```java
//Write a program to print the last modified date of file.

import java.io.File;

public class modi {
    public static void main(String[] args) {
        String filePath = "modi.java"; // Replace this with the actual file path
        File file = new File(filePath);
        if (file.exists()) {
    // Convert timestamp to human-readable format
    // Get the number of milliseconds since January 1, 1970, 00:00:00 GMT
        long milliseconds = file.lastModified();

    // Calculate seconds, minutes, hours, days since January 1, 1970, 00:00:00 GMT
        long seconds = milliseconds / 1000;
        long minutes = seconds / 60;
        long hours = minutes / 60;
        long days = hours / 24;

    // Extract individual components
        int day = (int) (days % 30) + 1;
        int month = (int) ((days / 30) % 12) + 1;
        int year = (int) (days / 365) + 1970;
        System.out.println("Last Modified Date: " + day + "/" + month + "/" + year);
        }
        else {
            System.out.println("File not found!");
        }
    }
}
-------------------------------------------------------------------------------------
-----------
/*ArrayDeque
In my restaurant I used to manage it well as per the order. I never want to skip any
order from the customer. So I prepare a rule: When I get the order I add it in the
last of my cook queue. And when the order is ready I used to pick it up from the
first of the cook queue.

So, write a java program with class Restaurant. Create a queue cook which contains
item names. Ask user weather he wants to Order or take food. If user press 1 then ask
for the name of the item and it should be added in the cook queue.
If user press 2 then The first item from the queue should be removed from the queue.
If user press 3 then he can see the items of the queue. If user press 4 then he
should move out from the Restaurant.
*/

import java.util.ArrayDeque;
import java.util.Deque;
import java.util.Scanner;

public class Restaurant {
    private Deque<String> cook;

    public Restaurant() {
        cook = new ArrayDeque<>();
    }

    public void takeOrder(String item) {
        cook.addLast(item);
```

```
53          System.out.println("Order added: " + item);
54      }
55
56      public void serveOrder() {
57          if (cook.isEmpty()) {
58              System.out.println("Cook queue is empty. No orders to serve.");
59          } else {
60              String servedItem = cook.removeFirst();
61              System.out.println("Served item: " + servedItem);
62          }
63      }
64
65      public void viewCookQueue() {
66          if (cook.isEmpty()) {
67              System.out.println("Cook queue is empty.");
68          } else {
69              System.out.println("Items in cook queue: " + cook);
70          }
71      }
72
73      public static void main(String[] args) {
74          Restaurant restaurant = new Restaurant();
75          Scanner scanner = new Scanner(System.in);
76
77          int choice;
78          do {
79              System.out.println("\n1. Order\n2. Take Food\n3. View Cook Queue\n4. Exit"
                );
80              System.out.print("Enter your choice: ");
81              choice = scanner.nextInt();
82
83              switch (choice) {
84                  case 1:
85                      System.out.print("Enter the name of the item to order: ");
86                      String item = scanner.next();
87                      restaurant.takeOrder(item);
88                      break;
89                  case 2:
90                      restaurant.serveOrder();
91                      break;
92                  case 3:
93                      restaurant.viewCookQueue();
94                      break;
95                  case 4:
96                      System.out.println("Thank you for visiting the restaurant!");
97                      break;
98                  default:
99                      System.out.println("Invalid choice. Please try again.");
100                     break;
101             }
102         } while (choice != 4);
103
104         scanner.close();
105     }
106 }
107
108 ------------------------------------------------------------------------------
    -----------
109 /*ArrayDeque
```

```
"Write a program that simulates a queue of customers at a coffee shop using an ArrayDeque. The program should allow the user to perform the following actions:

Add a new customer to the back of the queue
Serve the next customer in the queue (i.e. remove the customer from the front of the queue)
View the current queue of customers
The program should continue to prompt the user for actions until they choose to quit."
*/
import java.util.ArrayDeque;
import java.util.Scanner;

public class CoffeeShopQueueSimulation {
    public static void main(String[] args) {
        ArrayDeque<String> coffeeShopQueue = new ArrayDeque<>();
        Scanner scanner = new Scanner(System.in);
        int choice;

        do {
            System.out.println("Coffee Shop Queue Simulation");
            System.out.println("1. Add a new customer to the back of the queue");
            System.out.println("2. Serve the next customer in the queue");
            System.out.println("3. View the current queue of customers");
            System.out.println("4. Quit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();
            scanner.nextLine(); // Consume the newline character after reading the integer input

            switch (choice) {
                case 1:
                    System.out.print("Enter customer name: ");
                    String customerName = scanner.nextLine();
                    coffeeShopQueue.addLast(customerName);
                    System.out.println(customerName + " added to the queue.");
                    break;

                case 2:
                    if (!coffeeShopQueue.isEmpty()) {
                        String servedCustomer = coffeeShopQueue.removeFirst();
                        System.out.println(servedCustomer + " served and removed from the queue.");
                    } else {
                        System.out.println("Queue is empty. No customer to serve.");
                    }
                    break;

                case 3:
                    if (!coffeeShopQueue.isEmpty()) {
                        System.out.println("Current queue of customers: " + coffeeShopQueue);
                    } else {
                        System.out.println("Queue is empty. No customers in the queue.");
                    }
                    break;

                case 4:
                    System.out.println("Quitting the program.");
```

```
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
                    break;
            }

            System.out.println(); // Print an empty line for better readability

        } while (choice != 4);

        scanner.close();
    }
}
-------------------------------------------------------------------------------------------
/*ArrayDeque
