Java module 2

Exercises Day 5

1 Set	Library
Instructions	Implement a program to store book titles. The program should allow the user to register a new book, if the book already exists, it should not create a duplicate. The program should also allow the user to print the titles of the existing books. Use a Set to store the book titles.

Expected output	Would you like to (a) register a book or (b) see the titles of the existing books?
	>>> a
	Enter the name of the book:
	>>> Harry Potter
	Would you like to (a) register a new book or (b) see the titles of the existing books?
	>>> a
	Enter the name of the book:
	>>> 1984
	Would you like to (a) register a new book or (b) see the titles of the existing books?
	>>> b
	Book titles:
	Harry Potter
	1984
	Would you like to (a) register a new book or (b) see the titles of the existing books?
	>>> a
	Enter the name of the book:
	>>> Harry Potter
	Would you like to (a) register a new book or (b) see the titles of the existing books?
	>>> b
	Book titles:
	Harry Potter
	1984

Solution	

```
        Scanner scanner = new Scanner(System.in);

        while(true) {
            System.out.print("Would you like to (a)
register a book or (b) see the titles of the existing
books? ");
            String choice = scanner.nextLine();
            if (choice.equals("a")) {
                System.out.print("Enter the name of the
book: ");
                String bookName = scanner.nextLine();
                books.add(bookName);
            }
            if (choice.equals("b")) {
                System.out.println("Book titles:");
                for (String bookName : books) {
                    System.out.println(bookName);
                }
            }
        }
    }
}
```

<table>
<tr><td>2 Map</td><td>Days off</td></tr>
<tr><td>Instructions</td><td>Implement a program to allow employees to request days off.

To initialize the program, create a HashMap where the keys are the names of the employees and the values are how many days off they have left.

<table>
<tr><td>Employee</td><td>Days left</td></tr>
<tr><td>John Doe</td><td>3</td></tr>
<tr><td>Clara Green</td><td>25</td></tr>
<tr><td>Tim Grass</td><td>1</td></tr>
<tr><td>Laura Muller</td><td>2</td></tr>
</table>

To request days off, the employee should enter their name and how many days off they want. If they have enough days left, the system should grant the days off and subtract from their days left.</td></tr>
<tr><td>Expected output</td><td>Enter employee name:
>>> John Doe
Enter the amount of days off:
>>> 2
Days off granted. Days left for John Doe is 1.
Enter employee name:
>>> Tim Grass
Enter the amount of days off:
>>> 2
Days off NOT granted, not enough days left.</td></tr>
<tr><td>Solution</td><td>

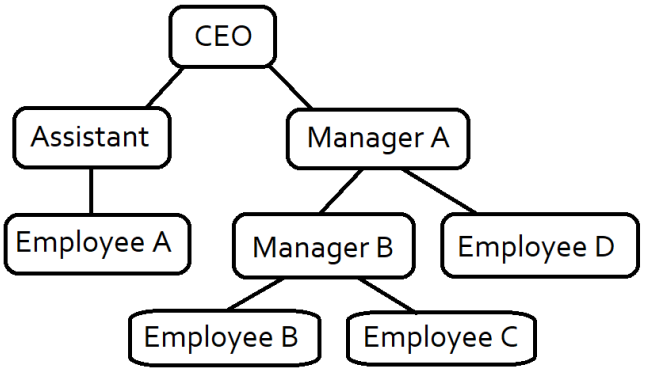
```
import java.util.Scanner;
import java.util.Map;
import java.util.HashMap;

public class Ex2 {
    public static void main(String[] args) {

        Map<String, Integer> daysOffMap =
createDaysOffMap();
        Scanner scanner = new Scanner(System.in);

        while(true){
            System.out.print("Enter the employee name: ");
            String employeeName = scanner.nextLine();
            System.out.print("Enter the amount of days
```

</td></tr>
</table>

```
requested: ");
            int daysRequested = scanner.nextInt();
            scanner.nextLine();

            int daysLeft = daysOffMap.get(employeeName);

            if (daysLeft >= daysRequested) {
                daysLeft -= daysRequested;
                daysOffMap.put(employeeName, daysLeft);
                System.out.println("Days off granted.
Amount of days left for " + employeeName + " is " +
daysLeft);
            } else {
                System.out.println("Days off NOT
granted.");
            }
        }
    }

    public static Map<String, Integer> createDaysOffMap() {
        Map<String, Integer> daysOffMap = new HashMap<>();
        daysOffMap.put("John Doe", 3);
        daysOffMap.put("Clara Green", 25);
        daysOffMap.put("Tim Grass", 1);
        daysOffMap.put("Laura Muller", 2);
        return daysOffMap;
    }
}
```

3 Tree	Draw the tree

Instructions	Run through the code by hand and draw the tree that will be created as a result of executing this code
	
	Then run it using the debugger and verify if your drawing is correct.
	
	TreeNode.java
	`import java.util.ArrayList;`
	`import java.util.List;`
	
	`public class TreeNode {`
	`  private String data;`
	`  private List<TreeNode> children;`
	
	`  public TreeNode(String data) {`
	`  this.data = data;`
	`  this.children = new ArrayList<>();`
	`  }`
	
	`  public void addChild(TreeNode childNode) {`
	`  this.children.add(childNode);`
	`  }`
	`}`
	
	CompanyHierarchy.java
	`public class CompanyHierarchy {`
	
	`  public static void main(String[] args) {`
	`  TreeNode ceoNode = new TreeNode("CEO");`
	`  TreeNode managerANode = new TreeNode("Manager A");`
	`  TreeNode managerBNode = new TreeNode("Manager B");`
	`  TreeNode assistantNode = new TreeNode("Assistant");`
	`  TreeNode employeeANode = new TreeNode("Employee A");`
	`  TreeNode employeeBNode = new TreeNode("Employee B");`
	`  TreeNode employeeCNode = new TreeNode("Employee C");`
	`  TreeNode employeeDNode = new TreeNode("Employee D");`
	
	`  managerBNode.addChild(employeeBNode);`
	`  managerBNode.addChild(employeeCNode);`
	`  assistantNode.addChild(employeeANode);`
	`  ceoNode.addChild(assistantNode);`
	`  ceoNode.addChild(managerANode);`
	`  managerANode.addChild(employeeDNode);`
	`  managerANode.addChild(managerBNode);`
	`  }`
	`}`

Solution	 CEO Assistant Manager A Employee A Manager B Employee D Employee B Employee C