);
                }
                count++;
            }
            System.out.println("Every alternate byte written to the destination file.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private static void compareProperties(String sourceFile, String destFile) {
        try {
            File file1 = new File(sourceFile);
            File file2 = new File(destFile);
            System.out.println("Properties of " + sourceFile + ":");
            System.out.println("Name: " + file1.getName());
            System.out.println("Path: " + file1.getAbsolutePath());
            System.out.println("Size: " + file1.length() + " bytes");
            System.out.println("Properties of " + destFile + ":");
            System.out.println("Name: " + file2.getName());
            System.out.println("Path: " + file2.getAbsolutePath());
            System.out.println("Size: " + file2.length() + " bytes");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Output:



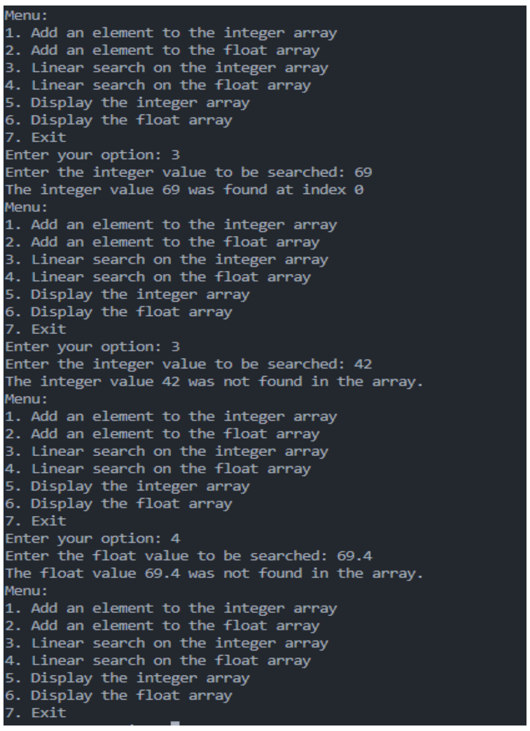
```
1
Enter the source file name:
./input.bin
Enter the destination file name:
./output.bin
Select an option:
1. Read the source and destination file names
2. Read user-defined text to be written to the source file
3. Write every alternate byte from the source to the destination file
4. Compare the properties of the file
5. Exit
2
Enter the text to be written to the source file:
hey
Select an option:
1. Read the source and destination file names
2. Read user-defined text to be written to the source file
3. Write every alternate byte from the source to the destination file
4. Compare the properties of the file
5. Exit
3
Every alternate byte has been written to the destination file.
Select an option:
1. Read the source and destination file names
2. Read user-defined text to be written to the source file
3. Write every alternate byte from the source to the destination file
4. Compare the properties of the file
5. Exit
4
Properties of ./input.bin:
Name: input.bin
Path: D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE\.\input.bin
Size: 53 bytes
Properties of ./output.bin:
Name: output.bin
Path: D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE\.\output.bin
Size: 27 bytes
Select an option:
1. Read the source and destination file names
2. Read user-defined text to be written to the source file
3. Write every alternate byte from the source to the destination file
4. Compare the properties of the file
5. Exit
```

[4] Write a menu-driven Java Program to create an ArrayList of (1) integers and (2) floats of user specified length. Write a set of overloaded methods to “add” and/or “remove” elements from the arrays and another set of overloaded methods to perform linear search on the arrays, given the key element. Create object(s) to demonstrate the above functionalities.

```java
//please make some legit changes and dont get caught if two or more programs are same
import java.util.ArrayList;
import java.util.Scanner;
public class Fourth {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the length of the integer array: ");
        int intLength = sc.nextInt();

        System.out.print("Enter the length of the float array: ");
        int floatLength = sc.nextInt();

        ArrayList<Integer> intList = new ArrayList<Integer>(intLength);
        ArrayList<Float> floatList = new ArrayList<Float>(floatLength);

        System.out.println("Menu:");
        System.out.println("1. Add an element to the integer array");
        System.out.println("2. Add an element to the float array");
        System.out.println("3. Linear search on the integer array");
        System.out.println("4. Linear search on the float array");
        System.out.println("5. Display the integer array");
        System.out.println("6. Display the float array");
        System.out.println("7. Exit");

        int option;
        do {
            System.out.print("Enter your option: ");
            option = sc.nextInt();

            switch (option) {
                case 1:
                    System.out.print("Enter the integer value to be added: ");
                    int intValue = sc.nextInt();
                    addElement(intList, intValue);
                    break;
                case 2:
                    System.out.print("Enter the float value to be added: ");
                    float floatValue = sc.nextFloat();
                    addElement(floatList, floatValue);
                    break;
                case 3:
                    System.out.print("Enter the integer value to be searched: ");
                    int intKey = sc.nextInt();
                    int intIndexFound = linearSearch(intList, intKey);
                    if (intIndexFound == -1) {
                        System.out.println("The integer value " + intKey + " was not found
in the array.");
                    } else {
                        System.out.println("The integer value " + intKey + " was found at
index " + intIndexFound);
                    }
                    break;
                case 4:
                    System.out.print("Enter the float value to be searched: ");
                    float floatKey = sc.nextFloat();
                    int floatIndexFound = linearSearch(floatList, floatKey);
                    if (floatIndexFound == -1) {
                        System.out.println("The float value " + floatKey + " was not found
in the array.");
                    } else {
                        System.out.println("The float value " + floatKey + " was found at
index " + floatIndexFound);
                    }
                    break;
                case 5:
                    System.out.println("The integer array is:");
                    displayArray(intList);
                    break;
                case 6:
                    System.out.println("The float array is:");
                    displayArray(floatList);
                    break;
                case 7:
                    System.out.println("Exiting program...");
                    break;
                default:
                    System.out.println("Invalid option, please try again.");
            }
        } while (option != 7);
    }

    // Overloaded method to add an element to an ArrayList of integers
    public static void addElement(ArrayList<Integer> arr, int index, int value) {
        arr.add(index, value);
    }

    // Overloaded method to add an element to an ArrayList of floats
```

```java
    public static void addElement(ArrayList<Float> arr, int index, float value) {
        arr.add(index, value);
    }
        // Overloaded method to perform linear search on an ArrayList of integers
    public static int linearSearch(ArrayList<Integer> arr, int key) {
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i) == key) {
                return i;
            }
        }
        return -1;
    }

    // Overloaded method to perform linear search on an ArrayList of floats
    public static int linearSearch(ArrayList<Float> arr, float key) {
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i) == key) {
                return i;
            }
        }
        return -1;
    }

    // Method to display an ArrayList of integers or floats
    public static <T> void displayArray(ArrayList<T> arr) {
        for (T element : arr) {
            System.out.print(element + " ");
        }
        System.out.println();
    }
}
```