Music player playlist: Write a program that simulates a music player playlist using an ArrayDeque. The program should allow the user to add songs to the playlist, play the next or previous song in the playlist, and shuffle the playlist. The program should keep track of the songs in the playlist using an ArrayDeque, where each element in the deque represents a song. When the user adds a song, you can use the addLast method to add it to the end of the playlist. When the user plays the next or previous song, you can use the removeFirst or removeLast method, respectively, to remove the current song from the front or back of the deque and add it to the end or beginning of the deque. When the user shuffles the playlist, you can use the shuffle method to randomly reorder the songs in the deque.
"Make the system as User interactive and give choices to user. */

import java.util.ArrayDeque;
import java.util.Collections;
import java.util.Scanner;

public class MusicPlayerPlaylist {
    public static void main(String[] args) {
        ArrayDeque<String> playlist = new ArrayDeque<>();
        Scanner scanner = new Scanner(System.in);
        int choice;

        do {
            System.out.println("\n-- Music Player Playlist --");
            System.out.println("1. Add song to playlist");
            System.out.println("2. Play next song");
            System.out.println("3. Play previous song");
            System.out.println("4. Shuffle playlist");
            System.out.println("5. Display playlist");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    scanner.nextLine(); // Consume newline character after nextInt()
                    System.out.print("Enter the song name: ");
                    String songName = scanner.nextLine();
                    playlist.addLast(songName);
                    System.out.println("Song added to the playlist.");
                    break;
                case 2:
```

```
                    if (!playlist.isEmpty()) {
                        String nextSong = playlist.removeFirst();
                        System.out.println("Playing next song: " + nextSong);
                        playlist.addLast(nextSong);
                    } else {
                        System.out.println("Playlist is empty. Add songs to the
                        playlist first.");
                    }
                    break;
                case 3:
                    if (!playlist.isEmpty()) {
                        String previousSong = playlist.removeLast();
                        System.out.println("Playing previous song: " + previousSong);
                        playlist.addFirst(previousSong);
                    } else {
                        System.out.println("Playlist is empty. Add songs to the
                        playlist first.");
                    }
                    break;
                case 4:
                    if (!playlist.isEmpty()) {
                        Collections.shuffle((ArrayDeque<String>) playlist);
                        System.out.println("Playlist shuffled.");
                    } else {
                        System.out.println("Playlist is empty. Add songs to the
                        playlist first.");
                    }
                    break;
                case 5:
                    if (!playlist.isEmpty()) {
                        System.out.println("Current playlist:");
                        for (String song : playlist) {
                            System.out.println(song);
                        }
                    } else {
                        System.out.println("Playlist is empty. Add songs to the
                        playlist first.");
                    }
                    break;
                case 6:
                    System.out.println("Goodbye!");
                    break;
                default:
                    System.out.println("Invalid choice. Please enter a valid option."
                    );
            }

        } while (choice != 6);

        scanner.close();
    }
}

-------------------------------------------------------------------------------
-----------
/*ArrayDeque
Web browser history: Write a program that simulates a web browser history using an
ArrayDeque. The program should allow the user to navigate back and forward through
their history of visited web pages, similar to how a web browser works. The program
```

```
should keep track of the URLs of visited web pages using an ArrayDeque, where each
element in the deque represents a visited web page. When the user navigates to a new
web page, you can push the URL onto the deque. When the user navigates back or
forward, you can pop URLs off the front or back of the deque, respectively."*/
import java.util.ArrayDeque;
import java.util.Deque;
import java.util.Scanner;

public class WebBrowserHistory {
    public static void main(String[] args) {
        Deque<String> webHistory = new ArrayDeque<>();
        Scanner scanner = new Scanner(System.in);
        String currentURL = "";

        while (true) {
            System.out.println("Enter the URL to visit or 'back' to navigate back,
            'forward' to navigate forward, or 'exit' to quit:");
            String input = scanner.nextLine();

            if (input.equalsIgnoreCase("exit")) {
                System.out.println("Exiting web browser history simulator.");
                break;
            } else if (input.equalsIgnoreCase("back")) {
                if (!webHistory.isEmpty()) {
                    // If there are URLs in history, navigate back by popping from
                    the front of the deque.
                    currentURL = webHistory.pollFirst();
                    System.out.println("Navigating back to: " + currentURL);
                } else {
                    System.out.println("No previous URLs in history.");
                }
            } else if (input.equalsIgnoreCase("forward")) {
                if (!webHistory.isEmpty()) {
                    // If there are URLs in history, navigate forward by popping from
                    the back of the deque.
                    currentURL = webHistory.pollLast();
                    System.out.println("Navigating forward to: " + currentURL);
                } else {
                    System.out.println("No forward URLs in history.");
                }
            } else {
                // If a new URL is entered, add it to the deque and update the
                current URL.
                webHistory.addFirst(currentURL);
                currentURL = input;
                System.out.println("Navigating to: " + currentURL);
            }
        }

        scanner.close();
    }
}

-------------------------------------------------------------------------------
-----------
/*HashMap   Write a program that reads in a list of names and corresponding phone
numbers from the user, and stores them in a HashMap. The program should then prompt
the user for a name and output the corresponding phone number, or a message
indicating that the name is not in the map.*/
```

