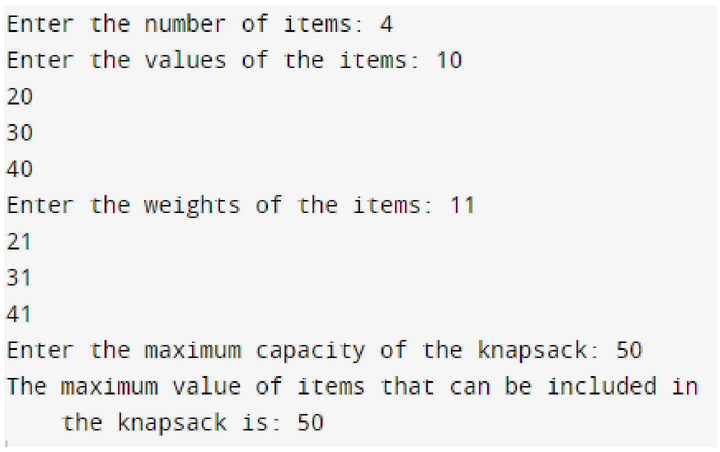
Practical – 9

Implementation of a knapsack problem using dynamic programming.

```
#include <stdio.h>
int max(int a, int b) {
return a > b ? a : b;
}
int knapsack(int W, int wt[], int val[], int n) {
int K[n + 1][W + 1];
for (int i = 0; i <= n; i++)
for (int w = 0; w <= W; w++)
K[i][w] = (i == 0 || w == 0) ? 0 :
(wt[i - 1] <= w) ? max(val[i - 1] + K[i - 1][w- wt[i - 1]], K[i - 1][w]) :
K[i - 1][w];
return K[n][W];
}
int main() {
printf("21012021003_AMIT GOSWAMI \n");
int n, W;
printf("Enter the number of items: ");
scanf("%d", &n);
int val[n], wt[n];
printf("Enter the values of the items: ");
for (int i = 0; i < n; i++)
scanf("%d", &val[i]);
printf("Enter the weights of the items: ");
for (int i = 0; i < n; i++)
scanf("%d", &wt[i]);
printf("Enter the maximum capacity of the knapsack: ");
scanf("%d", &W);
int result = knapsack(W, wt, val, n);
printf("The maximum value of items that can be included in the knapsack is: %d\n", result);
return 0;
}
```



```
Enter the number of items: 4
Enter the values of the items: 10
20
30
40
Enter the weights of the items: 11
21
31
41
Enter the maximum capacity of the knapsack: 50
The maximum value of items that can be included in
    the knapsack is: 50
```