Output:

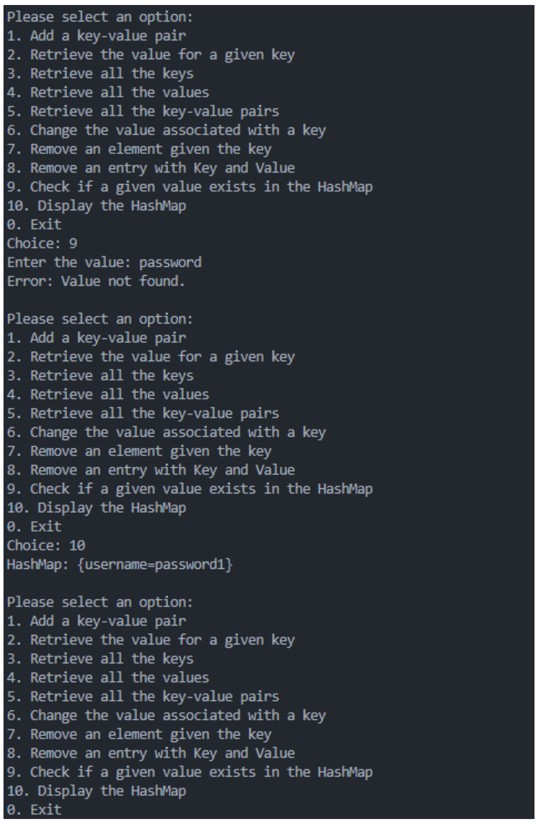


```
Menu:
1. Add an element to the integer array
2. Add an element to the float array
3. Linear search on the integer array
4. Linear search on the float array
5. Display the integer array
6. Display the float array
7. Exit
Enter your option: 3
Enter the integer value to be searched: 69
The integer value 69 was found at index 0
Menu:
1. Add an element to the integer array
2. Add an element to the float array
3. Linear search on the integer array
4. Linear search on the float array
5. Display the integer array
6. Display the float array
7. Exit
Enter your option: 3
Enter the integer value to be searched: 42
The integer value 42 was not found in the array.
Menu:
1. Add an element to the integer array
2. Add an element to the float array
3. Linear search on the integer array
4. Linear search on the float array
5. Display the integer array
6. Display the float array
7. Exit
Enter your option: 4
Enter the float value to be searched: 69.4
The float value 69.4 was not found in the array.
Menu:
1. Add an element to the integer array
2. Add an element to the float array
3. Linear search on the integer array
4. Linear search on the float array
5. Display the integer array
6. Display the float array
7. Exit
```

[5] Write a menu-driven Java Program to create a HashMap to store key-value pairs of login credentials. The menu options to be provided are for : adding a key-value pair, retrieve the “value” for a given “key” (first check if the specified key is present), retrieve all the keys, retrieve all the values, retrieve all the key-value pairs, change the value associated with a key in a HashMap, remove a HashMap element given the key, remove a HashMap entry with Key and Value, check if a given “value” exists in the Hashmap and display the HashMap. Read user input where required and display suitable error messages where applicable.

```java
// please make some legit changes and dont get caught if two or more programs are same
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
public class Five {
    public static void main(String[] args) {
        Map<String, String> credentials = new HashMap<>();
        Scanner scanner = new Scanner(System.in);
        int choice = 0;
        do {
            System.out.println("\nPlease select an option:");
            System.out.println("1. Add a key-value pair");
            System.out.println("2. Retrieve the value for a given key");
            System.out.println("3. Retrieve all the keys");
            System.out.println("4. Retrieve all the values");
            System.out.println("5. Retrieve all the key-value pairs");
            System.out.println("6. Change the value associated with a key");
            System.out.println("7. Remove an element given the key");
            System.out.println("8. Remove an entry with Key and Value");
            System.out.println("9. Check if a given value exists in the HashMap");
            System.out.println("10. Display the HashMap");
            System.out.println("0. Exit");

            System.out.print("Choice: ");
            choice = scanner.nextInt();
            scanner.nextLine(); // consume the newline character

            switch (choice) {
                case 1:
                    System.out.print("Enter the key: ");
                    String key = scanner.nextLine();
                    System.out.print("Enter the value: ");
                    String value = scanner.nextLine();
                    if (credentials.containsKey(key)) {
                        System.out.println("Error: Key already exists.");
                    } else {
                        credentials.put(key, value);
                        System.out.println("Key-value pair added successfully.");
                    }
                    break;
                case 2:
                    System.out.print("Enter the key: ");
                    key = scanner.nextLine();
                    if (credentials.containsKey(key)) {
                        System.out.println("Value: " + credentials.get(key));
                    } else {
                        System.out.println("Error: Key not found.");
                    }
                    break;
                case 3:
                    System.out.println("Keys: " + credentials.keySet());
                    break;
                case 4:
                    System.out.println("Values: " + credentials.values());
                    break;
                case 5:
                    System.out.println("Key-value pairs: " + credentials);
                    break;
                case 6:
                    System.out.print("Enter the key: ");
                    key = scanner.nextLine();
                    if (credentials.containsKey(key)) {
                        System.out.print("Enter the new value: ");
                        value = scanner.nextLine();
                        credentials.put(key, value);
                        System.out.println("Value updated successfully.");
                    } else {
                        System.out.println("Error: Key not found.");
                    }
                    break;
                case 7:
                    System.out.print("Enter the key: ");
                    key = scanner.nextLine();
                    if (credentials.containsKey(key)) {
                        credentials.remove(key);
                        System.out.println("Key-value pair removed successfully.");
                    } else {
                        System.out.println("Error: Key not found.");
                    }
                    break;
                case 8:
                    System.out.print("Enter the key: ");
                    key = scanner.nextLine();
                    System.out.print("Enter the value: ");
                    value = scanner.nextLine();
                    if (credentials.containsKey(key) && credentials.get(key).equals(value))
{
                        credentials.remove(key);
                        System.out.println("Key-value pair removed successfully.");
                    } else {
                        System.out.println("Error: Key-value pair not found.");
                    }
                    break;
                case 9:
                    System.out.print("Enter the value: ");
                    value = scanner.nextLine();
                    if (credentials.containsValue(value)) {
                        System.out.println("Value exists in the HashMap.");
                    } else {
                        System.out.println("Error: Value not found.");
                    }
                    break;
                case 10:
                    System.out.println("HashMap: " + credentials);
                    break;
                case 0:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice.");
            }
        } while (choice != 0);
    }
}
```

Output:

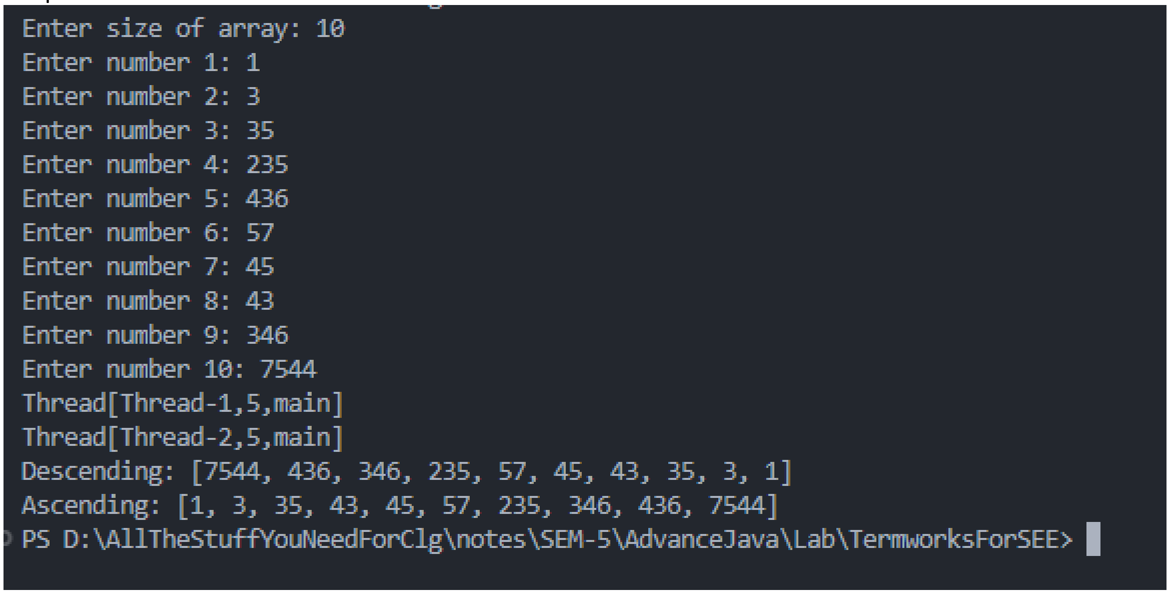


```
Please select an option:
1. Add a key-value pair
2. Retrieve the value for a given key
3. Retrieve all the keys
4. Retrieve all the values
5. Retrieve all the key-value pairs
6. Change the value associated with a key
7. Remove an element given the key
8. Remove an entry with Key and Value
9. Check if a given value exists in the HashMap
10. Display the HashMap
0. Exit
Choice: 9
Enter the value: password
Error: Value not found.

Please select an option:
1. Add a key-value pair
2. Retrieve the value for a given key
3. Retrieve all the keys
4. Retrieve all the values
5. Retrieve all the key-value pairs
6. Change the value associated with a key
7. Remove an element given the key
8. Remove an entry with Key and Value
9. Check if a given value exists in the HashMap
10. Display the HashMap
0. Exit
Choice: 10
HashMap: {username=password1}

Please select an option:
1. Add a key-value pair
2. Retrieve the value for a given key
3. Retrieve all the keys
4. Retrieve all the values
5. Retrieve all the key-value pairs
6. Change the value associated with a key
7. Remove an element given the key
8. Remove an entry with Key and Value
9. Check if a given value exists in the HashMap
10. Display the HashMap
0. Exit
```

