Task (Chore) Management System

Rational

This is a system for managing household chores, in my family of 5 taking care of chores and making sure that the work is distributed among all family members can be a chaotic and challenging experience, this chore management system (CMS) will assist with this everyday issue.

Description

The application will have the following features:

1. Storage for the list of chores
2. Capability to add new chores as they become necessary
3. Capability to remove chores after they have been completed
4. Capability to print the chores that need to be completed

CMS Structure

I will implement the following classes for this software application:

1. Chore – the task that needs to be completed in the house.
2. CMS – chore management system.
3. ChoreTesting – a tester class that will be used to test the software.

Class Chore	
String	name
int	duration
String	importance

Class CMS	
List <Chore>	Storage of chores
CMS methods	
void addChore(Chore)	Adds a chore to the storage of chores
boolean removeChore(Chore)	Removes a chore form storage
void printChores()	Prints the chores in the storage

Class Chore

The class Chore has the following fields: name, duration, importance (importance indicates how urgent the chore is so that a family member knows which chore to complete first). The class is implemented in the following way:

```
package choreManagment;

public class Chore {
    private String name, importance;
    private int duration; // it is measured in the number of minutes
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
```

```
    public int getDuration() {
        return duration;
    }
    public void setDuration(int duration) {
        this.duration = duration;
    }
    public String getImportance() {
        return importance;
    }
    public void setImportance(String importance) {
        this.importance = importance;
    }
}
```

The chore class includes setters and getters for its 3 attributes: name, duration, and importance. All of the fields are set to private.

Class CMS

The CMS (Chore Management System) Class includes the storage for the list of chores that need to be done, it has 3 methods, one is for adding new chores, second for removing chores once someone has completed it, third is for printing what chores still need to be completed. It is implemented in the following way:

```
package choreManagment;

import java.util.ArrayList;
import java.util.List;
public class CMS {
    // Mapping with Book and the number of this book in the library
    private List<Chore> storage = new ArrayList<Chore>();
    // adds the book to the library
    public void addChore(Chore chore) {
        storage.add(chore);
    }
    // removes the book from the library
    public boolean removeChore(Chore chore) {
        boolean removed = false;
        for (int i = 0; i < storage.size(); i++) {
            Chore sampleChore = storage.get(i);
            if (sampleChore.getName().equals(chore.getName()) &&
                    sampleChore.getImportance().equals(chore.getImportance())
&&
                    sampleChore.getDuration() == (chore.getDuration())) {
                storage.remove(i);
                removed = true;
                break;
            }
        }
        return removed;
    }
    public void printStorage() {
        if (storage.isEmpty()) {
            System.out.println("The storage is empty");
        } else {
            for (Chore sampleChore: storage) {
                System.out.println(sampleChore.getName() + ", " +
```

```
sampleChore.getImportance() + ", " + sampleChore.getDuration() + " minutes");
                System.out.println();
            }
        }
    }
}
```

In comparing objects from our class I use '.equals' for strings and '==' for duration since it is an int measured in the approximate number of minutes that the chore will require. We use iterate over the storage array list.

CMS Tester Class

This is a class that tests the CMS, it adds new chores with all of their attributes and removes them after they have been completed

```
package choreManagment;
public class ChoresTester {
    public static void main(String[] args) {
        Chore chore1 = new Chore();
        chore1.setName("Doing the dishes");
        chore1.setDuration(35); //doing the dishes will require 35 minutes
        chore1.setImportance("moderately urgent");
        Chore chore2 = new Chore();
        chore2.setName("Clean up the living room");
        chore2.setDuration(15); //cleaning up the living room will require 15
minutes
        chore2.setImportance("not urgent");
        Chore chore3 = new Chore();
        chore3.setName("Doing laundry");
        chore3.setDuration(10);
        chore3.setImportance("very urgent");
        CMS cms = new CMS();
        cms.addChore(chore1);
        cms.addChore(chore2);
        cms.addChore(chore3);
        cms.removeChore(chore3); //chore 3 has been completed and therefore
removed from the list
        cms.printStorage();
    }
}
```

At the end the tester class prints the chores that need to be completed with all of their attributes.