LUGGAGE PACKER

A

Mini Project Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

By

BAKSHI ABHINITH

1602-19-733-124

ZUBAIR AHMED

1602-19-733-182

Department of Computer Science & Engineering

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2021

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Hyderabad-500 031

Department of Computer Science & Engineering

DECLARATION BY THE CANDIDATE

I, **BAKSHI ABHINITH,** bearing hall ticket number, **1602-19-733-124**, hereby declare that the project report entitled **“LUGGAGE PACKER”** Department of Computer Science & Engineering, VCE, Hyderabad, is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Computer Science & Engineering**.

This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

Bakshi Abhinith,
1602-19-733-124.

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Hyderabad-500 031

Department of Computer Science & Engineering

BONAFIDE CERTIFICATE

This is to certify that the project entitled **“LUGGAGE PACKER”** being submitted by **BAKSHI ABHINITH,** bearing **1602-19-733-124,** in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science & Engineering is a record of bonafide work carried out by him/her under my guidance.

Ms. Sunita Reddy

Assistant Professor

Dept. of CSE

ACKNOWLEDGEMENT

With immense pleasure, we record our deep sense of gratitude to our guide Ms.Sunita Reddy, Assistant Professor, Vasavi College of Engineering, Hyderabad, for the valuable guidance and suggestions, keen interest and thorough encouragement extended throughout the period of the project work. I consider myself lucky enough to be part of this project. This project would add as an asset to my academic profile.

We express our thanks to all those who contributed for the successful completion of our project work.

ABSTRACT

The knapsack problem is an important combinatorial optimization problem that models binary and discrete decisions given limited resources. This project addresses theoretical, and algorithmic issues regarding two of the knapsack problem variations, with application to packaging luggage. This project presents dynamic programming algorithms that produce optimal solutions in pseudo-polynomial time, greedy heuristics, and fully polynomial time approximation schemes.

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

INTRODUCTION

Knapsack is basically means bag. A bag of given capacity.

We want to pack n items in your luggage.

- The ith item is worth vi dollars and weight wi pounds.
- Take as valuable a load as possible, but cannot exceed W pounds.
- V_i , w_i , W are integers.

2.1 OVERVIEW

This project helps the users pack their bags when they're in a state of dilemma of packing their things while going on a trip.

The user first gives the list of items he/she wants to pack in their bags and their importance.Also, he provides the maximum weight he can carry in his backpack.

We then offer the user two approaches by which his problem of packaging an be solved- The Greedy Method and the Dynamic programming method.

The greedy method gives the most importance to the items of the greater importacnce while the dynamic programming method gives the optimal set of items that the user can carry along with him.

2.2 APPROACHES USED

1. Greedy Method :

   The basic idea of the greedy approach is to calculate the ratio value/weight for each item and sort the item on basis of this ratio. Then take the item with the highest ratio and add them until we can't add the next item as a whole and at the end add the next item as much as we can. Which will always be the optimal solution to this problem. After sorting we need to loop over these items and add them in our knapsack satisfying above-mentioned criteria.

2. Dynamic Programming Approach :

   In this item cannot be broken which means the user should take the item as a whole or should leave it. That's why it is called **0/1 knapsack Problem**.

   - Each item is taken or not taken.
   - Cannot take a fractional amount of an item taken or take an item more than once.

2.3 PROBLEM STATEMENT

Given a knapsack with a limited weight capacity and few items having some weight and value,

