3. An electric car that runs on batteries must be periodically recharged for a certain number of hours. The battery technology in the car requires that the charge time not be interrupted.

   The cost for charging is based on the hour(s) during which the charging occurs. A rate table lists the 24 one-hour periods, numbered from 0 to 23, and the corresponding hourly cost for each period. The same rate table is used for each day. Each hourly cost is a positive integer. A sample rate table is given below.

Hour	Cost	Hour	Cost	Hour	Cost
0	50	8	150	16	200
1	60	9	150	17	200
2	160	10	150	18	180
3	60	11	200	19	180
4	80	12	40	20	140
5	100	13	240	21	100
6	100	14	220	22	80
7	120	15	220	23	60

GO ON TO THE NEXT PAGE.

The class `BatteryCharger` below uses a rate table to determine the most economic time to charge the battery. You will write two of the methods for the `BatteryCharger` class.

```
public class BatteryCharger
{
    /** rateTable has 24 entries representing the charging costs for hours 0 through 23.  */
    private int[] rateTable;

    /** Determines the total cost to charge the battery starting at the beginning of startHour.
     *   @param startHour  the hour at which the charge period begins
     *           Precondition: 0 ≤ startHour ≤ 23
     *   @param chargeTime  the number of hours the battery needs to be charged
     *           Precondition: chargeTime > 0
     *   @return  the total cost to charge the battery
     */
    private int getChargingCost(int startHour, int chargeTime)
    {   /* to be implemented in part (a) */   }

    /** Determines start time to charge the battery at the lowest cost for the given charge time.
     *   @param chargeTime  the number of hours the battery needs to be charged
     *           Precondition: chargeTime > 0
     *   @return  an optimal start time, with  0 ≤ returned value ≤ 23
     */
    public int getChargeStartTime(int chargeTime)
    {   /* to be implemented in part (b) */   }

    //  There may be instance variables, constructors, and methods that are not shown.
}
```

(a) Write the `BatteryCharger` method `getChargingCost` that returns the total cost to charge a battery given the hour at which the charging process will start and the number of hours the battery needs to be charged.

For example, using the rate table given at the beginning of the question, the following table shows the resulting costs of several possible charges.

Start Hour of Charge	Hours of Charge Time	Last Hour of Charge	Total Cost
12	1	12	40
0	2	1	110
22	7	4 (the next day)	550
22	30	3 (two days later)	3,710

Note that a charge period consists of consecutive hours that may extend over more than one day.

Complete method `getChargingCost` below.

```
/** Determines the total cost to charge the battery starting at the beginning of startHour.
 *   @param startHour  the hour at which the charge period begins
 *          Precondition: 0 ≤ startHour ≤ 23
 *   @param chargeTime  the number of hours the battery needs to be charged
 *          Precondition: chargeTime > 0
 *   @return  the total cost to charge the battery
 */
private int getChargingCost(int startHour, int chargeTime)
```

GO ON TO THE NEXT PAGE.

(b) Write the `BatteryCharger` method `getChargeStartTime` that returns the start time that will allow the battery to be charged at minimal cost. If there is more than one possible start time that produces the minimal cost, any of those start times can be returned.

For example, using the rate table given at the beginning of the question, the following table shows the resulting minimal costs and optimal starting hour of several possible charges.

Hours of Charge Time	Minimum Cost	Start Hour of Charge	Last Hour of Charge
1	40	12	12
2	110	0 **or** 23	1 0 (the next day)
7	550	22	4 (the next day)
30	3,710	22	3 (two days later)

Assume that `getChargingCost` works as specified, regardless of what you wrote in part (a).

Complete method `getChargeStartTime` below.

```
/** Determines start time to charge the battery at the lowest cost for the given charge time.
 *   @param chargeTime  the number of hours the battery needs to be charged
 *           Precondition: chargeTime > 0
 *   @return an optimal start time, with  0 ≤ returned value ≤ 23
 */
public int getChargeStartTime(int chargeTime)
```

GO ON TO THE NEXT PAGE.