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class PhoneBook {

    public static void main(String[] args) {
        // Create a HashMap to store names and phone numbers
        Map<String, String> phoneBook = new HashMap<>();

        // Read in a list of names and phone numbers from the user
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the number of contacts you want to add:");
        int numContacts = scanner.nextInt();
        scanner.nextLine(); // Consume the newline character after reading the integer

        for (int i = 0; i < numContacts; i++) {
            System.out.print("Enter name: ");
            String name = scanner.nextLine();
            System.out.print("Enter phone number: ");
            String phoneNumber = scanner.nextLine();
            phoneBook.put(name, phoneNumber);
        }

        // Prompt the user for a name and output the corresponding phone number
        System.out.print("\nEnter a name to find the corresponding phone number: ");
        String searchName = scanner.nextLine();
        String phoneNumber = phoneBook.get(searchName);

        if (phoneNumber != null) {
            System.out.println("Phone number for " + searchName + ": " + phoneNumber);
        } else {
            System.out.println("Name not found in the phone book.");
        }

        scanner.close();
    }
}


-------------------------------------------------------------------------------
-----------
/*HashMap
Write a program that reads in a list of words from a file and stores them in a
HashMap, along with their frequency (i.e. how many times they appear in the file).
The program should then prompt the user for a word and output the corresponding
frequency, or a message indicating that the word is not in the map.*/

