5. 0/1 Knapsack problem

```c
#include <stdio.h>

int items, capacity, weight[10], profit[10], dp[10][10];

int getMax(int a, int b) {
   return (a > b) ? a : b;
}

void knapsackProblem(int items, int weight[10], int profit[10], int capacity) {
   for (int i = 0; i <= items; i++) {
      for (int j = 0; j <= capacity; j++) {
         if (i == 0 || j == 0)
            dp[i][j] = 0;
         else if (weight[i] > j)
            dp[i][j] = dp[i - 1][j];
         else
            dp[i][j] = getMax(dp[i - 1][j], dp[i - 1][j - weight[i]] + profit[i]);
      }
   }

   printf("\nMaximum Profit: %d\n", dp[items][capacity]);

   printf("\nDynamic Programming Table:\n");
   for (int i = 0; i <= items; i++) {
      for (int j = 0; j <= capacity; j++) {
         printf("\t%d", dp[i][j]);
      }
      printf("\n");
   }
}

int main() {
   printf("Enter the number of items: ");
   scanf("%d", &items);

   printf("Enter the weights of the items: ");
   for (int i = 1; i <= items; i++) {
      scanf("%d", &weight[i]);
   }

   printf("Enter the profits of the items: ");
```

```
    for (int i = 1; i <= items; i++) {
        scanf("%d", &profit[i]);
    }

    printf("Enter the knapsack capacity: ");
    scanf("%d", &capacity);

    knapsackProblem(items, weight, profit, capacity);
    return 0;
}
```

```
Output
Maximum Profit: 7

Dynamic Programming Table:
    0 0 0 0 0 0
    0 0 0 0 3 3
    0 0 0 4 4 4
    0 0 0 4 5 5
    0 0 0 4 6 7
```