(S.S.G.M.C.E.SHEGAON)

Expt. No. 05 Date : 13-04-22

Experiment - 5

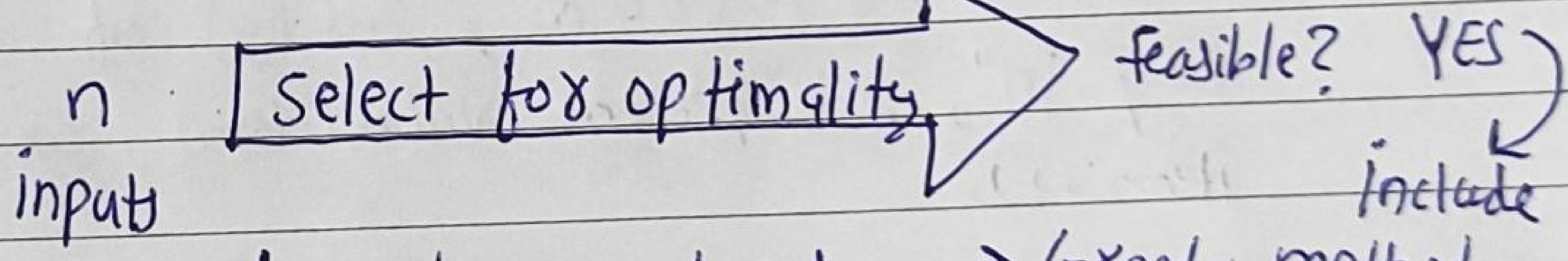
* Aim :→ To Study and implement Knapsack" Algorithm and understand about Greedy Algorithm.

* Facilities :→ Software:- TurboC++
Hardware:- Computer Machine

* Objective:- At the end of experiment, we will be able to run & implement Knapsack Algorithm & also we will be able to ~~ge~~ Understand Greedy Algorithm

* Theory :→

In Greedy method, there are n inputs for some values and an objective function. The method gives an optimal solution to the problem by checking if the solution is feasible Soluthion. All of the n inputs may not be included, only those needed to be form the optimal solution will be included. Each input may consume some resource, which is available in limited quantity.



depends upon Local Maximam $\longrightarrow$ Greedy method.

* Aim :→ To study and implement Knapsack's Algorithm and Understand about Greedy Algorithm.

* Facilities:- Software → Turbo C++
  Hardware → Computer Machine

* Program:→

```
        # include <conio.h>
        #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 item: \n");
scanf(" %d", &no_items);
printf(" Enter the weight and value of %d item : \n", no_items);

for (int i = 0 ; i< no_items; i++){
        printf("weight [%d] : \t", i);
        scanf( "%d", & weight[i]);
        Printf(" Value [%d] : \t", i);
        Scanf ("%d", & value[i]);
   }
```

The word greedy refers to allocating the maximum possible value of some limited resource to the first element which enters the optimal solution.

The feasibility of the solution is expressed in terms of obeying the constraints of the resource. Thus greedy method depends upon local (short range) maximum.

