Assignment – 3

Name – Abhishek Santosh Gaikwad

PRN – 22210975

Roll No. 371017

CODE:

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// Structure to represent an item with value, weight, and value-to-weight ratio

struct Item {

    int value, weight;

    // Constructor to initialize the value and weight of an item

    Item(int v, int w) {

        value = v;

        weight = w;

    }

};

// Comparison function to sort items based on value-to-weight ratio in descending order

bool compare(Item a, Item b) {

    double r1 = (double)a.value / a.weight;

    double r2 = (double)b.value / b.weight;

    return r1 > r2;  // Sort by decreasing ratio

}

// Function to calculate the maximum value in the knapsack using the greedy method

double fractionalKnapsack(int capacity, vector<Item> items) {

    // Sort items based on value-to-weight ratio

    sort(items.begin(), items.end(), compare);

    double totalValue = 0.0;  // To store the total value of the knapsack

    for (auto item : items) {

        if (capacity >= item.weight) {

            // If the item can be fully taken, add its full value

            totalValue += item.value;

            capacity -= item.weight;

        } else {

            // If the item can't be fully taken, take the fraction that fits

            totalValue += item.value \* ((double)capacity / item.weight);

            break;  // Knapsack is full

        }

    }

    return totalValue;

}

int main() {

    // Create a list of items with value and weight

    vector<Item> items = {

        {60, 10},   // Item 1

        {100, 20},  // Item 2

        {120, 30},  // Item 3

        {200, 40},  // Item 4

        {150, 50}   // Item 5

    };

    // Knapsack capacity

    int capacity = 20;

    // Get the maximum value that can be carried in the knapsack

    double maxValue = fractionalKnapsack(capacity, items);

    // Print the result

    cout << "The maximum value in the knapsack of capacity " << capacity << " is " << maxValue << endl;

    return 0;

}

OUTPUT:

