PROGRAM CODE:

```
import java.util.Scanner;
class Rectangle {
	private int length;
	private int breadth;
	public Rectangle() {
		System.out.println("Area of Rectangle");
	}
	public Rectangle(int l, int b) {
		length = l;
		breadth = b;
	}
	public void display() {
		System.out.println(length *breadth);
	}
}
public class Main {
	public static void main(String[] args) {
		Scanner scanner = new Scanner(System.in);
		Rectangle obj1 = new Rectangle();
		int length = scanner.nextInt();
		int breadth = scanner.nextInt();
		Rectangle obj2 = new Rectangle(length, breadth);
		obj2.display();
	}
}
```

Sample Input:

```
5

10
```

Sample Output:

```
Area of Rectangle

50
```

PROGRAM CODE:

```
import java.util.Scanner;
class Library {
  public void addBook(String title, String author, int publicationYear) {
    if (isValidString(title) && isValidString(author) && isValidYear(publicationYear)) {
      System.out.println("Book added: " + title + " by " + author + " (" + publicationYear + ")");
    } else {
      System.out.println("Invalid input");
      return;
    }
  }

  public void addBook(String title, String author) {
    if (isValidString(title) && isValidString(author)) {
      System.out.println("Book added: " + title + " by " + author);
    } else {
      System.out.println("Invalid input");
      return;
    }
  }

  public void addBook(String title) {
    if (isValidString(title)) {
      System.out.println("Book added: " + title);
    } else {
      System.out.println("Invalid input");
      return;
    }
  }
```

```
    protected boolean isValidString(String input) {
        return input != null && !input.trim().isEmpty();
    }

    protected boolean isValidYear(int year) {
        return year >= 1000 && year <= 9999;
    }
}

class DigitalLibrary extends Library {
    public void addBook(String title, String author, int publicationYear) {
        if (isValidString(title) && isValidString(author) && isValidYear(publicationYear)) {
            System.out.println("Digital Book added: " + title + " by " + author + " (" + publicationYear + ")");
        } else {
            System.out.println("Invalid input");
            return;
        }
    }

    public void addBook(String title, String author, int publicationYear, String format, String downloadLink)
{
        if (isValidString(title) && isValidString(author) && isValidYear(publicationYear)
                && isValidFormat(format) && isValidString(downloadLink)) {
            System.out.println("Digital Book added: " + title + " by " + author + " (" + publicationYear + ")");
            System.out.println("Format: " + format + ", Download Link: " + downloadLink);
        } else {
            System.out.println("Invalid input");
            return;
        }
    }

    private boolean isValidFormat(String format) {
        String[] validFormats = {"PDF", "EPUB", "MOBI", "AZW", "TXT"};
        for (String validFormat : validFormats) {
            if (validFormat.equalsIgnoreCase(format)) {
                return true;
            }
        }
        return false;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        String title = scanner.nextLine();
        String author = scanner.nextLine();
        int publicationYear = scanner.nextInt();
        scanner.nextLine();

        Library library = new Library();
        library.addBook(title, author, publicationYear);



        String digitalTitle = scanner.nextLine();
        String digitalAuthor = scanner.nextLine();
        int digitalPublicationYear = scanner.nextInt();
        scanner.nextLine();
        String format = scanner.nextLine();
        String downloadLink = scanner.nextLine();

        DigitalLibrary digitalLibrary = new DigitalLibrary();
        digitalLibrary.addBook(digitalTitle, digitalAuthor, digitalPublicationYear, format, downloadLink);
    }
}
```

Sample Input:

```
To Kill a Mockingbird

Harper Lee

1960

Clean Code

Robert C. Martin

2008

PDF

https://example.com/download/CleanCode
```

Sample Output:

```
Book added: To Kill a Mockingbird by Harper Lee (1960)

Digital Book added: Clean Code by Robert C. Martin (2008)

Format: PDF, Download Link:

https://example.com/download/CleanCode
```

PROGRAM CODE:

```
import java.util.Scanner;


class Vehicle {
   private String make;
   private String model;

   public Vehicle(String make, String model) {
      if (make == null || make.isEmpty() || model == null || model.isEmpty()) {
         throw new IllegalArgumentException("Make and model must be non-empty strings.");
      }
      this.make = make;
      this.model = model;
   }

   public String displayInfo() {
      return "Vehicle: " + make + " " + model;
   }
}

class ElectricVehicle extends Vehicle {
   private double batteryCapacity;
   private boolean chargingStatus;

   public ElectricVehicle(String make, String model, double batteryCapacity, boolean chargingStatus) {
      super(make, model);
      if (batteryCapacity <= 0) {
         throw new IllegalArgumentException("Battery capacity must be a positive number.");
      }
```

```
        this.batteryCapacity = batteryCapacity;
        this.chargingStatus = chargingStatus;
    }

    @Override
    public String displayInfo() {
        return super.displayInfo() +
               "\nBattery Capacity: " + batteryCapacity + " kWh" +
               "\nCharging Status: " + (chargingStatus ? "Charging" : "Not Charging");
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input for Vehicle
        System.out.print("Enter vehicle make: ");
        String make = scanner.nextLine();
        System.out.print("Enter vehicle model: ");
        String model = scanner.nextLine();

        Vehicle vehicle = new Vehicle(make, model);
        System.out.println(vehicle.displayInfo());

        // Input for ElectricVehicle
        System.out.print("\nEnter electric vehicle make: ");
        make = scanner.nextLine();
        System.out.print("Enter electric vehicle model: ");
        model = scanner.nextLine();
        System.out.print("Enter battery capacity (kWh): ");
        double batteryCapacity = scanner.nextDouble();
        System.out.print("Enter charging status (true/false): ");
        boolean chargingStatus = scanner.nextBoolean();

        ElectricVehicle ev = new ElectricVehicle(make, model, batteryCapacity, chargingStatus);
        System.out.println(ev.displayInfo());

        scanner.close();
    }
}
```