[6] Write a multithreaded Java program to create a list of numbers and then sort the contents in ascending (thread 1) and descending (thread 2).

```java
//please make some legit changes and dont get caught if two or more programs are same
import java.util.*;
public class Six extends Thread{
    private int[] nums;
    public Six(int size) {
        nums = new int[size];
        for (int i = 0; i < size; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            nums[i] = new Scanner(System.in).nextInt();
        }
    }
    public void sortAscending() {
        Arrays.sort(nums);
        System.out.println("Ascending: " + Arrays.toString(nums));
    }
    public void sortDescending() {
        Arrays.sort(nums);
        for (int i = 0; i < nums.length / 2; i++) {
            int temp = nums[i];
            nums[i] = nums[nums.length - i - 1];
            nums[nums.length - i - 1] = temp;
        }
        System.out.println("Descending: " + Arrays.toString(nums));
    }
    public static void main(String[] args) {
        System.out.print("Enter size of array: ");
        int size = new Scanner(System.in).nextInt();
        tw5 st = new tw5(size);
        Thread thread1 = new Thread(new Runnable() {
            @Override
            public void run() {
                st.sortAscending();
            }
        });
        thread1.start();
        System.out.println(thread1);
        Thread thread2 = new Thread(new Runnable() {
            @Override
            public void run() {
                st.sortDescending();
            }
        });
        thread2.start();
        System.out.println(thread2);
    }
}
```

Output:

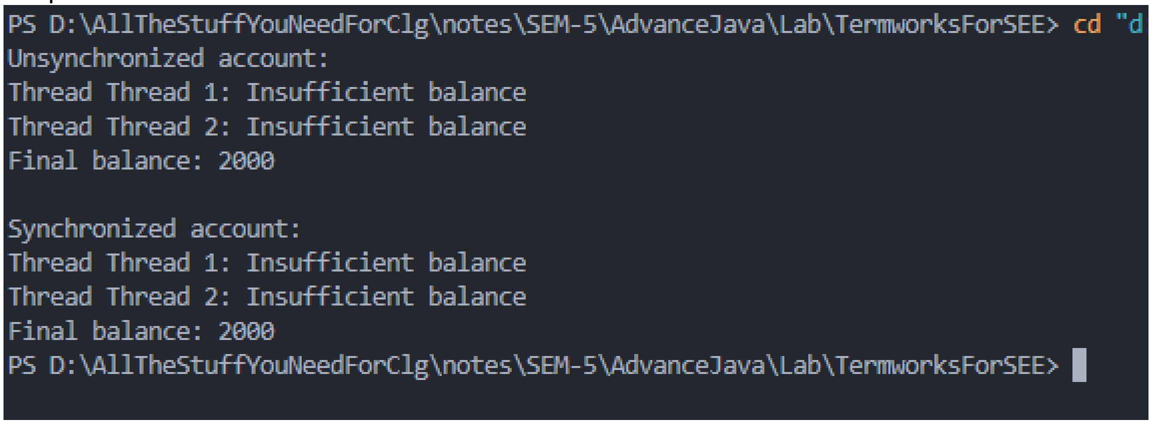


```
Enter size of array: 10
Enter number 1: 1
Enter number 2: 3
Enter number 3: 35
Enter number 4: 235
Enter number 5: 436
Enter number 6: 57
Enter number 7: 45
Enter number 8: 43
Enter number 9: 346
Enter number 10: 7544
Thread[Thread-1,5,main]
Thread[Thread-2,5,main]
Descending: [7544, 436, 346, 235, 57, 45, 43, 35, 3, 1]
Ascending: [1, 3, 35, 43, 45, 57, 235, 346, 436, 7544]
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE>
```

[7] Write a Java program to demonstrate how the standard operations on a bank account can be synchronized

```java
//please make some legit changes and dont get caught if two or more programs are same
import java.util.*;
public class Seven {
    public static void main(String[] args) {
        BankAccount unsyncAccount = new BankAccount();
        BankAccount syncAccount = new SynchronizedBankAccount();
        System.out.println("Unsynchronized account:");
        runThreads(unsyncAccount);
        System.out.println("Final balance: " + unsyncAccount.getBalance());
        System.out.println("\nSynchronized account:");
        runThreads(syncAccount);
        System.out.println("Final balance: " + syncAccount.getBalance());
    }
    private static void runThreads(BankAccount account) {
        Thread t1 = new Thread(() -> {
            account.deposit(1000);
            account.withdraw(1500);
        }, "Thread 1");
        Thread t2 = new Thread(() -> {
            account.deposit(1000);
            account.withdraw(1500);
        }, "Thread 2");
        t1.start();
        t2.start();
        try {
            t1.join();
            t2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
class BankAccount {
    private int balance = 0;
    private static final int MIN_BALANCE = 1000;
    void deposit(int amount) {
        balance += amount;
    }
    void withdraw(int amount) {
        if (balance - amount >= MIN_BALANCE) {
            balance -= amount;
        }
        else{
            System.out.println("Thread " + Thread.currentThread().getName() + ":
Insufficient balance");
        }
    }
    int getBalance() {
        return balance;
    }
}
class SynchronizedBankAccount extends BankAccount {
    @Override
    synchronized void deposit(int amount) {
        super.deposit(amount);
    }
    @Override
    synchronized void withdraw(int amount) {
        super.withdraw(amount);
    }
}
```

