

**Name:** Phadtare Shivendra Uday

**Class:** BE Div: B

**Roll No:** COB434

**Batch:** B2

---

### **Assignment No:3**

**Write a program to solve Fractional Knapsack Problem using Greedy method.**

**CODE:**

```
#include <bits/stdc++.h>
using namespace std;
struct Item {
    int profit; int weight;

    // Constructor Item(int p, int w) {
    profit = p; weight = w;
    }
};
// Comparator function to sort items according to profit/weight ratio
bool cmp(Item a, Item b) {
    double r1 = (double)a.profit / a.weight;
    double r2 = (double)b.profit / b.weight;
    return r1 > r2;
}

// Function to solve fractional knapsack problem
double fractionalKnapsack(int W, Item arr[], int N) {
    // Sort items by descending ratio of profit/weight
    sort(arr, arr + N, cmp);
    double finalValue = 0.0;
    for (int i = 0; i < N; i++) {
        // If adding item weight won't exceed capacity
        if (arr[i].weight <= W) {
            W -= arr[i].weight;
            finalValue += arr[i].profit;
        } else {
            // Take fraction of the last item that fits
            finalValue += arr[i].profit * ((double)W / arr[i].weight); break;
        }
    }

    return finalValue;
}

int main() {

    int W = 20;
```

```
// Capacity of knapsack

// Array of items {profit, weight}
Item arr[] = { {20, 30}, {10, 30}, {10, 10} };
int N = sizeof(arr) / sizeof(arr[0]);

cout << "Maximum profit in Knapsack = " << fractionalKnapsack(W, arr, N) << endl;
return 0;
}
```

OUTPUT:

```
(base) sspm@sspm:~$ g++ daa1.cpp (base)
sspm@sspm:~$ ./a.out
Maximum profit in Knapsack = 16.6667
```