```java
// Abstract class representing a generic Robber
abstract class HouseRobber {
    // Abstract methods to be implemented by child classes
    abstract int robRowHouses(int x[]);
    abstract int robRoundHouses(int x[]);
    abstract int robSquareHouses(int x[]);
    abstract int robMultiHouses(int x[]);

    // Non-abstract methods with default implementation
    void displayRobbingClassInfo() {
        System.out.println("MScAIML");
    }

    void expressLoveForMachineLearning() {
        System.out.println("I love Machine Learning");
    }
}

// Child class inheriting from HouseRobber
public class robber extends HouseRobber {
    // Common logic for robbing houses
    int commonRobberyLogic(int x[]) {
        // Check if exactly 4 houses are provided
        if (x.length != 4) {
            System.out.println("Only 4 houses accepted!");
            return -1;
        }

        // Calculate the total loot for two pairs of houses
        int lootPair1 = x[0] + x[2];
        int lootPair2 = x[1] + x[3];

        // Return the maximum loot from the two pairs
        return lootPair1 > lootPair2 ? lootPair1 : lootPair2;
    }

    // Implementation for robbing square houses using common logic
    int robSquareHouses(int x[]) {
        return commonRobberyLogic(x);
    }

    // Implementation for robbing row houses using modified logic
    int robRowHouses(int x[]) {
        // Make a small tweak since the first and last houses can be robbed
        return commonRobberyLogic(x) > x[0] + x[3] ? commonRobberyLogic(x) :
x[0] + x[3];
    }
```

```java
    // Implementation for robbing round houses using common logic
    int robRoundHouses(int x[]) {
        return commonRobberyLogic(x);
    }

    // Implementation for robbing multi houses using common logic
    int robMultiHouses(int x[]) {
        return commonRobberyLogic(x);
    }

    // Main method to test the HouseRobber functionality
    public static void main(String[] args) {
        // Create an instance of Lab4HouseRobber
        robber robberInstance = new robber();

        // Special input to demonstrate different results for row houses
        int houseValues[] = {5, 2, 6, 10};

        // Display results for different types of houses
        System.out.println("For square houses: " +
robberInstance.robSquareHouses(houseValues));
        System.out.println("For round houses: " +
robberInstance.robRoundHouses(houseValues));
        System.out.println("For row houses: " +
robberInstance.robRowHouses(houseValues));
        System.out.println("For multi houses: " +
robberInstance.robMultiHouses(houseValues));
    }
}
```

Output:

```
PS C:\Abhinav\Test\Java> javac robber.java
PS C:\Abhinav\Test\Java> java robber
For square houses: 12
For round houses: 12
For row houses: 15
For multi houses: 12
PS C:\Abhinav\Test\Java>
```