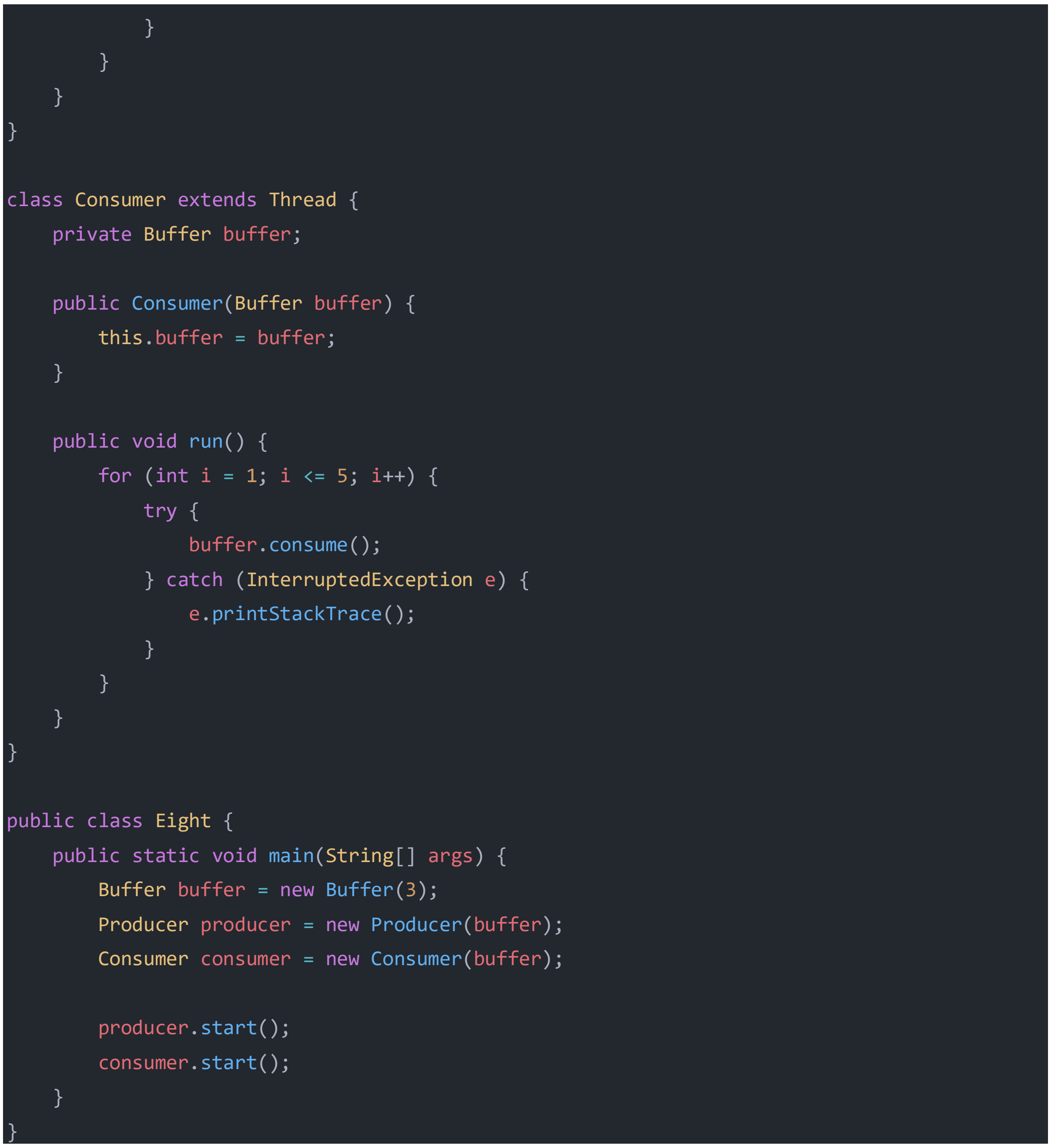
Output:



```
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE> cd "d
Unsynchronized account:
Thread Thread 1: Insufficient balance
Thread Thread 2: Insufficient balance
Final balance: 2000

Synchronized account:
Thread Thread 1: Insufficient balance
Thread Thread 2: Insufficient balance
Final balance: 2000
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE>
```

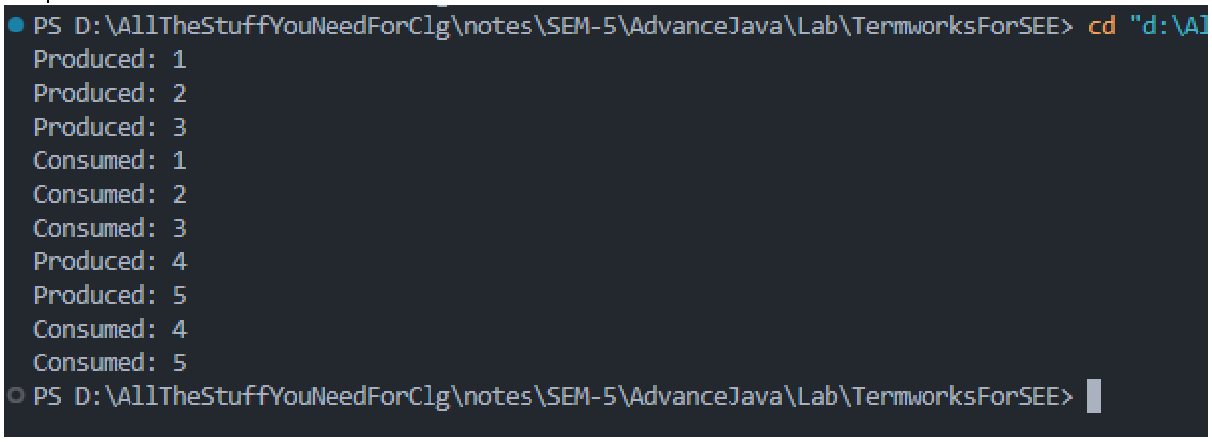
[8] Write a multithreaded Java program to demonstrate the Producer-Consumer problem.

```
//please make some legit changes and dont get caught if two or more programs are same
import java.util.LinkedList;
class Buffer {
    private LinkedList<Integer> buffer = new LinkedList<>();
    private int capacity;

    public Buffer(int capacity) {
        this.capacity = capacity;
    }

    public synchronized void produce(int item) throws InterruptedException {
        while (buffer.size() == capacity) {
            wait();
        }

        buffer.add(item);
        System.out.println("Produced: " + item);
        notify();
    }

    public synchronized int consume() throws InterruptedException {
        while (buffer.size() == 0) {
            wait();
        }

        int item = buffer.removeFirst();
        System.out.println("Consumed: " + item);
        notify();
        return item;
    }
}

class Producer extends Thread {
    private Buffer buffer;

    public Producer(Buffer buffer) {
        this.buffer = buffer;
    }

    public void run() {
        for (int i = 1; i <= 5; i++) {
            try {
                buffer.produce(i);
            } catch (InterruptedException e) {
                e.printStackTrace();
```



```java
            }
        }
    }
}

class Consumer extends Thread {
    private Buffer buffer;

    public Consumer(Buffer buffer) {
        this.buffer = buffer;
    }

    public void run() {
        for (int i = 1; i <= 5; i++) {
            try {
                buffer.consume();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class Eight {
    public static void main(String[] args) {
        Buffer buffer = new Buffer(3);
        Producer producer = new Producer(buffer);
        Consumer consumer = new Consumer(buffer);

        producer.start();
        consumer.start();
    }
}
```

