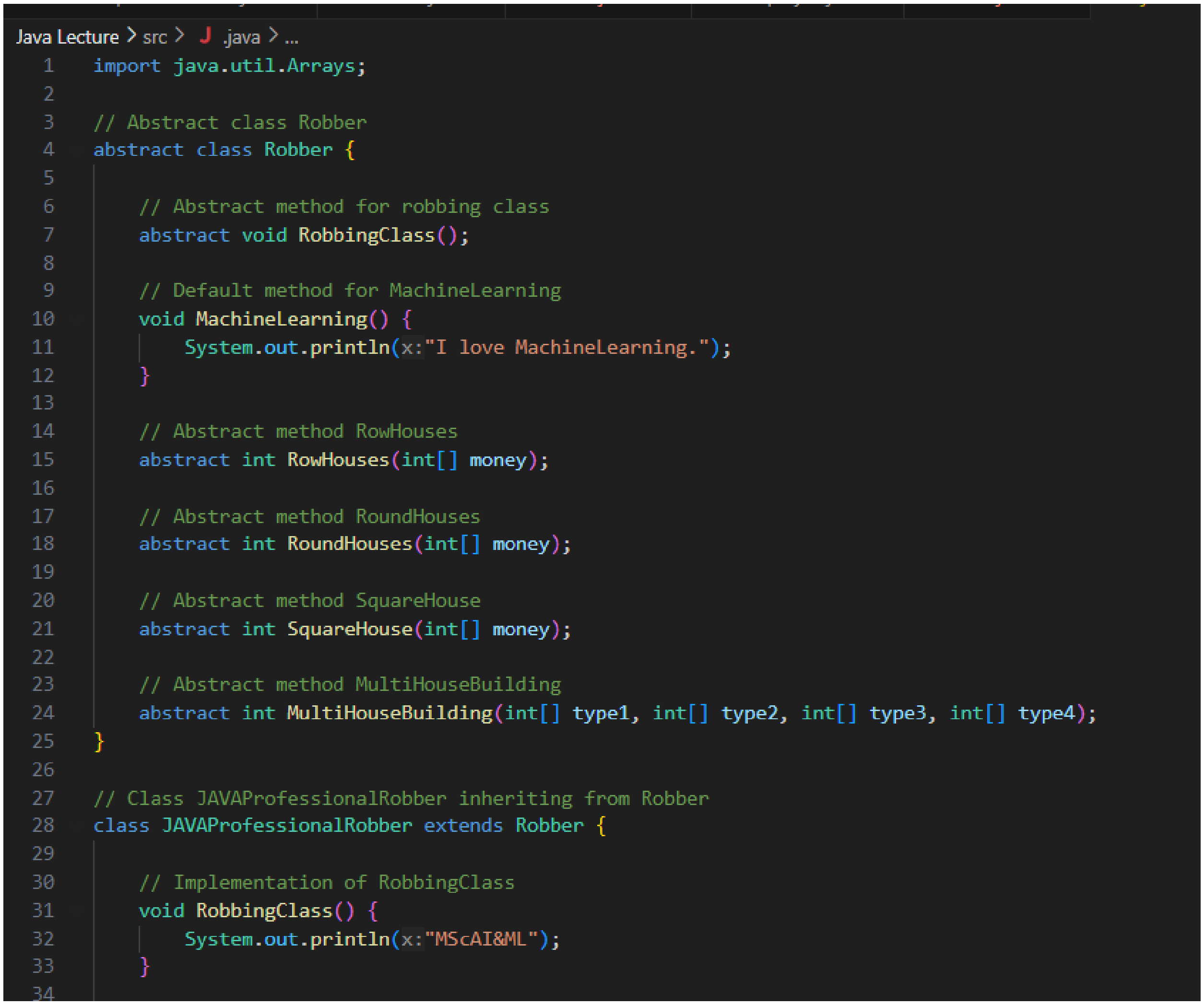
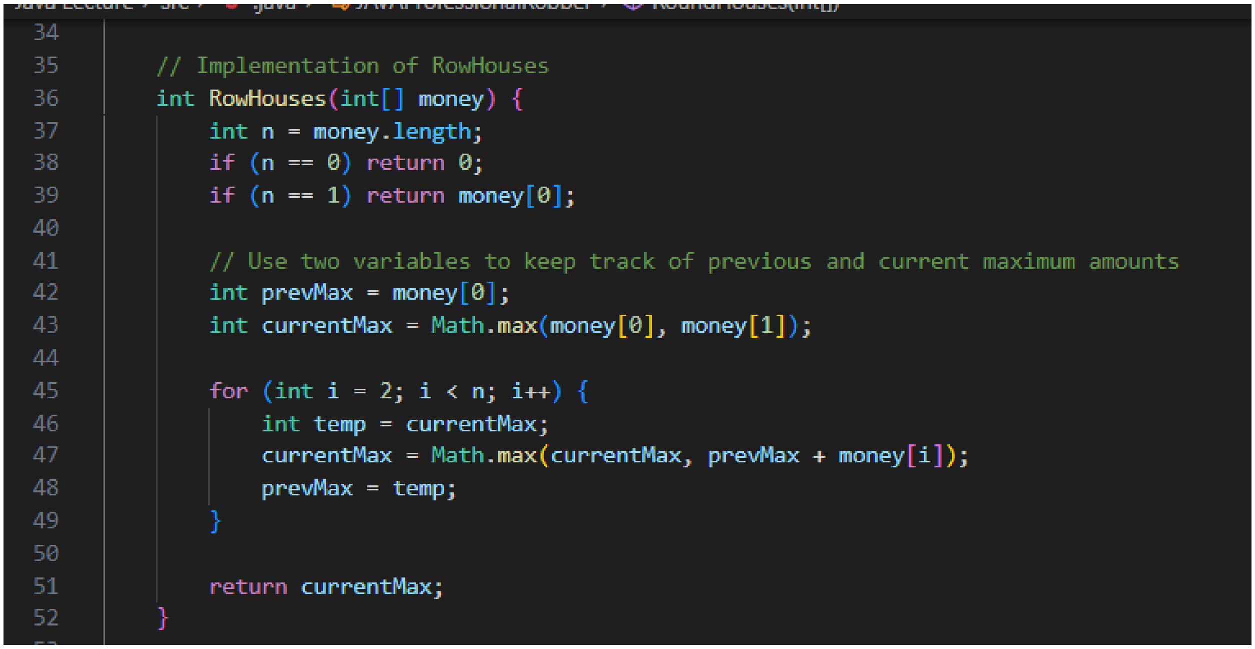
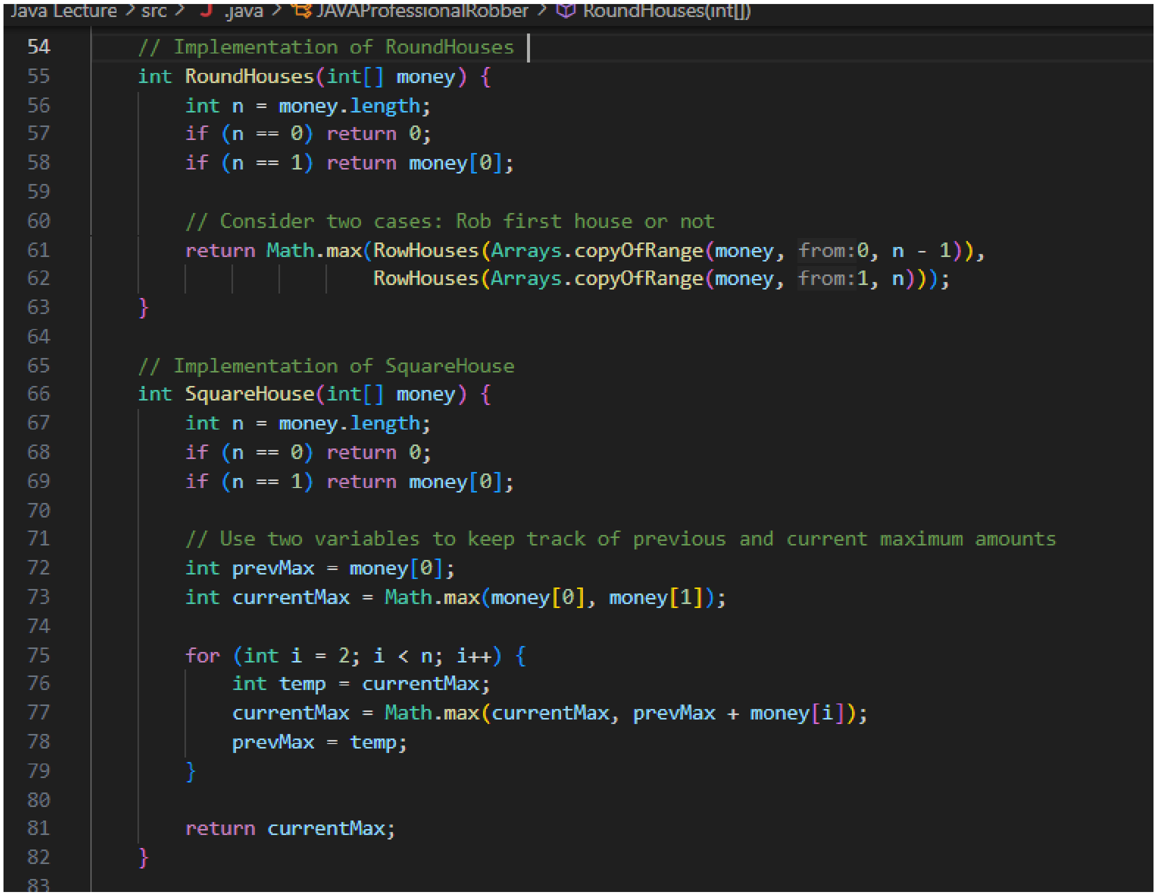


Java Lecture > src > J .java > ...

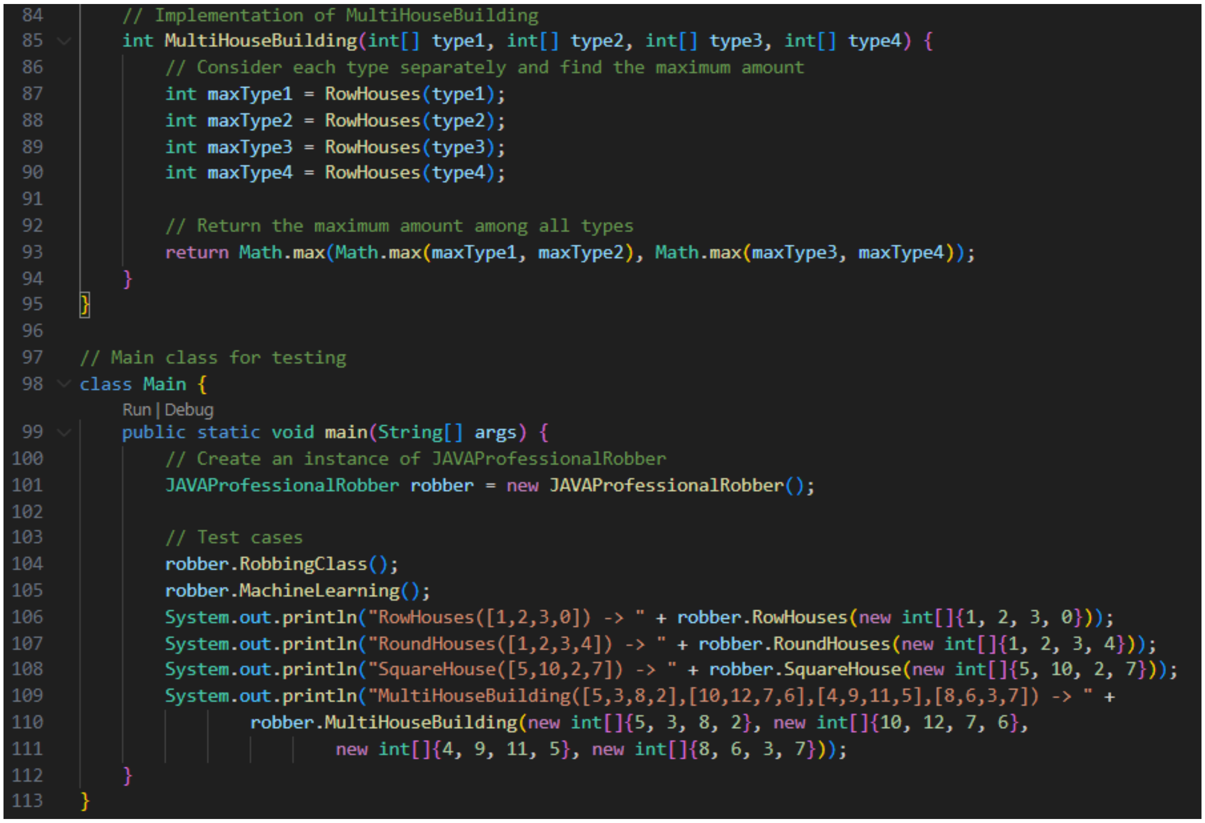
```java
import java.util.Arrays;

// Abstract class Robber
abstract class Robber {

    // Abstract method for robbing class
    abstract void RobbingClass();

    // Default method for MachineLearning
    void MachineLearning() {
        System.out.println(x:"I love MachineLearning.");
    }

    // Abstract method RowHouses
    abstract int RowHouses(int[] money);

    // Abstract method RoundHouses
    abstract int RoundHouses(int[] money);

    // Abstract method SquareHouse
    abstract int SquareHouse(int[] money);

    // Abstract method MultiHouseBuilding
    abstract int MultiHouseBuilding(int[] type1, int[] type2, int[] type3, int[] type4);
}

// Class JAVAProfessionalRobber inheriting from Robber
