111. Knapsack problem using greedy

PROGRAM:-

```
class Item:
    def __init__(self, value, weight):
        self.value = value
        self.weight = weight

    def __lt__(self, other):
        return (self.value / self.weight) > (other.value / other.weight)

def fractional_knapsack(capacity, items):
    # Sort items by value-to-weight ratio in descending order
    items.sort()

    total_value = 0.0  # Total value in the knapsack
    for item in items:
        if capacity > 0 and item.weight <= capacity:
            # Take the whole item
            capacity -= item.weight
            total_value += item.value
        else:
            # Take the fraction of the remaining item
            fraction = capacity / item.weight
            total_value += item.value * fraction
            break

    return total_value

# Example usage
if __name__ == "__main__":
    items = [
        Item(60, 10),
        Item(100, 20),
        Item(120, 30)
    ]
    capacity = 50

    max_value = fractional_knapsack(capacity, items)
    print(f"Maximum value in the knapsack: {max_value}")
```

OUTPUT:-

```
Maximum value in the knapsack: 240.0

=== Code Execution Successful ===
```

TIME COMPLEXITY:-O(n log N)