Output:

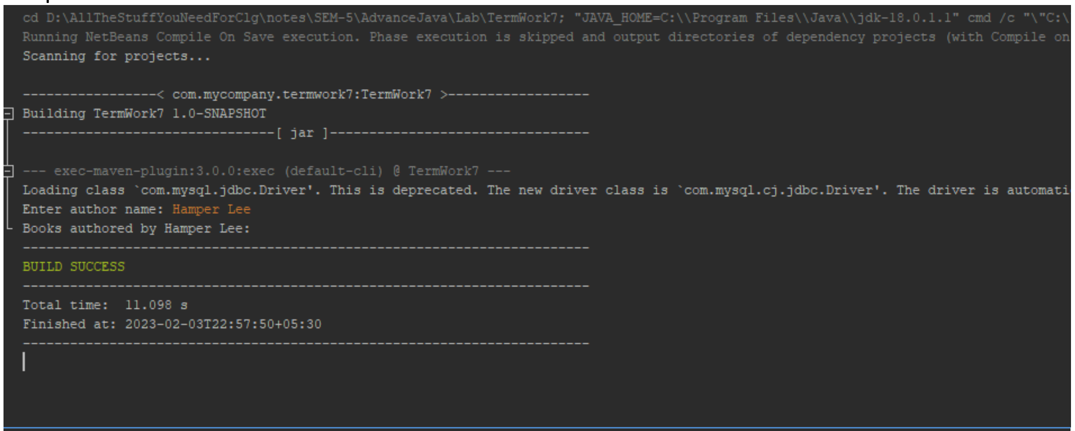


```
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE> cd "d:\Al
Produced: 1
Produced: 2
Produced: 3
Consumed: 1
Consumed: 2
Consumed: 3
Produced: 4
Produced: 5
Consumed: 4
Consumed: 5
PS D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermworksForSEE>
```

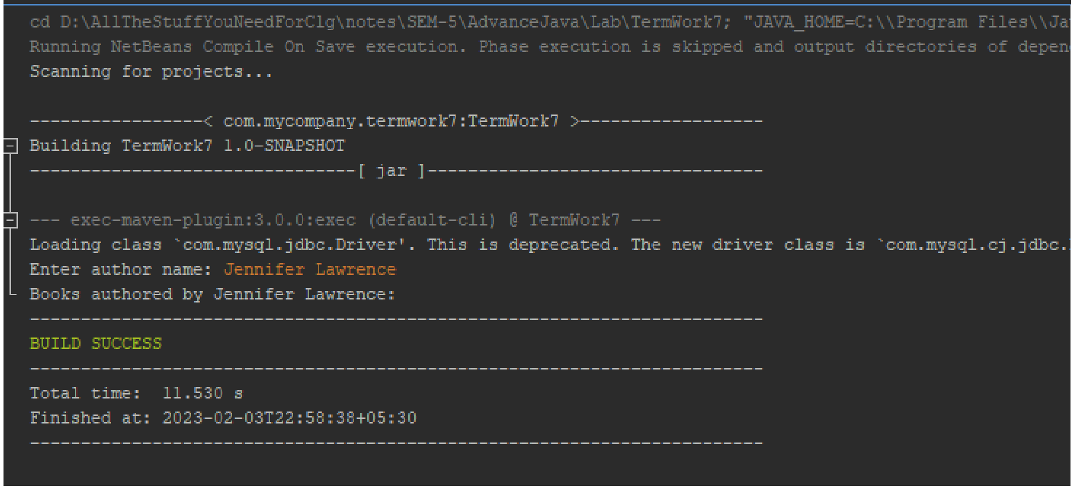
[9] Write a Java program to search and display details of book(s) authored by a particular author from a “BOOKS” table. Assume an appropriate structure and attributes for the table.

```
//please make some legit changes and dont get caught if two or more programs are same
//database create table statements:
//create table books(id int primary key, Bname text, year int);
//create table authors(id int primary key, Aname text, bid int, foreign key(bid) references
books(id));
import java.sql.*;
import java.util.Scanner;
public class tw7b {
    static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/database_name";
    static final String USER = "username";
    static final String PASS = "password";

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;

        try {
            Class.forName(JDBC_DRIVER);

            Scanner sc = new Scanner(System.in);
            System.out.print("Enter author name: ");
            String authorName = sc.nextLine();

            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();

            String sql = "SELECT books.Bname, books.year " +
                        "FROM books, authors " +
                        "WHERE authors.Aname='" + authorName + "' AND
books.id=authors.bid";
            ResultSet rs = stmt.executeQuery(sql);

            System.out.println("Books authored by " + authorName + ":");
            while (rs.next()) {
                String Bname = rs.getString("Bname");
                int year = rs.getInt("year");

                System.out.println("Name: " + Bname + ", Year: " + year);
            }
            rs.close();
            stmt.close();
            conn.close();
        } catch (SQLException se) {
```

```java
            se.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (stmt != null)
                    stmt.close();
            } catch (SQLException se2) {
                se2.printStackTrace();
            }
            try {
                if (conn != null)
                    conn.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}
```
Output:



```
cd D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermWork7; "JAVA_HOME=C:\\Program Files\\Java\\jdk-18.0.1.1" cmd /c "\"C:\
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on
Scanning for projects...

----------------< com.mycompany.termwork7:TermWork7 >-----------------
Building TermWork7 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ TermWork7 ---
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automati
Enter author name: Hamper Lee
Books authored by Hamper Lee:
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  11.098 s
Finished at: 2023-02-03T22:57:50+05:30
------------------------------------------------------------------------
```