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class WordFrequencyCounter {

    public static void main(String[] args) {
        HashMap<String, Integer> wordFrequencyMap = new HashMap<>();
```

```
365
366         // Read the file and populate the HashMap with word frequencies
367         readWordsFromFile("words.txt", wordFrequencyMap);
368
369         // Prompt the user for a word and display the frequency
370         Scanner scanner = new Scanner(System.in);
371         System.out.print("Enter a word to get its frequency: ");
372         String wordToCheck = scanner.nextLine().trim();
373
374         int frequency = wordFrequencyMap.getOrDefault(wordToCheck, 0);
375         if (frequency > 0) {
376             System.out.println("Frequency of \"" + wordToCheck + "\": " + frequency);
377         } else {
378             System.out.println("\"" + wordToCheck + "\" is not in the map.");
379         }
380     }
381
382     private static void readWordsFromFile(String filePath, Map<String, Integer>
    wordFrequencyMap) {
383         try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
384             String line;
385             while ((line = br.readLine()) != null) {
386                 String word = line.trim();
387                 wordFrequencyMap.put(word, wordFrequencyMap.getOrDefault(word, 0) + 1
                    );
388             }
389         } catch (IOException e) {
390             e.printStackTrace();
391         }
392     }
393 }
394
395
396 ------------------------------------------------------------------------------
    -----------
397 /*HashMap
398 Write a program that reads in a list of stock prices for a company and stores them in
    a HashMap, where the key is the date and the value is the price. The program should
    then compute the average price for the entire period and output it to the user.
399 */
400
401 import java.util.HashMap;
402 import java.util.Scanner;
403
404 public class StockPriceCalculator {
405     public static void main(String[] args) {
406         HashMap<String, Double> stockPrices = new HashMap<>();
407         Scanner scanner = new Scanner(System.in);
408
409         System.out.println("Enter stock prices for each date (date price), separated
            by space (e.g., '2023-07-23 100.50').");
410         System.out.println("Enter 'done' when finished:");
411
412         while (true) {
413             String input = scanner.nextLine();
414             if (input.equalsIgnoreCase("done")) {
415                 break;
416             }
417
```

```
            String[] data = input.split(" ");
            if (data.length != 2) {
                System.out.println("Invalid input format. Please use 'date price' format.");
                continue;
            }

            String date = data[0];
            double price;
            try {
                price = Double.parseDouble(data[1]);
            } catch (NumberFormatException e) {
                System.out.println("Invalid price format. Please enter a valid number.");
                continue;
            }
            stockPrices.put(date, price);
        }
        scanner.close();

        double totalPrice = 0.0;
        for (Double price : stockPrices.values()) {
            totalPrice += price;
        }
        double averagePrice = totalPrice / stockPrices.size();
        System.out.println("Average stock price for the period: " + averagePrice);
    }
}

-------------------------------------------------------------------------------------
/*HashSet
Write  a java program that reads in a list of integers from the user and stores them in a HashSet. The program should then compute the sum of all unique integers in the HashSet and output the result to the user. If the user enters the same integer multiple times, it should only be counted once in the sum.
*/
import java.util.HashSet;
import java.util.Scanner;

public class UniqueIntegerSum {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter a list of integers (one per line), and enter 'done' to finish:");

        HashSet<Integer> uniqueIntegers = new HashSet<>();
        String input;

        while (true) {
            input = scanner.nextLine().trim();
            if (input.equalsIgnoreCase("done")) {
                break;
            }

            try {
                int number = Integer.parseInt(input);
```

```
                uniqueIntegers.add(number);
            } catch (NumberFormatException e) {
                System.out.println("Invalid input. Please enter a valid integer or
                'done' to finish.");
            }
        }
        scanner.close();

        int sum = 0;
        for (int num : uniqueIntegers) {
            sum += num;
        }
        System.out.println("Sum of unique integers: " + sum);
    }
}

