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:

```
PS C:\Users\user\OneDcd "c:\Users\user\OneDrive - College of Applied Busi
k } ; if ($?) { .\8_fractional_Knapsack }

Enter the no. of objects: 15

Enter the capacity of knapsack:-1500
weight :41.000000 profit 467.000000
weight :334.000000 profit 500.000000
weight :169.000000 profit 724.000000
weight :478.000000 profit 358.000000
weight :962.000000 profit 464.000000
weight :705.000000 profit 145.000000
weight :281.000000 profit 827.000000
weight :961.000000 profit 491.000000
weight :995.000000 profit 942.000000
weight :827.000000 profit 436.000000
weight :391.000000 profit 604.000000
weight :902.000000 profit 153.000000
weight :292.000000 profit 382.000000
weight :421.000000 profit 716.000000
weight :718.000000 profit 895.000000

Maximum profit is:- 3632.910156
Time=3.000000 millseconds
```

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).