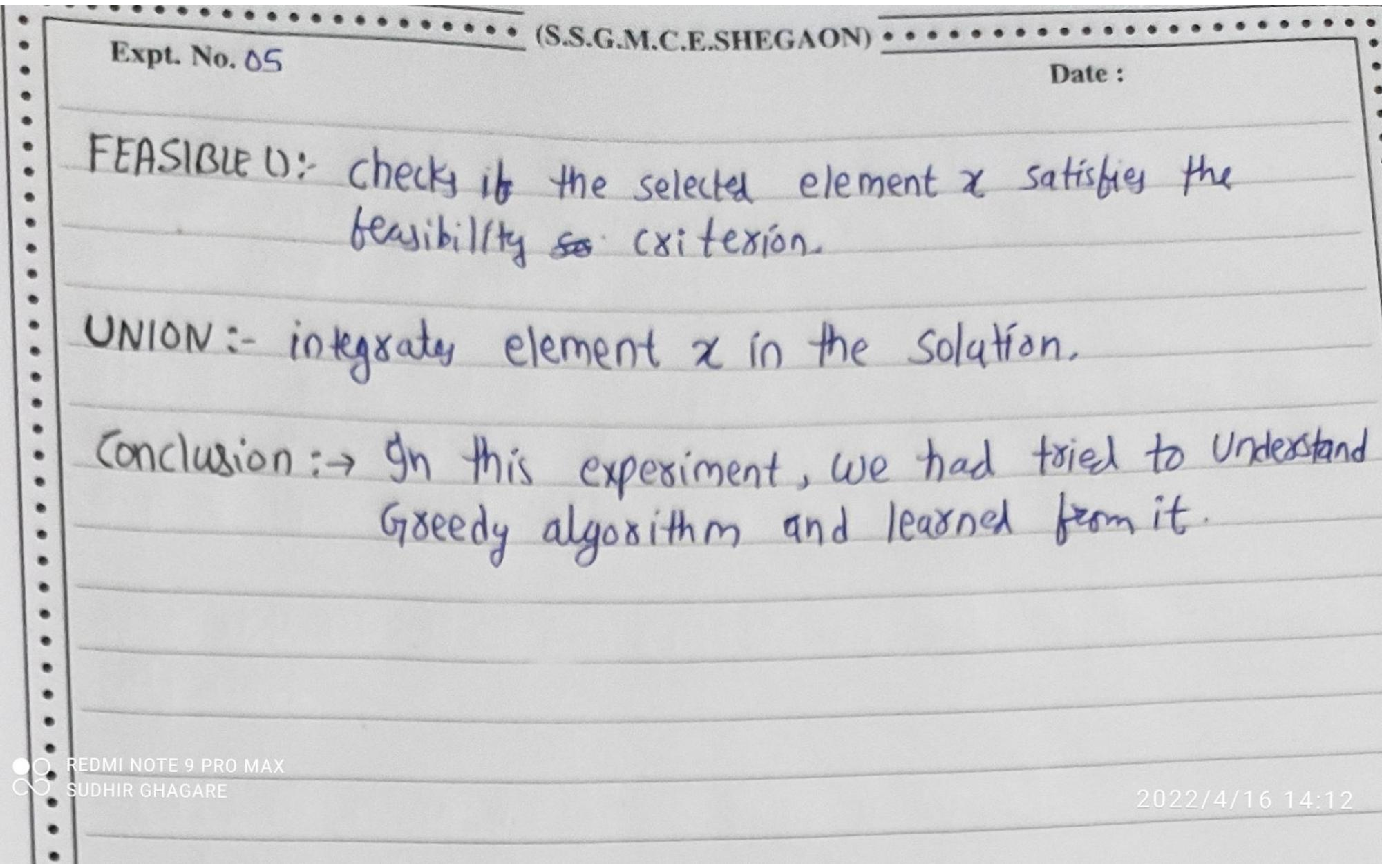
The abstract algorithm can be expressed as the control algorithm:

```
[a[1:n] is an array containing n inputs]
Solution ← empty;
for i ← 1 to n do
    x ← SELECT(a); [as per some optimization criteria]
    if FEASIBLE (solution, x) then
        Solution ← UNION (solution, x);
    end
end
return solution;
```

The three function SELECT(), FEASIBLE() and UNION() do the detailed work of this abstract algorithm.

SELECT():- Select an element from a[] such that it has a potential for satisfying the optimality criterion or selection policy.

```
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 [i]/ (float) weight [i])
        item = i;
        used [item] = 1;

    curr_weight  -= weight [item];
    total_profit  += value [item];

    if (cur_weight >0) {
      printf("Added object %d (%d Rs, %d Kg) Completly in a
        bay. 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., %d Kg) of object %d in the
          bay. \n", item_percent, value[item], weight [item],
          item +1);
  total_profit -= value [item]
  total_profit += (1+ (float) cur_weight / weight [item])* Value [item];
  }
```



(S.S.G.M.C.E.SHEGAON)

Expt. No. 05

Date :

FEASIBLE ():- checks if the selected element x satisfies the feasibility ~~so~~ criterion.

UNION :- integrates element x in the solution.

Conclusion :→ In this experiment, we had tried to Understand Greedy algorithm and learned from it.

```
}
printf("Filled the bag with objects worth %.2f Rs \n",
        total_profit);
}
```

* Conclusion:- In this experiment, we had tried to understand Greedy algorithm and learned from it.