-------------------------------------------------------------------------------
-----------
/*HashSet
Write a program that reads in a list of words from the user and stores them in a
HashSet. The program should then prompt the user for a prefix and output all the
words in the set that start with that prefix.*/
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class PrefixMatcher {
    public static void main(String[] args) {
        Set<String> wordSet = new HashSet<>();
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter words (type 'exit' to stop):");

        // Read words from the user until they enter "exit"
        String word;
        while (!(word = scanner.nextLine().trim()).equalsIgnoreCase("exit")) {
            wordSet.add(word);
        }

        // Prompt for a prefix
        System.out.print("Enter a prefix to search for: ");
        String prefix = scanner.nextLine().trim();

        // Search and output words with the given prefix
        System.out.println("Words starting with prefix \"" + prefix + "\":");
        for (String w : wordSet) {
            if (w.startsWith(prefix)) {
                System.out.println(w);
            }
        }
    }
}


-------------------------------------------------------------------------------
-----------
/*Hashtable
Write a Java program that takes in a list of student names and their grades from the
user and stores them in a HashTable. The program should then prompt the user for a
```

```
student name and output their grade. If the student name is not found in the
HashTable, the program should output an error message.*/
import java.util.*;

public class StudentGradeLookup {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Hashtable<String, Double> studentGrades = new Hashtable<>();

        // Input student names and grades from the user
        System.out.println("Enter student names and their grades (type 'exit' to
        stop):");
        while (true) {
            System.out.print("Student name: ");
            String name = scanner.nextLine();
            if (name.equalsIgnoreCase("exit")) {
                break;
            }
            System.out.print("Grade: ");
            double grade = Double.parseDouble(scanner.nextLine());
            studentGrades.put(name, grade);
        }

        // Input student name and output grade
        System.out.print("Enter a student name to get their grade: ");
        String studentName = scanner.nextLine();

        // Look up the grade for the student name
        Double grade = studentGrades.get(studentName);
        if (grade != null) {
            System.out.println("Grade of " + studentName + ": " + grade);
        } else {
            System.out.println("Error: Student name not found.");
        }

        scanner.close();
    }
}

-------------------------------------------------------------------------------
-----------
/*Hashtable
"Write a Java program that implements a HashTable to store a dictionary of words and
their definitions. The program should prompt the user for a word and output its
definition. If the word is not found in the HashTable, the program should ask the
user if they would like to add the word and its definition to the HashTable.

The program should use the following methods of the HashTable class:

put(K key, V value): Adds a key-value pair to the HashTable. In this program, it
should be used to add new words and their definitions.

get(Object key): Retrieves the value associated with the given key from the
HashTable. In this program, it should be used to retrieve the definition of a word.

containsKey(Object key): Returns true if the HashTable contains a mapping for the
specified key. In this program, it should be used to check if a word is already in
the HashTable.
```

```
571
572 keySet(): Returns a Set of all keys in the HashTable. In this program, it should be
    used to print out a list of all words in the dictionary."
573 */
574 import java.util.HashMap;
575 import java.util.Scanner;
576 import java.util.Set;
577
578 public class DictionaryProgram {
579     public static void main(String[] args) {
580         HashMap<String, String> dictionary = new HashMap<>();
581         Scanner scanner = new Scanner(System.in);
582
583         // Adding some initial words and definitions to the dictionary
584         dictionary.put("apple", "a fruit that is red or green and is usually round");
585         dictionary.put("dog", "a domesticated mammal with four legs");
586         dictionary.put("book", "a written or printed work consisting of pages");
587
588         while (true) {
589             System.out.print("Enter a word to get its definition: ");
590             String word = scanner.nextLine();
591
592             if (dictionary.containsKey(word)) {
593                 String definition = dictionary.get(word);
594                 System.out.println("Definition: " + definition);
595             } else {
596                 System.out.println("Word not found in the dictionary.");
597                 System.out.print("Would you like to add it to the dictionary?
                    (yes/no): ");
598                 String answer = scanner.nextLine().trim().toLowerCase();
599
600                 if (answer.equals("yes")) {
601                     System.out.print("Enter the definition of the word: ");
602                     String definition = scanner.nextLine();
603                     dictionary.put(word, definition);
604                     System.out.println("Word and definition added to the dictionary."
                        );
605                 } else {
606                     System.out.println("Word not added to the dictionary.");
607                 }
608             }
609
610             System.out.print("Do you want to look up another word? (yes/no): ");
611             String anotherWord = scanner.nextLine().trim().toLowerCase();
612             if (anotherWord.equals("no")) {
613                 break;
614             }
615         }
616
617         // Printing out all the words in the dictionary
618         System.out.println("\nWords in the dictionary:");
619         Set<String> words = dictionary.keySet();
620         for (String word : words) {
621             System.out.println(word);
622         }
623
624         scanner.close();
625     }
626 }
```

