**Experiment 8**

**WAP to perform the empirical analysis of greedy algorithm to solve the fractional Knapsack problem.**

**Program:-**

//fractional knapsack

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <time.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("\nMaximum profit is:- %f", tp);

}

int main() {

float weight[1000], profit[1000], capacity;

int n, i,j, randNum1, randNum2;

float ratio[1000], temp;

double time;

clock\_t start, end;

printf("\nEnter the no. of objects: ");

scanf("%d", &n);

printf("\nEnter the capacity of knapsack:-");

scanf("%f", &capacity);

start = clock();

for (i = 0; i < n; i++) {

randNum1 = (rand() % 1000);

weight[i] = randNum1;

randNum2 = (rand() % 1000);

profit[i] = randNum2;

printf("weight :%f profit %f\n", weight[i], profit[i]);

}

for (i = 0; i < n; i++) {

ratio[i] = profit[i] / weight[i];

}

for (i = 0; i < n; i++) {

for (int j = i + 1; j < n; 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(n, weight, profit, capacity);

end = clock();

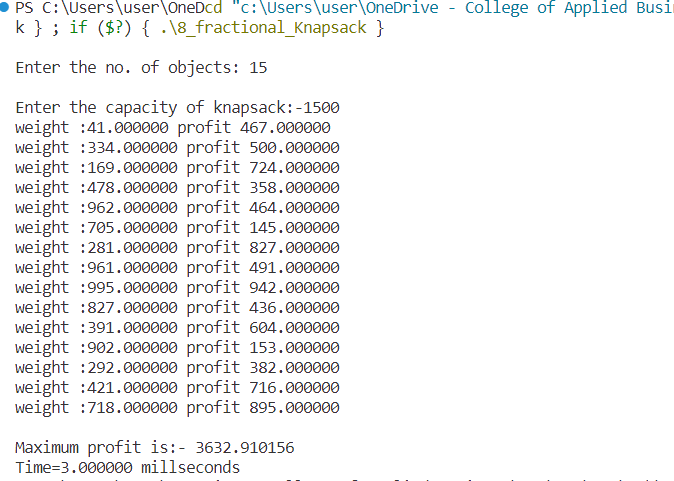
time = ((double)(end - start) \* 1000) / CLOCKS\_PER\_SEC;

printf("\nTime=%lf millseconds", time);

return 0;

}

**Output:**



**Conclusion:**

This experiment had been conducted in a 64-bit system with 16 GB RAM and Processor 12th Gen Intel(R) Core (TM) i5-12500H 3.10 GHz. The algorithm was implemented in C programming language in Visual Studio Code 1.85.1 Code Editor. The time taken by this algorithm for 15 number of input size is 3 milliseconds. The running time is analyzed as O(nlogn).