**Practical – 8**

**Aim :-** Implement Program for fractional knapsack using Greedy design technique.

**Code:**

# include<stdio.h> void knapsack(int n, float weight[], float profit[], float capacity) { float x[20], tp = 0;

int i, j, u;

u = capacity;

for (i = 0; i < n; i++) x[i] = 0.0; for (i = 0; i < n; i++) { if (weight[i] >

u) break; else { x[i] = 1.0; tp = tp + profit[i]; u = u - weight[i];

} } if (i < n) x[i] = u / weight[i]; tp = tp + (x[i] \* profit[i]); printf("\nThe result vector is:- "); for (i = 0; i < n; i++) printf("%f\t", x[i]); printf("\nMaximum profit is:- %f", tp);

} int main() { float weight[20], profit[20], capacity; int num, i, j; float ratio[20], temp; printf("\nEnter the no. of objects:- ");

scanf("%d", &num);

printf("\nEnter the wts and profits of each object:- "); for (i = 0; i < num; i++) { scanf("%f %f", &weight[i], &profit[i]);

} printf("\nEnter the capacityacity of knapsack:- "); scanf("%f", &capacity); for (i = 0; i < num; i++) { ratio[i] = profit[i] / weight[i];

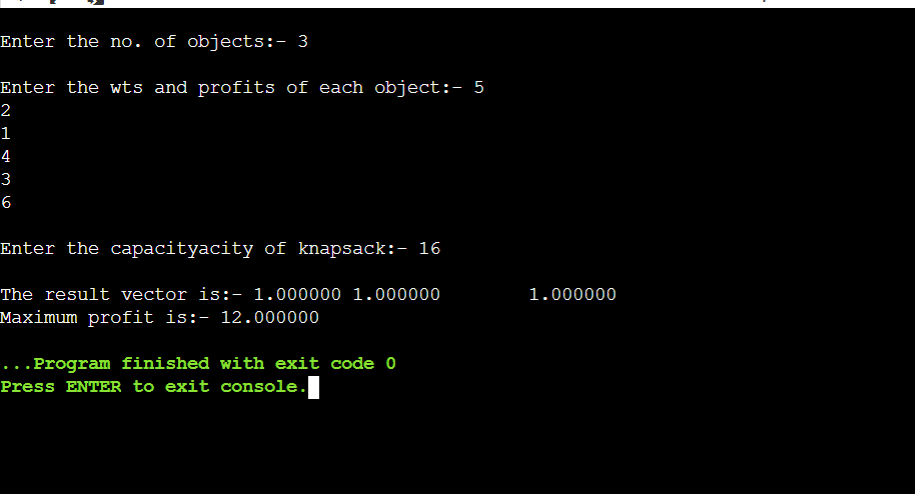
} for (i = 0; i < num; i++) { for (j = i + 1; j < num; j++) { if (ratio[i] < ratio[j]) { temp = ratio[j]; ratio[j] = ratio[i]; ratio[i] = temp;

temp = weight[j]; weight[j] = weight[i]; weight[i] = temp; temp = profit[j]; profit[j] = profit[i]; profit[i] = temp;

}

} } knapsack(num, weight, profit, capacity); return(0); }

**Output:**

****