```
627
628 ------------------------------------------------------------------------------
    -----------
629 /*Hashtable
630 "Write a Java program that creates a HashTable to store information about books in a
    library. The HashTable should use the book titles as keys and store information about
    each book, such as the author, publisher, and year of publication. The program should
    provide the following functionality:
631
632 Add a book: The program should prompt the user for a book title, author, publisher,
    and year of publication, and add the book to the HashTable.
633
634 Remove a book: The program should prompt the user for a book title and remove the
    corresponding entry from the HashTable.
635
636 Search for a book: The program should prompt the user for a book title and output the
    corresponding information stored in the HashTable, such as the author, publisher, and
    year of publication. If the book title is not found in the HashTable, the program
    should output an error message.
637
638 List all books: The program should list all books in the HashTable, along with their
    corresponding information.
639
640 To accomplish these tasks, you can use the following HashTable methods:
641
642 put(key, value): Inserts a key-value pair into the HashTable.
643 remove(key): Removes a key-value pair from the HashTable.
644 get(key): Returns the value corresponding to a given key in the HashTable.
645 containsKey(key): Returns true if the HashTable contains a given key.
646 keySet(): Returns a Set of all the keys in the HashTable."
647 */
648 import java.util.Hashtable;
649 import java.util.Set;
650 import java.util.Scanner;
651
652 class Library {
653     private Hashtable<String, Book> books;
654
655     public Library() {
656         books = new Hashtable<>();
657     }
658
659     public void addBook(String title, String author, String publisher, int year) {
660         Book book = new Book(title, author, publisher, year);
661         books.put(title, book);
662     }
663
664     public void removeBook(String title) {
665         books.remove(title);
666     }
667
668     public void searchBook(String title) {
669         if (books.containsKey(title)) {
670             Book book = books.get(title);
671             System.out.println("Title: " + book.title);
672             System.out.println("Author: " + book.author);
673             System.out.println("Publisher: " + book.publisher);
674             System.out.println("Year of Publication: " + book.year);
675         } else {
```

```
            System.out.println("Book not found in the library.");
        }
    }

    public void listAllBooks() {
        Set<String> bookTitles = books.keySet();
        for (String title : bookTitles) {
            Book book = books.get(title);
            System.out.println("Title: " + book.title);
            System.out.println("Author: " + book.author);
            System.out.println("Publisher: " + book.publisher);
            System.out.println("Year of Publication: " + book.year);
            System.out.println("--------------------");
        }
    }

    private static class Book {
        private String title;
        private String author;
        private String publisher;
        private int year;

        public Book(String title, String author, String publisher, int year) {
            this.title = title;
            this.author = author;
            this.publisher = publisher;
            this.year = year;
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Library library = new Library();
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("1. Add a book");
            System.out.println("2. Remove a book");
            System.out.println("3. Search for a book");
            System.out.println("4. List all books");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume the newline character after reading the
            integer.