class JAVAProfessionalRobber extends Robber {

    // Implementation of RobbingClass
    void RobbingClass() {
        System.out.println(x:"MScAI&ML");
    }

```



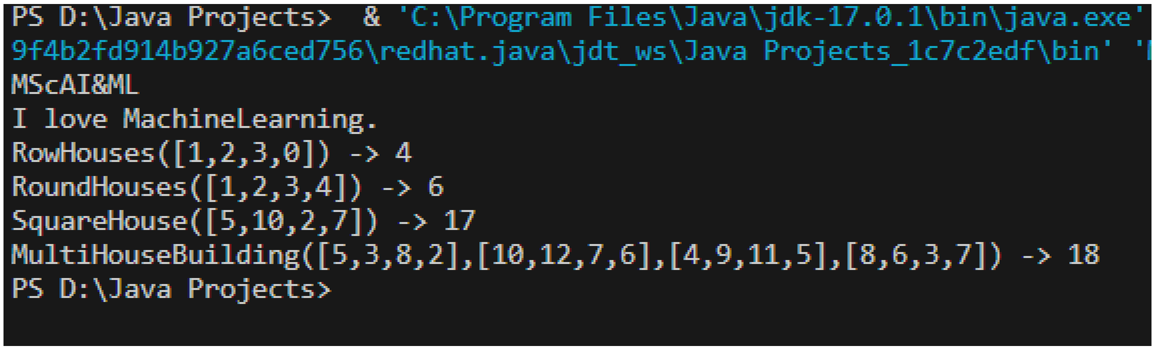
```java

    // Implementation of RowHouses
    int RowHouses(int[] money) {
        int n = money.length;
        if (n == 0) return 0;
        if (n == 1) return money[0];

        // Use two variables to keep track of previous and current maximum amounts
        int prevMax = money[0];
        int currentMax = Math.max(money[0], money[1]);

        for (int i = 2; i < n; i++) {
            int temp = currentMax;
            currentMax = Math.max(currentMax, prevMax + money[i]);
            prevMax = temp;
        }

        return currentMax;
    }
```


Java Lecture > src > J .java > JAVAProfessionalRobber > RoundHouses(int[])

```java
