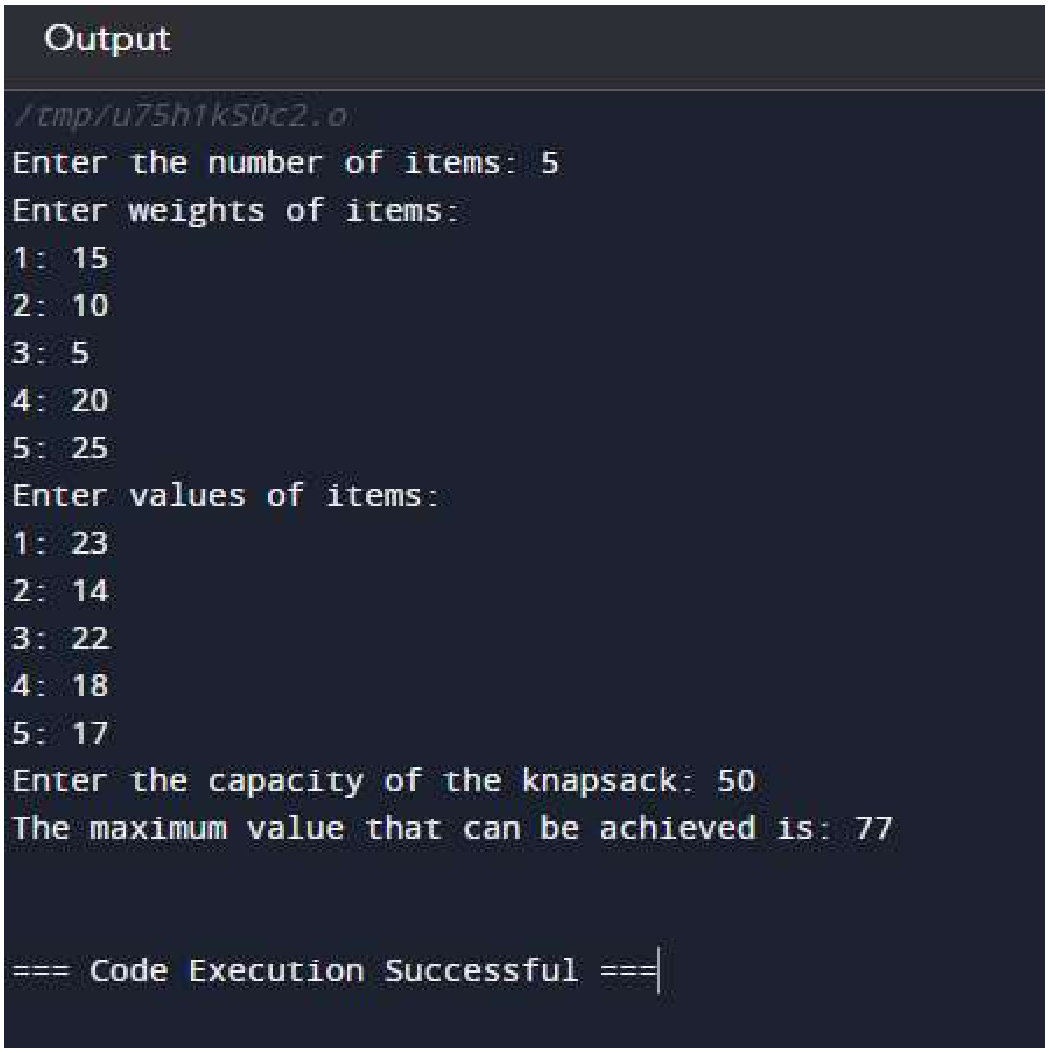
0-1 KNAPSACK

CODE:

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

int knapsack(vector<int>& weights, vector<int>& values, int capacity, int
index, vector<vector<int>>& memo) {
   if (index < 0 || capacity <= 0) {
      return 0;
   }

   if (memo[index][capacity] != -1) {
      return memo[index][capacity];
   }

   if (weights[index] > capacity) {
      memo[index][capacity] = knapsack(weights, values, capacity, index - 1,
memo);
   } else {
      int withItem = values[index] + knapsack(weights, values, capacity -
weights[index], index - 1, memo);
      int withoutItem = knapsack(weights, values, capacity, index - 1,
memo);
      memo[index][capacity] = max(withItem, withoutItem);
   }

   return memo[index][capacity];
}

int main() {
    int n;
    cout << "Enter the number of items: ";
    cin >> n;

    vector<int> weights(n);
    vector<int> values(n);

    cout << "Enter weights of items:\n";
    for (int i = 0; i < n; ++i) {
        cin >> weights[i];
    }

    cout << "Enter values of items:\n";
    for (int i = 0; i < n; ++i) {
        cin >> values[i];
    }

    int capacity;
    cout << "Enter the capacity of the knapsack: ";
    cin >> capacity;

    vector<vector<int>> memo(n, vector<int>(capacity + 1, -1));

    int maxValue = knapsack(weights, values, capacity, n - 1, memo);

    cout << "The maximum value that can be achieved is: " << maxValue <<
endl;

    return 0;
}
```

OUTPUT:



```
/tmp/u75h1kS0c2.o
Enter the number of items: 5
Enter weights of items:
1: 15
2: 10
3: 5
4: 20
5: 25
Enter values of items:
1: 23
2: 14
3: 22
4: 18
5: 17
Enter the capacity of the knapsack: 50
The maximum value that can be achieved is: 77


=== Code Execution Successful ===
```