1BM19CS087
Mithun.R.K
4-B

Question:- Implement 0/1 knapsack problem using dynamic programming.

Modification: Give the count of items selected.

Code:

```
#include<stdio.h>
int max(int,int);
int m,i,j,n,p[10],w[10],v[10][10],x[10],op-soln,flag=0;
int knapsack();
void object-selected();

void main(){
  printf("Enter number of objects\n");
  scanf("%d",&n);
  printf("Enter weights of n objects\n");
  for(i=1;i<=n;i++)
      scanf("%d",&w[i]);
  printf("Enter profits of n objects\n");
  for(i=1;i<=n;i++)
      scanf("%d",&p[i]);
  printf("Enter the capacity of knapsack\n");
      scanf("%d",&m);
  op-soln = knapsack(n,w,m,v,p);
  printf("Output is\n");
  for(i=0;i<=n;i++){
    for(j=0;j<=m;j++){
        printf("%d\t",v[i][j]);
    }
    printf("\n");
  }
  printf("optimal solution = %d\n",op-soln);
  object-selected();
}
int max(int a,int b){
   return(a>b?a:b);
}
```

```
int knapsack() {
  int i, j;
  for (i=0; i<=n; i++) {
    for (j=0; j<=m; j++) {
      if (i==0 || j==0)
        v[i][j]=0;
      else {
        if (w[i]>j)
          v[i][j]=v[i-1][j];
        else
          v[i][j]=max(v[i-1][j], v[i-1][j-w[i]]+p[i]);
      }
    }
  }
  return v[n][m];
}
void object_selected() {
  i=n;
  j=m;
  while (i!=0 && j!=0) {
    if (v[i][j] != v[i-1][j]) {
      x[i]=1;
      j=j-w[i];
    }
    i--;
  }
  printf("objects selected\n");
  for (i=1; i<=n; i++) {
    if (x[i]==1) {
      flag++;
      printf("%d\t", i);
      printf("count of items stocked : %d", flag);
    }
  }
  printf("count of items selected : %d", flag);
}
```