            switch (choice) {
                case 1:
                    System.out.print("Enter the title: ");
                    String title = scanner.nextLine();
                    System.out.print("Enter the author: ");
                    String author = scanner.nextLine();
                    System.out.print("Enter the publisher: ");
                    String publisher = scanner.nextLine();
                    System.out.print("Enter the year of publication: ");
                    int year = scanner.nextInt();
                    scanner.nextLine(); // Consume the newline character after
                    reading the integer.
```

```
                    library.addBook(title, author, publisher, year);
                    System.out.println("Book added successfully.");
                    break;
                case 2:
                    System.out.print("Enter the title of the book to remove: ");
                    String bookToRemove = scanner.nextLine();
                    library.removeBook(bookToRemove);
                    System.out.println("Book removed successfully.");
                    break;
                case 3:
                    System.out.print("Enter the title of the book to search: ");
                    String bookToSearch = scanner.nextLine();
                    library.searchBook(bookToSearch);
                    break;
                case 4:
                    library.listAllBooks();
                    break;
                case 5:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    System.exit(0);
                default:
                    System.out.println("Invalid choice. Please try again.");
            }

            System.out.println();
        }
    }
}

-------------------------------------------------------------------------------------
-----------
/*PriorityQueue
Write a Java program that takes a list of integers as input from the user and stores
them in a PriorityQueue. The program should then remove and display the top three
highest integers from the PriorityQueue.*/
import java.util.*;

public class PriorityQueueExample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of integers you want to input: ");
        int n = scanner.nextInt();

        PriorityQueue<Integer> priorityQueue = new PriorityQueue<>(Collections.
        reverseOrder());

        System.out.println("Enter " + n + " integers:");
        for (int i = 0; i < n; i++) {
            int num = scanner.nextInt();
            priorityQueue.offer(num);
        }

        System.out.println("Top three highest integers:");

        for (int i = 0; i < 3; i++) {
            Integer highestInt = priorityQueue.poll();
            if (highestInt != null) {
                System.out.println(highestInt);
```

```
            } else {
                System.out.println("PriorityQueue is empty.");
            }
        }

        scanner.close();
    }
}

-------------------------------------------------------------------------------
-----------
/*PriorityQueue
Sports tournament organizer: Write a program that simulates a sports tournament using
a PriorityQueue. The program should allow the user to input team names and their
win-loss records. The program should then prioritize teams based on their win-loss
records and add them to the PriorityQueue. When it's time for the next match, the
program should remove the two highest priority teams from the PriorityQueue and
display their names.*/
import java.util.*;

class Team {
    private String name;
    private int wins;
    private int losses;

    public Team(String name, int wins, int losses) {
        this.name = name;
        this.wins = wins;
        this.losses = losses;
    }

    public String getName() {
        return name;
    }

    public int getWins() {
        return wins;
    }

    public int getLosses() {
        return losses;
    }

    public int getWinLossDifference() {
        return wins - losses;
    }
}

public class SportsTournamentSimulation {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        PriorityQueue<Team> teamsQueue = new PriorityQueue<>(Comparator.comparing(Team
        ::getWinLossDifference).reversed());

        // Input teams and their win-loss records
        System.out.println("Enter the number of teams:");
        int numTeams = scanner.nextInt();
        scanner.nextLine(); // Consume the new line

```

```
        for (int i = 0; i < numTeams; i++) {
            System.out.println("Enter team name:");
            String name = scanner.nextLine();
            System.out.println("Enter wins:");
            int wins = scanner.nextInt();
            System.out.println("Enter losses:");
            int losses = scanner.nextInt();
            scanner.nextLine(); // Consume the new line

            Team team = new Team(name, wins, losses);
            teamsQueue.add(team);
        }

        // Display the names of the two highest priority teams (next match)
        System.out.println("\nNext Match:");
        if (teamsQueue.size() < 2) {
            System.out.println("Not enough teams for a match.");
        } else {
            Team team1 = teamsQueue.poll();
            Team team2 = teamsQueue.poll();
            System.out.println(team1.getName() + " vs. " + team2.getName());
        }

        scanner.close();
    }
}
```