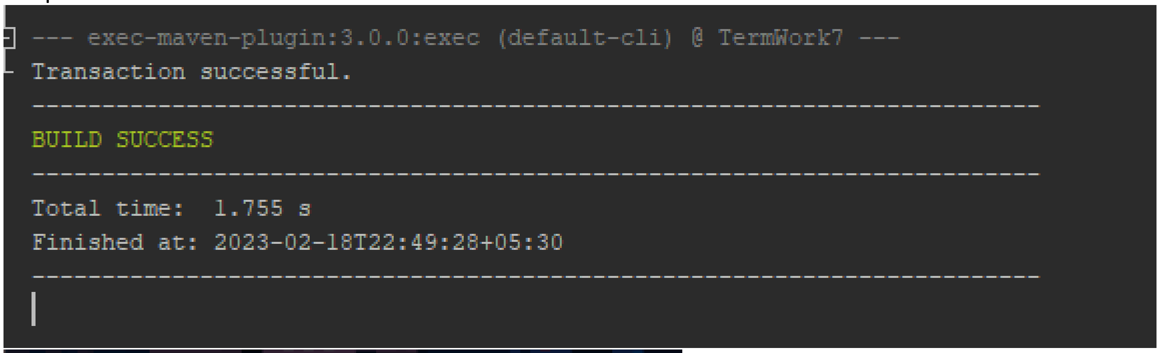
```
cd D:\AllTheStuffYouNeedForClg\notes\SEM-5\AdvanceJava\Lab\TermWork7; "JAVA_HOME=C:\\Program Files\\Ja
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of depen
Scanning for projects...

----------------< com.mycompany.termwork7:TermWork7 >-----------------
Building TermWork7 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ TermWork7 ---
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.
Enter author name: Jennifer Lawrence
Books authored by Jennifer Lawrence:
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  11.530 s
Finished at: 2023-02-03T22:58:38+05:30
------------------------------------------------------------------------
```

[10] Program to demonstrate transaction processing. Assume an appropriate database/table.

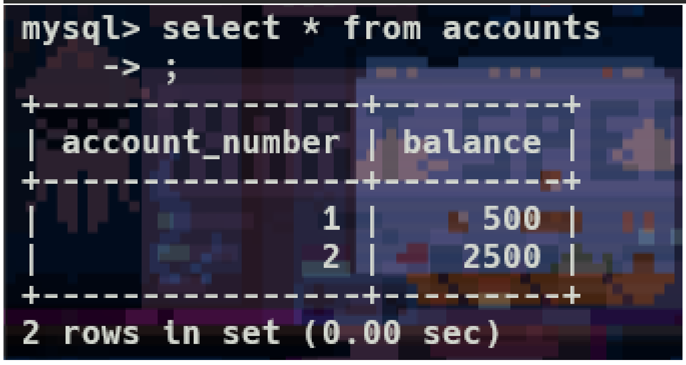
```
//please make some legit changes and dont get caught if two or more programs are same
//create table accounts (account_number int primary key, balance double);
import java.sql.*;
public class Ten {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/database_name"; // database URL
        String user = "root"; // database username
        String password = "password"; // database password

        Connection conn = null;
        try {
            conn = DriverManager.getConnection(url, user, password); // connect to the
database
            conn.setAutoCommit(false); // start a transaction

            // insert initial data into the accounts table
            PreparedStatement stmt = conn.prepareStatement("INSERT INTO accounts
(account_number, balance) VALUES (?, ?)");
            stmt.setInt(1, 1);
            stmt.setDouble(2, 1000.0);
            stmt.executeUpdate();
            stmt.setInt(1, 2);
            stmt.setDouble(2, 2000.0);
            stmt.executeUpdate();
            stmt.close();

            // withdraw 500 from account 1
            PreparedStatement withdrawStmt = conn.prepareStatement("UPDATE accounts SET
balance = balance - ? WHERE account_number = ?");
            withdrawStmt.setDouble(1, 500.0);
            withdrawStmt.setInt(2, 1);
            withdrawStmt.executeUpdate();
            withdrawStmt.close();

            // deposit 500 to account 2
            PreparedStatement depositStmt = conn.prepareStatement("UPDATE accounts SET
balance = balance + ? WHERE account_number = ?");
            depositStmt.setDouble(1, 500.0);
            depositStmt.setInt(2, 2);
            depositStmt.executeUpdate();
            depositStmt.close();

            conn.commit(); // commit the transaction
            System.out.println("Transaction successful.");
        } catch (SQLException e) {
```

```java
            try {
                conn.rollback(); // rollback the transaction if there was an error
            } catch (SQLException e1) {
                e1.printStackTrace();
            }
            System.err.println("Transaction failed: " + e.getMessage());
        } finally {
            try {
                conn.close(); // close the connection to the database
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Output:



```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ TermWork7 ---
Transaction successful.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.755 s
Finished at: 2023-02-18T22:49:28+05:30
------------------------------------------------------------------------
```

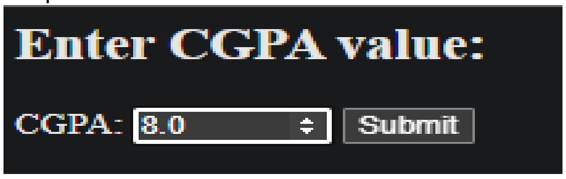


```
mysql> select * from accounts
    -> ;
+----------------+---------+
| account_number | balance |
+----------------+---------+
|              1 |     500 |
|              2 |    2500 |
+----------------+---------+
2 rows in set (0.00 sec)
```

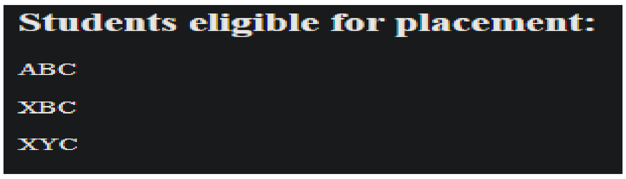
[11] Write a servlet program that allows receives a CGPA score and retrieves (from a db) the list of students who are eligible for a particular placement drive?

```
//please make some legit changes and dont get caught if two or more programs are same

//create table student (id int primary key, name varchar(50) cgpa decimal(3,2));
// load some data into database before executing
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/placement")
public class Eleven extends HttpServlet {
    private static final long serialVersionUID = 1L;

    private Connection connect() throws ClassNotFoundException, SQLException {
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/termwork_seven", "root",
"2gi20cs181");
        return conn;
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println("<html><body>");
        out.println("<h2>Enter CGPA value:</h2>");
        out.println("<form method=\"post\">");
        out.println("  <label for=\"cgpa\">CGPA:</label>");
        out.println("  <input type=\"number\" step=\"0.01\" min=\"0\" max=\"10\"
name=\"cgpa\" id=\"cgpa\" required>");
        out.println("  <button type=\"submit\">Submit</button>");
        out.println("</form>");
        out.println("</body></html>");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String cgpaString = request.getParameter("cgpa");
        double cgpa = Double.parseDouble(cgpaString);

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try {
            Connection conn = connect();
            PreparedStatement stmt = conn.prepareStatement("SELECT name FROM student WHERE
cgpa >= ?");
            stmt.setDouble(1, cgpa);
            ResultSet rs = stmt.executeQuery();

            out.println("<html><body>");
            out.println("<h2>Students eligible for placement:</h2>");
            while (rs.next()) {
                out.println("<p>" + rs.getString("name") + "</p>");
            }
            out.println("</body></html>");

            rs.close();
            stmt.close();
            conn.close();
        } catch (ClassNotFoundException | SQLException e) {
            out.println("<html><body><h2>Error retrieving data from
database</h2></body></html>");
            e.printStackTrace(out);
        }
    }
}
```

Output:


Enter CGPA value:

CGPA: 8.0


Students eligible for placement:

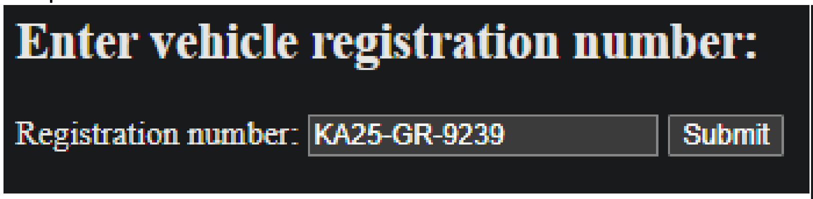
ABC

XBC

XYC

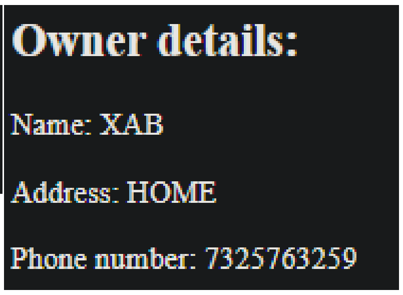
[12] Write a Servlet program that accepts a vehicle registration number and displays the owner's details (name, address, phone number). Assume that the details are stored in a database.

```
//please make some legit changes and dont get caught if two or more programs are same

//create table owners (regno varchar(20) primary key, name varchar(50), address text, phone
bigint);
// load some data into database before executing
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/vehicle")
public class Twelve extends HttpServlet {
    private static final long serialVersionUID = 1L;

    private Connection connect() throws ClassNotFoundException, SQLException {
        Class.forName("com.mysql.jdbc.Driver");
        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/database_name", "username",
"password");
        return conn;
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println("<html><body>");
        out.println("<h2>Enter vehicle registration number:</h2>");
        out.println("<form method=\"post\">");
        out.println("  <label for=\"regno\">Registration number:</label>");
        out.println("  <input type=\"text\" name=\"regno\" id=\"regno\" required>");
        out.println("  <button type=\"submit\">Submit</button>");
        out.println("</form>");
        out.println("</body></html>");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String regno = request.getParameter("regno");

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try {
            Connection conn = connect();
            PreparedStatement stmt = conn.prepareStatement("SELECT name, address, phone
FROM owners WHERE regno = ?");
            stmt.setString(1, regno);
            ResultSet rs = stmt.executeQuery();

            out.println("<html><body>");
            out.println("<h2>Owner details:</h2>");
            if (rs.next()) {
                out.println("<p>Name: " + rs.getString("name") + "</p>");
                out.println("<p>Address: " + rs.getString("address") + "</p>");
                out.println("<p>Phone number: " + rs.getString("phone") + "</p>");
            } else {
                out.println("<p>No owner found for registration number " + regno + "</p>");
            }
            out.println("</body></html>");
            rs.close();
            stmt.close();
            conn.close();
        } catch (ClassNotFoundException | SQLException e) {
            out.println("<html><body><h2>Error retrieving data from
database</h2></body></html>");
            e.printStackTrace(out);
        }
    }
}
```
Output:



Enter vehicle registration number:

Registration number: KA25-GR-9239 Submit



Owner details:

Name: XAB

Address: HOME

Phone number: 7325763259