

Code :

```
import java.util.Arrays;
import java.util.Comparator;
import java.util.Scanner;

// Class representing an item with weight and value
class Item {
    double weight, value;

    Item(double weight, double value) {
        this.weight = weight;
        this.value = value;
    }
}

public class FractionalKnapsack {

    // Function to solve Fractional Knapsack problem
    static double getMaxValue(Item[] items, double capacity) {
        // Sort items by descending value-to-weight ratio
        Arrays.sort(items, new Comparator<Item>() {
            public int compare(Item a, Item b) {
                double r1 = a.value / a.weight;
                double r2 = b.value / b.weight;
                if (r1 < r2) return 1;
                else if (r1 > r2) return -1;
                else return 0;
            }
        });
    }
}
```

```
};

double totalValue = 0.0; // Total value accumulated
double remaining = capacity; // Remaining capacity

for (Item item : items) {
    if (item.weight <= remaining) {
        // Take the whole item
        remaining -= item.weight;
        totalValue += item.value;
    } else {
        // Take fractional part
        totalValue += item.value * (remaining / item.weight);
        break; // Knapsack is full
    }
}

return totalValue;
}

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

    System.out.print("Enter number of items: ");
    int n = sc.nextInt();

    Item[] items = new Item[n];

    for (int i = 0; i < n; i++) {
```

```

        System.out.print("Enter weight of item " + (i + 1) + ": ");
        double w = sc.nextDouble();
        System.out.print("Enter value of item " + (i + 1) + ": ");
        double v = sc.nextDouble();
        items[i] = new Item(w, v);
    }

    System.out.print("Enter capacity of knapsack: ");
    double capacity = sc.nextDouble();
    sc.close();

    double maxValue = getMaxValue(items, capacity);
    System.out.println("\nMaximum value in knapsack = " + maxValue);
}

}

```

Output :

Enter number of items: 3

Enter weight of item 1: 10

Enter value of item 1: 60

Enter weight of item 2: 20

Enter value of item 2: 80

Enter weight of item 3: 30

Enter value of item 3: 100

Enter capacity of knapsack: 40

Maximum value in knapsack = 173.3333333333331