|  |  |
| --- | --- |
| **NAME:** | Vaishnavi Bhagawan Borkar |
| **UID:** | 2021300016 |
| **SUBJECT** | Data Analysis and Algorithm |
| **EXPERIMENT NO:** | Experiment 5 |
| **DATE OF PERFORMANCE:** | 12/03/23 |
| **AIM:** | Experiment based on greedy approach. |
| **THEORY:** | The knapsack problem states that − given a set of items, holding weights and profit values, one must determine the subset of the items to be added in a knapsack such that, the total weight of the items must not exceed the limit of the knapsack and its total profit value is maximum.  It is one of the most popular problems that take greedy approach to be solved. It is called as the **Fractional Knapsack Problem**. |
| **ALGORITHM:** | The weights (Wi) and profit values (Pi) of the items to be added in the knapsack are taken as an input for the fractional knapsack algorithm and the subset of the items added in the knapsack without exceeding the limit and with maximum profit is achieved as the output. **Algorithm**  * Consider all the items with their weights and profits mentioned respectively. * Calculate Pi/Wi of all the items and sort the items in descending order based on their Pi/Wi values. * Without exceeding the limit, add the items into the knapsack. * If the knapsack can still store some weight, but the weights of other items exceed the limit, the fractional part of the next time can be added. * Hence, giving it the name fractional knapsack problem. |
| **PROGRAM:** | *#include* <stdio.h>  void main()  {      int capacity, no\_items, cur\_weight, item;      int used[10];      float total\_profit;      int i;      int weight[10];      int value[10];      printf("Enter the capacity of knapsack:\n");      scanf("%d", &capacity);      printf("Enter the number of items:\n");      scanf("%d", &no\_items);      printf("Enter the weight and value of %d item:\n", no\_items);  *for* (i = 0; i < no\_items; i++)      {          printf("Weight[%d]:\t", i);          scanf("%d", &weight[i]);          printf("Value[%d]:\t", i);          scanf("%d", &value[i]);      }  *for* (i = 0; i < no\_items; ++i)          used[i] = 0;      cur\_weight = capacity;  *while* (cur\_weight > 0)      {          item = -1;  *for* (i = 0; i < no\_items; ++i)  *if* ((used[i] == 0) &&                  ((item == -1) || ((float)value[i] / weight[i] > (float)value[item] / weight[item])))                  item = i;          used[item] = 1;          cur\_weight -= weight[item];          total\_profit += value[item];  *if* (cur\_weight >= 0)              printf("Added object %d (%d Rs., %dKg) completely in the bag. Space left: %d.\n", item + 1, value[item], weight[item], cur\_weight);  *else*          {              int item\_percent = (int)((1 + (float)cur\_weight / weight[item]) \* 100);              printf("Added %d%% (%d Rs., %dKg) of object %d in the bag.\n", item\_percent, value[item], weight[item], item + 1);              total\_profit -= value[item];              total\_profit += (1 + (float)cur\_weight / weight[item]) \* value[item];          }      }      printf("Filled the bag with objects worth %.2f Rs.\n", total\_profit);  } |
| **RESULT:** | |
| **CONCLUSION:** | By performing the experiment, I understood the concepts of knapsack and fractional knapsack problem. |