**Program – 15**

**Aim – Write an algorithm and program to implement 0/1 Knapsack.**

**Algorithm –**

**KNAPSACK (n, W)**

1. for w = 0, W

2. do V [0,w] ← 0

3. fori=0, n

4. do V [i, 0] ← 0

5. for w = 0, W

6. do if (wi≤ w & vi + V [i-1, w - wi]> V [i -1,W])

7. then V [i, W] ← vi + V [i - 1, w - wi]

8. else V [i, W] ← V [i - 1, w]

**Source Code –**

#include <bits/stdc++.h>

using namespace std;

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

intknapSack(int W, intwt[], intval[], int n)

{

if (n == 0 || W == 0)

return 0;

if (wt[n - 1] > W)

returnknapSack(W, wt, val, n - 1);

else

return max(

val[n - 1]

+ knapSack(W - wt[n - 1],

wt, val, n - 1),

knapSack(W, wt, val, n - 1));

}

int main()

{

intval[] = { 60, 100, 120 };

intwt[] = { 10, 20, 30 };

int W = 50;

int n = sizeof(val) / sizeof(val[0]);

cout<<knapSack(W, wt, val, n);

return 0;

}