DESIGN AND ANALYSIS OF ALGORITHMS

PRACTICAL 4 (B) : 0/1 KNAPSACK ALGORITHM

-Khushi Sidana
BTech CSBS A034

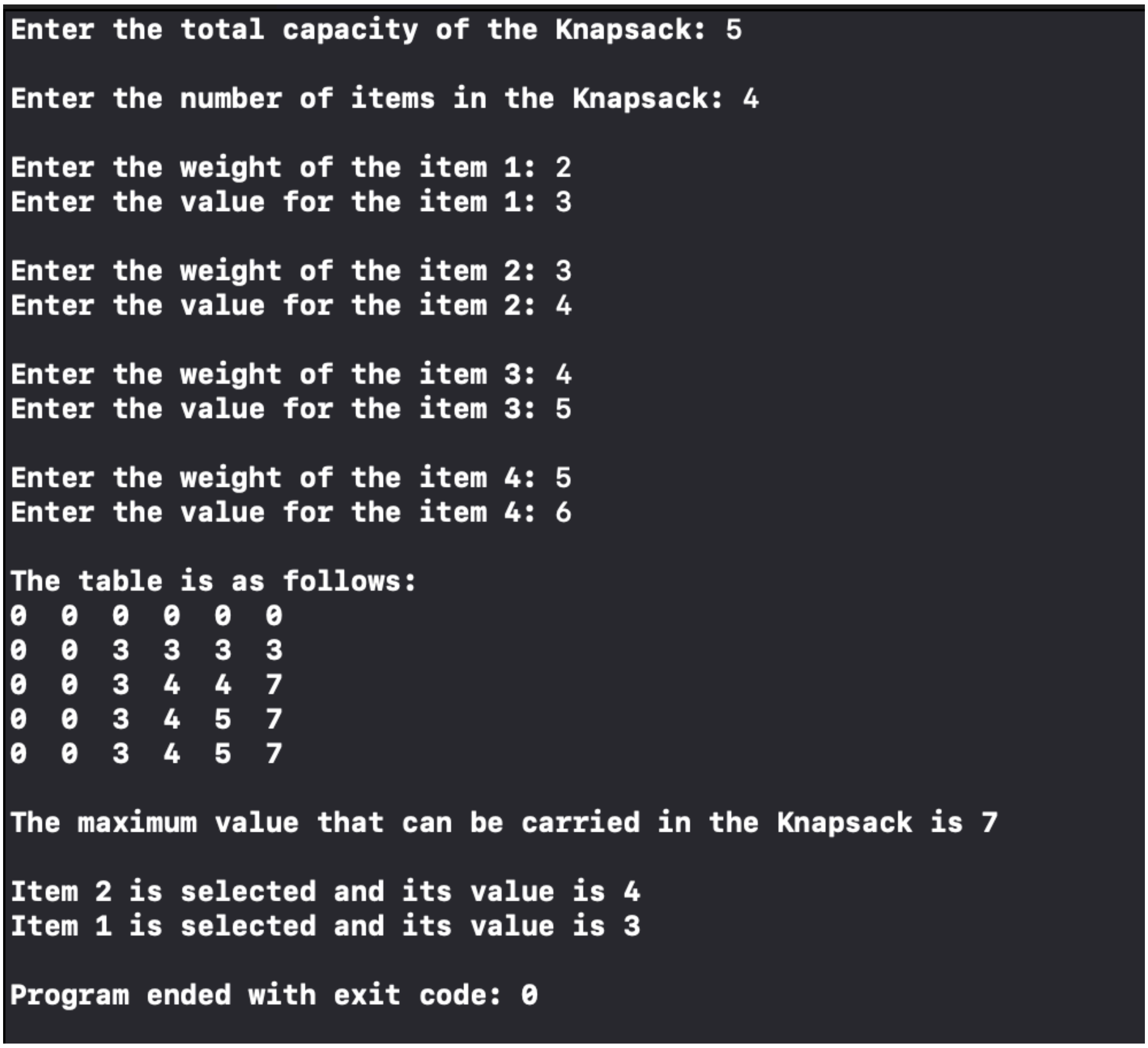
Aim: To implement 0/1 Knapsack.
Language used: C++

CODE:

```
#include <iostream>
using namespace std;

//Function to get maximum of two integers
int max(int x, int y)
{
   return (x > y) ? x : y;
}

//Function for Knapsack implementation
void knapSack(int W, int w[], int v[], int n)
{
   int i, weight, max_val, wt ;
   int Knap[n+1][W+1];
   for(i=0;i<=n;i++)
   {
      for (weight=0; weight<=W; weight++)
      {
         if (i==0 || weight == 0)
            Knap[i][weight]=0;
         else if (w[i-1]<=weight)
            Knap[i][weight] = max(v[i-1]+Knap[i-1][weight-w[i-1]],Knap[i-1][weight]);
         else
            Knap [i][weight] = Knap[i-1][weight];
      }
   }

   //Displaying the table
   cout<<endl<<"The table is as follows: "<<endl;
   for(i=0;i<=n;i++)
   {
      for (weight=0; weight<=W; weight++)
      {
         cout<<Knap[i][weight]<<"  ";
      }
      cout<<endl;
   }
```

```
    //Finding the maximum value that can be carried in the Knapsack
    max_val= Knap[n][W];
    cout<<endl<<"The maximum value that can be carried in the Knapsack is "<<max_val;
    cout <<endl<<endl;

    //Printing out the selected items and its corresponding value
    wt=W;
    for (i=n; i>=0;i--)
    {
        if (max_val<=0)
            break;

        else if ( max_val == Knap[i-1][wt])
            continue;

        else
            cout<<"Item "<<i<<" is selected and its value is "<<v[i-1]<<endl;
            max_val=max_val-v[i-1];
            wt=wt-w[i-1];
    }

    cout<<endl;
}

 //Driver Code

int main()
{
    //Taking user input
    int num, capacity;

    cout<<"Enter the total capacity of the Knapsack: ";
    cin>>capacity;

    cout<<endl<<"Enter the number of items in the Knapsack: ";
    cin>>num;



    int V[num], W[num];
    for(int i=0; i<num; i++)
    {
        cout<<endl<<"Enter the weight of the item "<<i<<": ";
        cin>>W[i];

        cout<<"Enter the value for the item "<<i<<": ";
        cin>>V[i];
    }

    //Calling the knapsack function to perform 0/1 knapsack algorithm
     knapSack( capacity, W, V, num);

    return 0;
}
```

OUTPUT:



```
Enter the total capacity of the Knapsack: 5

Enter the number of items in the Knapsack: 4

Enter the weight of the item 1: 2
Enter the value for the item 1: 3

Enter the weight of the item 2: 3
Enter the value for the item 2: 4

Enter the weight of the item 3: 4
Enter the value for the item 3: 5

Enter the weight of the item 4: 5
Enter the value for the item 4: 6

The table is as follows:
0  0  0  0  0  0
0  0  3  3  3  3
0  0  3  4  4  7
0  0  3  4  5  7
0  0  3  4  5  7

The maximum value that can be carried in the Knapsack is 7

Item 2 is selected and its value is 4
Item 1 is selected and its value is 3

Program ended with exit code: 0
```