    // Implementation of RoundHouses
    int RoundHouses(int[] money) {
        int n = money.length;
        if (n == 0) return 0;
        if (n == 1) return money[0];

        // Consider two cases: Rob first house or not
        return Math.max(RowHouses(Arrays.copyOfRange(money, from:0, n - 1)),
                        RowHouses(Arrays.copyOfRange(money, from:1, n)));
    }

    // Implementation of SquareHouse
    int SquareHouse(int[] money) {
        int n = money.length;
        if (n == 0) return 0;
        if (n == 1) return money[0];

        // Use two variables to keep track of previous and current maximum amounts
        int prevMax = money[0];
        int currentMax = Math.max(money[0], money[1]);

        for (int i = 2; i < n; i++) {
            int temp = currentMax;
            currentMax = Math.max(currentMax, prevMax + money[i]);
            prevMax = temp;
        }

        return currentMax;
    }

```



```java
    // Implementation of MultiHouseBuilding
    int MultiHouseBuilding(int[] type1, int[] type2, int[] type3, int[] type4) {
        // Consider each type separately and find the maximum amount
        int maxType1 = RowHouses(type1);
        int maxType2 = RowHouses(type2);
        int maxType3 = RowHouses(type3);
        int maxType4 = RowHouses(type4);

        // Return the maximum amount among all types
        return Math.max(Math.max(maxType1, maxType2), Math.max(maxType3, maxType4));
    }
}

// Main class for testing
class Main {
    public static void main(String[] args) {
        // Create an instance of JAVAProfessionalRobber
        JAVAProfessionalRobber robber = new JAVAProfessionalRobber();

        // Test cases
        robber.RobbingClass();
        robber.MachineLearning();
