NAME: Aniruddh Sawant

ROLL NO: 52

SUBJECT: AOA

EXPERIMENT NO: 5

To implement fractional knapsack using greedy

method.

/\* A Naive recursive implementation

of 0-1 Knapsack problem \*/

#include <stdio.h>

// A utility function that returns

// maximum of two integers

int max(int a, int b) { return (a > b) ? a : b; }

// Returns the maximum value that can be

// put in a knapsack of capacity W

int knapSack(int W, int wt[], int val[], int n)

{

// Base Case

if (n == 0 || W == 0)

return 0;

// If weight of the nth item is more than

// Knapsack capacity W, then this item cannot

// be included in the optimal solution

if (wt[n - 1] > W)

return knapSack(W, wt, val, n - 1);

// Return the maximum of two cases:

// (1) nth item included

// (2) not included

else

return max(

val[n - 1]

+ knapSack(W - wt[n - 1], wt, val, n - 1),

knapSack(W, wt, val, n - 1));

}

// Driver code

int main()

{

int profit[] = { 60, 100, 120 };

int weight[] = { 10, 20, 30 };

int W = 50;

int n = sizeof(profit) / sizeof(profit[0]);

printf("%d", knapSack(W, weight, profit, n));

return 0;

}

OUTPUT:

