|  |  |
| --- | --- |
|  | DEPARTMENT OF ARTIFICIAL INTELLIGNECE & DATA SCIENCE |

|  |  |
| --- | --- |
| Subject: Analysis of Algorithm | Course Code: CSC402 |
| Semester: 4 | Course: AI & DS |
| Laboratory No: 205 | Name of Subject Teacher: Pramod Bhavarthe |
| Name of Student: Sahil Shaikh | Roll Id: VU2S2223012 |

|  |  |
| --- | --- |
| Title of Practical | knapsack Algorithm. |

**Theory –**

In **Fractional Knapsack**, we can break items for maximizing the total value of the knapsack.

**Naive Approach:** To solve the problem follow the below idea:

Try all possible subsets with all different fractions.

**Time Complexity:** O(2N)

An efficient solution is to use the Greedy approach.

The basic idea of the greedy approach is to calculate the ratio **profit/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 as much as we can (can be the whole element or a fraction of it).

This will always give the maximum profit because, in each step it adds an element such that this is the maximum possible profit for that much weight.

**Program –**

class Item:

    def \_\_init\_\_(self, profit, weight):

        self.profit = profit

        self.weight = weight

# Main greedy function to solve problem

def fractionalKnapsack(W, arr):

    # Sorting Item on basis of ratio

    arr.sort(key=lambda x: (x.profit/x.weight), reverse=True)

    # Result(value in Knapsack)

    finalvalue = 0.0

    # Looping through all Items

    for item in arr:

        # If adding Item won't overflow,

        # add it completely

        if item.weight <= W:

            W -= item.weight

            finalvalue += item.profit

        # If we can't add current Item,

        # add fractional part of it

        else:

            finalvalue += item.profit \* W / item.weight

            break

    # Returning final value

    return finalvalue

# Driver Code

if \_\_name\_\_ == "\_\_main\_\_":

    W = 50

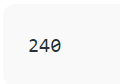
    arr = [Item(60, 10), Item(100, 20), Item(120, 30)]

    # Function call

    max\_val = fractionalKnapsack(W, arr)

    print(max\_val)

**Output –**

****

**Conclusion –**

**Therefore, we have successfully understood and Implemented Knapsack Algorithm.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Grade and Dated Signature of Teacher** | **Total (10)** | **Remark** | **Dated signature of teacher** |
|  |  |  |