Sample Input:

```
Tesla

Model 3

75

True
```

Sample Output:

```
Vehicle: Tesla Model 3

Battery Capacity: 75.0 kWh

Charging Status: Charging
```

PROGRAM CODE:

```
import java.util.Scanner;
import java.util.regex.Pattern;

// PaymentMethod Interface
interface PaymentMethod {
    boolean authorizePayment();
    String getTransactionStatus();
}

// Abstract Payment Class
abstract class Payment implements PaymentMethod {
    protected String paymentType;

    public String getPaymentType() {
        return paymentType;
    }

    public abstract String getPaymentDetails();
}

// CreditCardPayment Class
class CreditCardPayment extends Payment {
    private String cardNumber;
    private String cardHolder;
    private String expirationDate;

    public CreditCardPayment(String cardNumber, String cardHolder, String expirationDate) {
        if (!isValidCardNumber(cardNumber)) {
```

```
        throw new IllegalArgumentException("Invalid card number format.");
    }
    if (!isValidExpirationDate(expirationDate)) {
        throw new IllegalArgumentException("Invalid expiration date format.");
    }
    this.cardNumber = cardNumber;
    this.cardHolder = cardHolder;
    this.expirationDate = expirationDate;
    this.paymentType = "Credit Card";
}

private boolean isValidCardNumber(String cardNumber) {
    return Pattern.matches("\\d{4}-\\d{4}-\\d{4}-\\d{4}", cardNumber);
}

private boolean isValidExpirationDate(String expirationDate) {
    return Pattern.matches("(0[1-9]|1[0-2])/\\d{2}", expirationDate);
}

@Override
public boolean authorizePayment() {
    // Dummy authorization logic
    return true;
}

@Override
public String getTransactionStatus() {
    return authorizePayment() ? "Success" : "Failed";
}
```

```
    @Override
    public String getPaymentDetails() {
        return "Card Number: " + cardNumber +
                "\nCardholder: " + cardHolder +
                "\nExpiration Date: " + expirationDate;
    }
}

// PayPalPayment Class
class PayPalPayment extends Payment {
    private String email;
    private String transactionId;

    public PayPalPayment(String email, String transactionId) {
        if (!isValidEmail(email)) {
            throw new IllegalArgumentException("Invalid email format.");
        }
        if (transactionId == null || transactionId.isEmpty()) {
            throw new IllegalArgumentException("Transaction ID must be a non-empty alphanumeric
string.");
        }
        this.email = email;
        this.transactionId = transactionId;
        this.paymentType = "PayPal";
    }

    private boolean isValidEmail(String email) {
        return Pattern.matches("^[A-Za-z0-9+_.-]+@[A-Za-z0-9.-]+$", email);
    }

    @Override
    public boolean authorizePayment() {
        // Dummy authorization logic
        return true;
    }

    @Override
    public String getTransactionStatus() {
        return authorizePayment() ? "Success" : "Failed";
    }

    @Override
    public String getPaymentDetails() {
        return "Email: " + email +
               "\nTransaction ID: " + transactionId;
    }
}

// Main Class
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Choose Payment Method: 1 for Credit Card, 2 for PayPal");
        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        Payment payment = null;

        if (choice == 1) {
            System.out.println("Enter Credit Card details: card_number card_holder expiration_date");
            String cardNumber = scanner.next();
            String cardHolder = scanner.next();
            String expirationDate = scanner.next();
            payment = new CreditCardPayment(cardNumber, cardHolder, expirationDate);
        } else if (choice == 2) {
            System.out.println("Enter PayPal details: email transaction_id");
            String email = scanner.next();
            String transactionId = scanner.next();
            payment = new PayPalPayment(email, transactionId);
        } else {
            System.out.println("Invalid choice!");
            scanner.close();
            return;
        }

        System.out.println("Payment Method: " + payment.getPaymentType());
        System.out.println(payment.getPaymentDetails());
        System.out.println("Transaction Status: " + payment.getTransactionStatus());

        scanner.close();
    }
}
```

Sample Input:

```
1234-5678-9876-5432
JohnDoe
12/25
```

Sample Output:

```
Payment Method: Credit Card
Card Number: 1234-5678-9876-5432
Cardholder: JohnDoe
Expiration Date: 12/25
Transaction Status: Success
```