DAA ASSIGNMENT

K.PRANAVI

21071A6732

Fractional Knapsack

The basic idea of the greedy approach is to calculate the ratio value/weight for each item and sort the item on the basis of this ratio. Then take the item with the highest ratio and add them until we can't add the next item as a whole and at the end add the next item as much as we can. Which will always be the optimal solution to this problem.

Follow the given steps to solve the problem using the above approach:

- Calculate the ratio(value/weight) for each item.
- Sort all the items in decreasing order of the ratio.
- Initialize res =0, curr_cap = given_cap.
- Do the following for every item "i" in the sorted order:
- If the weight of the current item is less than or equal to the remaining capacity then add the value of that item into the result
- Else add the current item as much as we can and break out of the loop.
- Return res.

ALGORITHM:

```
Algorithm Fr-Knapsack (M,n)
{
for (i:=1; i <= n ;i++)
{
S[i] :=0.0;
}
balance:= M;
for(i:=1; i <= n ;i++)
{
if(W [i] > balance ) /
break;
S[i] :=1.0; .
balance := balance - W[i];
}
f(i <= n)
S[i]:= (balance / W[i] ) ;
return S;
    }
```

PROGRAM:

```
class Item:
  def _init_(self, value, weight):
    self.value = value
    self.weight = weight
  arr.sort(key=lambda x: (x.value/x.weight), reverse=True)
  finalvalue = 0.0
  for item in arr:
    if item.weight <= W:
      W -= item.weight
      finalvalue += item.value
    else:
      finalvalue += item.value * W / item.weight
      break
  return finalvalue
```

```
if _name_ == "_main_":
    W = 50
    arr = [Item(60, 10), Item(100, 20), Item(120, 30)]
    max_val = fractionalKnapsack(W, arr)
    print(max_val)
```

OUTPUT:

Maximum value we can obtain = 240

class Item: Untitled-1

```python
class Item:
    def _init_(self, value, weight):
        self.value = value
        self.weight = weight
    arr.sort(key=lambda x: (x.value/x.weight), reverse=True)
    finalvalue = 0.0
    for item in arr:
        if item.weight <= W:
            W -= item.weight
            finalvalue += item.value
        else:
            finalvalue += item.value * W / item.weight
            break
    return finalvalue
if _name_ == "_main_":
    W = 50
    arr = [Item(60, 10), Item(100, 20), Item(120, 30)]
    max_val = fractionalKnapsack(W, arr)
    print(max_val)
```