Week 3 homework (Submit all codes to TutorJ)

Title: **Scheduler (week 3.5.1)**

Write a program that allows students to schedule appointments at either 1 or 2 pm. The scheduled time values must be of integer value from 1 to 2 (inclusive). Use an array of two strings to store the names in the available time slots. Write a loop that iterates as long as the array has a free space. Within a try block, allow the user to enter a time and a name. If the time is free, put the name in the array. If the time is not 1 or 2, throw an InvalidTimeExcception. If the time is not free, throw a TimeInUseException. Use a catch block for each different kind of exception.

After all the time-slots are filled, print the names stored in each time-slot.

```
import java.util.*;

public class Scheduler2   {

   public static void main(String[] args) {
      System.out.println("Welcome to the Appointment Scheduler");
      System.out.println("You can schedule an appointment at 1 or 2 pm.");

      Scanner reader = new Scanner(System.in);

      String appointments[] = new String[3];
      int appointmentsMade = 0;

      Type your codes here

}


public class InvalidTimeException extends Exception{

     Type your codes here

}

public class TimeInUseException extends Exception{


      Type your codes here
  }
```

Possible solution:

```
Welcome to the Appointment Scheduler
You can schedule an appointment at 1 or 2 pm.
What is your name?
Sam Loh
What time would you like the appointment?
1
What is your name?
Peter Low Ah Ting
What time would you like the appointment?
1
Appointment already made at that time
Sorry, that time is already in use
What is your name?
Sandy Tan
What time would you like the appointment?
7
Time value not in range
Sorry, that is not a legal time
What is your name?
Jay Chow Man Low
What time would you like the appointment?
2
All appointments made
At 1 pm: Sam Loh
At 2 pm: Jay Chow Man Low
```

Title: **SongCard (week 3.5.2)**

Create a class SongCard that represents a gift card for the purchase of songs online.

It should have the following private attributes:

- songs—the number of songs on the card
- activated—true if the card has been activated

and the following methods:

- SongCard(*n*)—a constructor for a card with maximum of *n* songs.

- activate()—activates the gift card.

- buyASong()—records the purchase of one song by decreasing the number of songs left for purchase. Throw an exception if the gift card is either completely used or not active.
- songsRemaining()—returns the number of songs that can be purchased.

In the main method, the following activities are implemented.

(a) Invoke the contructor, SongCard(*n*) with *n*=10 songs and set card activate to false.
(b) Try to buy a song. Use try … catch block to catch the exception if card is not activated. Print an error message.
(c) Activate the card by changing card activate from false to true

(d) Try to buy 11 songs. Use try …catch block to catch the exception if card is completely used. Print an error message.

```
import java.util.*;

public class SongCard {

   private int songs;
   private boolean activated;


   public SongCard(int n) {
     Type your codes here
   }

   public void activate(){
     Type your codes here
   }


   public void buyASong()
   throws CardNotActivatedException, CardEmptyException {
      Type your codes here
   }

   public int songsRemaining(){
     Type your codes here
   }



   public String toString(){
      Type your codes here
   }

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      SongCard sc = new SongCard(10);
      System.out.println(sc);

      System.out.println("Trying to buy a song");
      try{
         sc.buyASong();
      } catch (CardEmptyException e){
```

```
            System.out.println("Caught error: " +e.getMessage() );
        } catch (CardNotActivatedException e){
            System.out.println("Caught error: " +e.getMessage() );
        }
        System.out.println(sc);

        System.out.println("Activating the card");
        sc.activate();
        System.out.println(sc);

        System.out.println("Buying songs");

       Type your codes here
    }
}

public class CardNotActivatedException extends Exception{

    Type your codes here
    }

}
public class CardEmptyException extends Exception{

    Type your codes here
    }

}
```

Possible solution:

```
Card has 10 songs and is not activated

Trying to buy a song
Caught error: Card not activated
Card has 10 songs and is not activated
Activating the card
Card has 10 songs and is activated

Buying songs
Bought a song: Card has 9 songs and is activated
Bought a song: Card has 8 songs and is activated
Bought a song: Card has 7 songs and is activated
Bought a song: Card has 6 songs and is activated
Bought a song: Card has 5 songs and is activated
Bought a song: Card has 4 songs and is activated
Bought a song: Card has 3 songs and is activated
Bought a song: Card has 2 songs and is activated
Bought a song: Card has 1 songs and is activated
Bought a song: Card has 0 songs and is activated
Caught error: No more songs on the card
Card has 0 songs and is activated
```