Electricity Bill Program in Java

In Java, there are various ways through which we can calculate electricity bills. We can calculate electricity bills using static values, command-line argument, method and functions, user-defined method, and do-while and for loop.

Let's understand each one of them one by one:

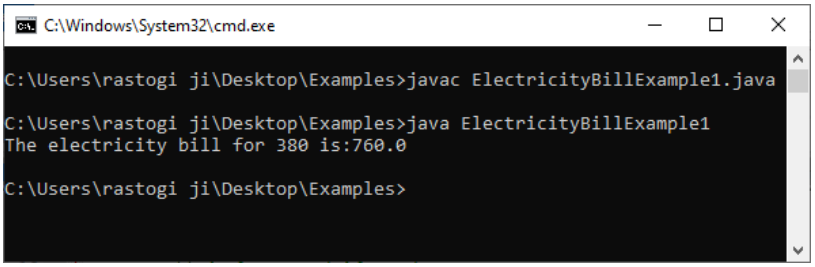
By Using the Static Method

In this way of calculating the electricity bill, we use the static method with static values, i.e., **units**. We use **if** and **else if** statements to check the number of units. Let's implement the logic for calculating the electricity bill by using the static method:

ElectricityBillExample1.java

```
// import required classes and package if any
// create class ElectricityBillExample1 to calculate electricity bill
class ElectricityBillExample1
{
   // main() method start
   public static void main(String args[])
   {
      // declare and initialize variable units
      int units = 380;
      // variable to calculate electricity bill to pay
      double billToPay = 0;
      // check whether units are less than 100
      if(units < 100)
      {
         billToPay = units * 1.20;
      }
      // check whether the units are less than 300
      else if(units < 300){
         billToPay = 100 * 1.20 + (units - 100) * 2;
      }
      // check whether the units are greater than 300
      else if(units > 300)
      {
         billToPay = 100 * 1.20 + 200 * 2 + (units - 300) * 3;
      }
      System.out.println("The electricity bill for " +units+ " is : " + billToPay);
   }
}
```

Output:



By Using the Scanner Class

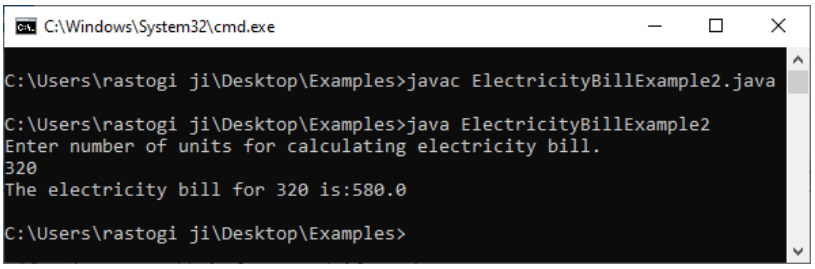
In this way of calculating the electricity bill, we also use the static method. The only difference between both of them is that we take input from the user for the number of units rather than using static values. Let's implement the logic for calculating the electricity bill by using the Scanner class:

ElectricityBillExample2.java

```
// import required classes and package if any
import java.util.*;
// create class ElectricityBillExample1 to calculate electricity bill
class ElectricityBillExample2
{
   // main() method start
   public static void main(String args[])
   {
      // declare variable units
      int units;

      // variable to calculate electricity bill to pay
      double billToPay = 0;

      // create Scanner class object to take input from user
      Scanner sc = new Scanner(System.in);

      System.out.println("Enter number of units for calculating electricity bill.");
      units = sc.nextInt();

      // check whether units are less than 100
      if(units < 100)
      {
         billToPay = units * 1.20;
      }
      // check whether the units are less than 300
      else if(units < 300){
         billToPay = 100 * 1.20 + (units - 100) * 2;
      }
      // check whether the units are greater than 300
      else if(units > 300)
      {
         billToPay = 100 * 1.20 + 200 * 2 + (units - 300) * 3;
      }
      System.out.println("The electricity bill for " +units+ " is : " + billToPay);
```

```
    }
}
```

Output:



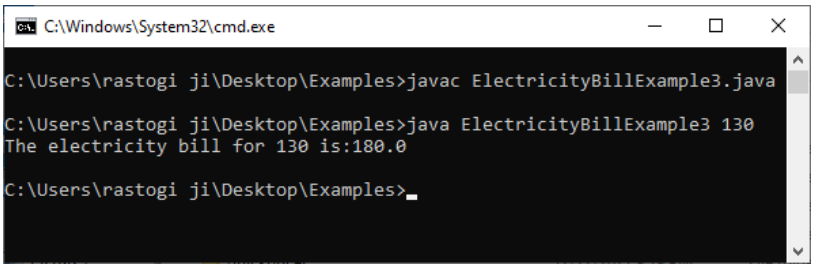
By Using Command Line Argument

In this way, we also use the same approach which we used for the previous ones. Here, we use the command line argument to take value for the number of units. Let's implement the logic for calculating the electricity bill by using the command line argument:

ElectricityBillExample3.java

```
// import required classes and package if any
// create class ElectricityBillExample3 to calculate electricity bill by taking input from command line argument
class ElectricityBillExample3
{
   // main() method start
   public static void main(String args[])
   {
      // declare variable units
      long units;

      units = Long.parseLong(args[0]);
      // variable to calculate electricity bill to pay
      double billToPay = 0;
      // check whether units are less than 100
      if(units < 100)
      {
         billToPay = units * 1.20;
      }
      // check whether the units are less than 300
      else if(units < 300){
         billToPay = 100 * 1.20 + (units - 100) * 2;
      }
      // check whether the units are greater than 300
      else if(units > 300)
      {
         billToPay = 100 * 1.20 + 200 * 2 + (units - 300) * 3;
      }
      System.out.println("The electricity bill for " +units+ " is:" + billToPay);
   }
}
```

Output:



By Using Inheritance

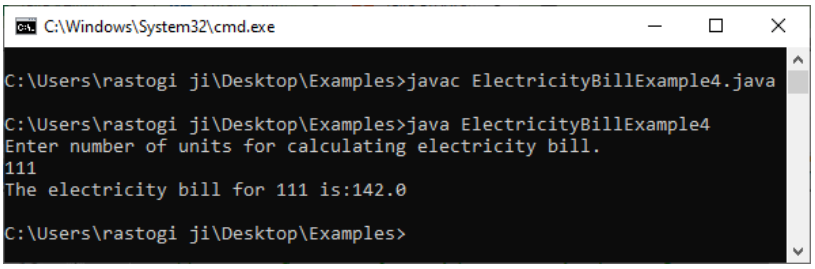
In this way of calculating electricity bills, we use the **Inheritance** concept of Oops. We create a child class, i.e., **CalculateBill,** that calculates the electricity bill for the user given input units. Let's implement the logic for calculating the electricity bill by using Inheritance.

ElectricityBillExample4.java

```
// import required classes and package if any
import java.util.*;

// create class ElectricityBillExample4 to calculate electricity bill using Inheritance
class ElectricityBillExample4 extends CalculateBill
{
   // main() method start
   public static void main(String args[])
   {
      // declare variable units
      int units;

      // create Scanner class object to take input from user
      Scanner sc = new Scanner(System.in);

      System.out.println("Enter number of units for calculating electricity bill.");
      units = sc.nextInt();

      System.out.println("The electricity bill for "+units+" is:" + getBill(units));
   }
}

// create simple class CalculateBill
class CalculateBill
{
   // variable to calculate electricity bill to pay
   static double billToPay;

   static double getBill(long units)
   {
      // check whether units are less than 100
      if(units < 100)
      {
         billToPay = units*1.20;
```

```
        }
        // check whether the units are less than 300
        else if(units < 300){
            billToPay = 100*1.20+(units - 100)*2;
        }
        // check whether the units are greater than 300
        else if(units > 300)
        {
            billToPay = 100*1.20+200*2+(units - 300)*3;
        }
        return billToPay;
    }
}
```

Output:



By Using a Separate Class (without Inheritance)

In this way, we create a new separate class, **CalculateElectricityBill**, for calculating electricity bills. In the parameterized constructor of this class, we calculate the electricity bill for the given number of units.

In the main() method of the main class, we create an object of the **CalculateElectricityBill** class and pass the number of units to its constructor.

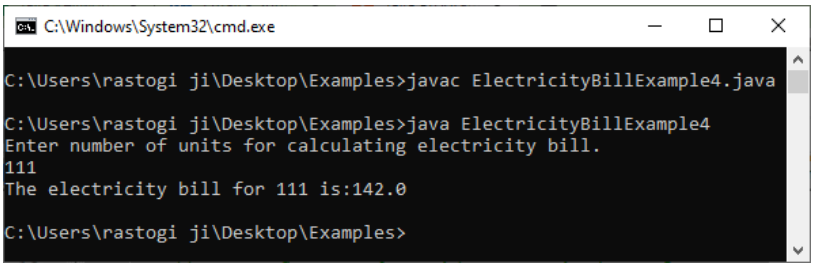
Let's implement the code to understand how we can use the separate class for calculating electricity bills:

ElectricityBillExample5.java

```
// import required classes and package if any
import java.util.*;
// create class ElectricityBillExample5 to calculate electricity bill using a separate class
class ElectricityBillExample5
{
    // main() method start
    public static void main(String args[])
    {
        // declare variable billToPay
        int units;
        // create Scanner class object to take input from user
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of units for calculating electricity bill.");
        units = sc.nextInt();
        // create an object of the CalculateElectricityBill class by passing number of units to its constructor
        CalculateElectricityBill obj = new CalculateElectricityBill(units);
        // access CalculateElectricityBill class variable through object
```

```
        System.out.println("The electricity bill for "+units+" is:" + obj.billToPay);
    }
}
// create seperate class CalculateElectricityBill
class CalculateElectricityBill
{
    // variable to calculate electricity bill to pay
    double billToPay;
    // constructor of CalculateElectricityBill class
    CalculateElectricityBill(long units)
    {
        // check whether units are less than 100
        if(units < 100)
        {
            billToPay = units*1.20;
        }
        // check whether the units are less than 300
        else if(units < 300){
            billToPay = 100*1.20+(units - 100)*2;
        }
        // check whether the units are greater than 300
        else if(units > 300)
        {
            billToPay = 100*1.20+200*2+(units - 300)*3;
        }
    }
}
```

Output:



By Using the User-Defined Method

It is one of the simplest ways to calculate electricity bills for the given number of units. In this approach, we put the code of calculating the electricity bill in a user-defined method and call it from anywhere in the code.

Let's implement the code to understand how we can use the user-defined method for calculating electricity bills:

ElectricityBillExample6.java

```
// import required classes and package if any
import java.util.*;

// create class ElectricityBillExample6 to calculate electricity bill using the user-defined method
```

```
class ElectricityBillExample6
{
   // main() method start
   public static void main(String args[])
   {
      // declare variable billToPay
      int units;

      // create Scanner class object to take input from user
      Scanner sc = new Scanner(System.in);

      System.out.println("Enter number of units for calculating electricity bill.");
      units = sc.nextInt();

      // call calculateBill() method and print the data returned from it
      System.out.println("The electricity bill for "+units+" is:" + calculateBill(units));
   }
   // create a user-defined method that calculates electricity bills for the given units
   static double calculateBill(double units)
   {
      // variable to calculate electricity bill to pay
      double billToPay = 0;

      // check whether units are less than 100
      if(units < 100)
      {
         billToPay = units*1.20;
      }
      // check whether the units are less than 300
      else if(units < 300){
         billToPay = 100*1.20+(units - 100)*2;
      }
      // check whether the units are greater than 300
      else if(units > 300)
      {
         billToPay = 100*1.20+200*2+(units - 300)*3;
      }

      // returned result
      return billToPay;
   }
}
```

Output:

```
C:\Windows\System32\cmd.exe

C:\Users\rastogi ji\Desktop\Examples>javac ElectricityBillExample6.java

C:\Users\rastogi ji\Desktop\Examples>java ElectricityBillExample6
Enter number of units for calculating electricity bill.
310
The electricity bill for 310 is:550.0

C:\Users\rastogi ji\Desktop\Examples>
```

All the above ways are used in different scenarios. Each and every method is helpful in calculating electricity bills, and each one has its own importance in Java.

← Prev Next →

Preparation

Aptitude
Aptitude

Logical Reasoning
Reasoning

Verbal Ability
Verbal Ability

Interview Questions
Interview Questions

Company Interview Questions
Company Questions

Trending Technologies

Artificial Intelligence
Artificial Intelligence

AWS Tutorial
AWS

Selenium tutorial
Selenium

Cloud Computing
Cloud Computing

Hadoop tutorial
Hadoop

ReactJS Tutorial
ReactJS

Data Science Tutorial
Data Science

Angular 7 Tutorial
Angular 7

Blockchain Tutorial
Blockchain

Git Tutorial
Git

Machine Learning Tutorial
Machine Learning

DevOps Tutorial
DevOps

B.Tech / MCA

DBMS tutorial
DBMS

Data Structures tutorial
Data Structures

DAA tutorial
DAA

Operating System
Operating System

Computer Network tutorial
Computer Network

Compiler Design tutorial
Compiler Design

Computer Organization and Architecture
Computer Organization

Discrete Mathematics Tutorial
Discrete Mathematics

Ethical Hacking
Ethical Hacking

Computer Graphics Tutorial
Computer Graphics

Software Engineering
Software Engineering

html tutorial
Web Technology

Cyber Security tutorial
Cyber Security

Automata Tutorial
Automata

C Language tutorial
C Programming

C++ tutorial
C++

Java tutorial
Java

.Net Framework tutorial
.Net

Python tutorial
Python

List of Programs
Programs

Control Systems tutorial
Control System

Data Mining Tutorial
Data Mining

Data Warehouse Tutorial
Data Warehouse