        System.out.println("RowHouses([1,2,3,0]) -> " + robber.RowHouses(new int[]{1, 2, 3, 0}));
        System.out.println("RoundHouses([1,2,3,4]) -> " + robber.RoundHouses(new int[]{1, 2, 3, 4}));
        System.out.println("SquareHouse([5,10,2,7]) -> " + robber.SquareHouse(new int[]{5, 10, 2, 7}));
        System.out.println("MultiHouseBuilding([5,3,8,2],[10,12,7,6],[4,9,11,5],[8,6,3,7]) -> " +
                robber.MultiHouseBuilding(new int[]{5, 3, 8, 2}, new int[]{10, 12, 7, 6},
                        new int[]{4, 9, 11, 5}, new int[]{8, 6, 3, 7}));
    }
}
```



```
PS D:\Java Projects>  & 'C:\Program Files\Java\jdk-17.0.1\bin\java.exe'
9f4b2fd914b927a6ced756\redhat.java\jdt_ws\Java Projects_1c7c2edf\bin' '
MScAI&ML
I love MachineLearning.
RowHouses([1,2,3,0]) -> 4
RoundHouses([1,2,3,4]) -> 6
SquareHouse([5,10,2,7]) -> 17
MultiHouseBuilding([5,3,8,2],[10,12,7,6],[4,9,11,5],[8,6,3,7]) -> 18
PS D:\Java Projects>
```