Subject :

Software :

Hardware :

Branch :

Semester :

Page No.

Prog No.

PROBLEM STATEMENT : Write a Program to solve a Knapsack Problem Using Dynamic Programming.

ALGORITHM & CODE :

```
#include <stdio.h>
int main()
{ int n, W;
  printf("Enter the number of items: ");
  scanf("%d", &n);
  printf("Enter the capacity of the knapsack: ");
  scanf("%d", &W);
  int value[n], weight[n];
  printf("Enter values and weights of items: \n");
for (int i=0; i<n; i++)
{  printf("Item %d - Value :", i+1);
   scanf("%d", &value[i]);
   printf("Item %d - Weight :", i+1);
   scanf("%d", &weight[i]);
}
int dp[n+1][W+1];
For (int i=0; i<=n; i++)
{  for (int w=0; w<=W; w++)
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS

GRADE :

Signature of Faculty

Date :

Signature of Student

Date :

| Subject : | Software : |
|---|---|
| | Hardware : |
| Branch : / Semester : | Page No. / Prog No. |

PROBLEM STATEMENT

ALGORITHM & CODE :

```
{ if( i==0 || w==0)
      dp[i][w] =0;
  elseif (weight[i-1]<=w)
     dp[i][w]= (value[i-1] + dp[i-1][w-weight
                  [i-1]] > dp[i-1][w])?
            (value[i-1]+dp[i-1][w-weight[i-1]])
             : dp[i-1][w];
  else
      dp[i][w] = dp[i-1][w];
  }
}
printf(" Maximum Value in Knapsack = %d\n",
        dp[n][W]);
  return 0;
}
```

Output Obtained

```
Enter the number of items: 3
Enter the capacity of the knapsack : 5
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS

| GRADE : | Signature of Faculty<br>Date : | Signature of Student<br>Date : |
|---|---|---|

| | | | | |
|---|---|---|---|---|
| | Subject : | | Software : | |
| | | | Hardware : | |
| | Branch : | Semester : | Page No. | Prog No. |

PROBLEM STATEMENT

ALGORITHM & CODE :

Enter values and weights of items:

Item 1 - Value : 60

Item 1 - Weight : 2

Item 2 - Value : 100

Item 2 - Weight: 3

Item 3 - value : 120

Item 3 - Weight: 4

Maximum Value in Knapsack = 160

PUT GIVEN

TPUT OBTAINED

EMARKS

| | | |
|---|---|---|
| ADE : | Signature of Faculty<br>Date : | Signature of Student<br>Date : |