```
File  Edit  Format  Run  Options  Window  Help
from abc import ABC, abstractmethod

# Abstract class
class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

# Derived class Rectangle
class Rectangle(Shape):
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

# Derived class Circle
class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius


# Input
shape_type = input("Enter Shape (Rectangle/Circle): ")

if shape_type.lower() == "rectangle":
    length = int(input("Enter length: "))
    breadth = int(input("Enter breadth: "))
    shape = Rectangle(length, breadth)
    print("Area of Rectangle")
```

```python
from abc import ABC, abstractmethod

# Abstract class
class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

# Derived class Rectangle
class Rectangle(Shape):
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

# Derived class Circle
class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius


# Input
shape_type = input("Enter Shape (Rectangle/Circle): ")

if shape_type.lower() == "rectangle":
    length = int(input("Enter length: "))
    breadth = int(input("Enter breadth: "))
    shape = Rectangle(length, breadth)
    print("Area of Rectangle")
```

```python
from abc import ABC, abstractmethod

# Abstract class
class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

# Derived class Rectangle
class Rectangle(Shape):
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

# Derived class Circle
class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius


# Input
shape_type = input("Enter Shape (Rectangle/Circle): ")

if shape_type.lower() == "rectangle":
    length = int(input("Enter length: "))
    breadth = int(input("Enter breadth: "))
    shape = Rectangle(length, breadth)
    print("Area of Rectangle")
```

```python
from abc import ABC, abstractmethod

# Abstract class
class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

# Derived class Rectangle
class Rectangle(Shape):
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

# Derived class Circle
class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius


# Input
shape_type = input("Enter Shape (Rectangle/Circle): ")

if shape_type.lower() == "rectangle":
    length = int(input("Enter length: "))
    breadth = int(input("Enter breadth: "))
    shape = Rectangle(length, breadth)
    print("Area of Rectangle")
```

```python
from abc import ABC, abstractmethod

# Abstract class
class Shape(ABC):
    @abstractmethod
    def area(self):
        pass

# Derived class Rectangle
class Rectangle(Shape):
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

# Derived class Circle
class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return 3.14 * self.radius * self.radius


# Input
shape_type = input("Enter Shape (Rectangle/Circle): ")

if shape_type.lower() == "rectangle":
    length = int(input("Enter length: "))
    breadth = int(input("Enter breadth: "))
    shape = Rectangle(length, breadth)
    print("Area of Rectangle")
```

```
>
========== RESTART:
Enter Shape (Rectangle/Circle): Rectangle
Enter length: 10
Enter breadth: 5
Area of Rectangle
> length*breadth
50
>
```