We have to select the items that can be placed into the knapsack such that

(i) The profit obtained by placing those items in the knapsack is maximum

(ii) The weight limit of knapsack does not exceed

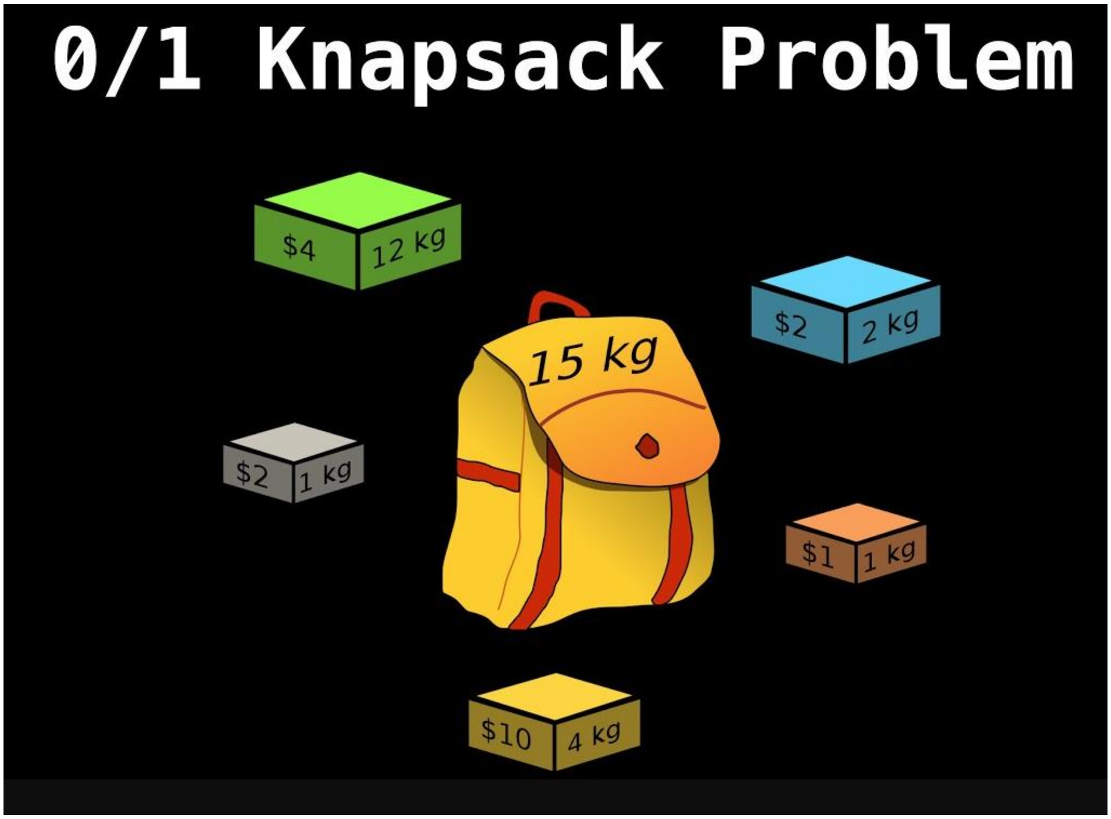


Figure 2.3.1

2.4 OBJECTIVE

To help users pack their bags efficiently so that ach item of utmost importance is included in their luggage .

Figure 2.4.1

SYSTEM REQUIREMENTS

Hardware:

- Minimum RAM required: 512 MB
- Input devices: Mouse, Keyboard
- Output devices: Monitor

Software:

- CODE::BLOCKS IDE
- Windows 7 or above

IMPLEMENTATION

```
#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>


int n;   //Number of items

int W;   //Luggage Weight Limit


struct knapsackGreedy{

        int value;

        int weight;

        int no;

};


int comp(const void *a,const void *b)

{

   struct knapsackGreedy *x = (struct knapsackGreedy *)a;

   struct knapsackGreedy *y = (struct knapsackGreedy *)b;

   double p= (double)(x->value)/(double)(x->weight);

   double q = (double)(y->value)/(double)(y->weight);

   return p<q;

}
```

```
int max(int a, int b)

{

    return (a > b)? a : b;

}


int DPKnapsack(int wt[], int val[])

{

   int i, w;

   int K[n+1][W+1]; //T-Table

   for (i = 0; i <= n; i++)

   {

       for (w = 0; w <= W; w++)

       {

           if (i==0 || w==0)

               K[i][w] = 0;

           else if (wt[i-1] <= w)

                 K[i][w] = max(val[i-1] + K[i-1][w-wt[i-1]],  K[i-1][w]);

           else

                 K[i][w] = K[i-1][w];

       }

   }

   int result = K[n][W];
   int res = result;

   w = W;

   printf("Items you must put in your luggage are : \n");

    for (i = n; i > 0 && res > 0; i--)

    {

        if (res == K[i - 1][w])

            continue;

        else

        {

            if(wt[i-1]==1400)

                printf("LAPTOP\n");

            else if(wt[i-1]==75)

                printf("FOOTBALL\n");

            else if(wt[i-1]==2000)

                printf("BOOKS\n");

            else if(wt[i-1]==800)

                printf("PILLOWS AND BLANKETS\n");

            else if(wt[i-1]==3500)

                printf("FOOD AND GROCERIES\n");

            else if(wt[i-1]==1500)

                printf("SOFT DRINKS\n");

            else if(wt[i-1]==4500)

                printf("CLOTHES\n");
```

```
            else if(wt[i-1]==950)

                printf("CAMERA\n");

            else if(wt[i-1]==6000)

                printf("CRICKET KIT\n");

            else if(wt[i-1]==2700)

                printf("SHOES\n");


            res = res - val[i - 1];

            w = w - wt[i - 1];

        }

    }

    return result;

}


int GreedyKnapsack()

{

            struct knapsackGreedy k[n];

            printf("Enter the Sno. and values of all the items you wish to keep in the
luggage : \n");

            for(int i=0;i<n;i++)

    {

        printf("Item %d : ",i+1);

                        scanf("%d %d",&k[i].no,&k[i].value);
    if(k[i].no==1)
        k[i].weight = 1400;
    else if(k[i].no==2)
        k[i].weight = 75;
    else if(k[i].no==3)
        k[i].weight = 2000;
    else if(k[i].no==4)
        k[i].weight = 800;
    else if(k[i].no==5)
        k[i].weight = 3500;
    else if(k[i].no==6)
        k[i].weight = 1500;
    else if(k[i].no==7)
        k[i].weight = 4500;
    else if(k[i].no==8)
        k[i].weight = 950;
    else if(k[i].no==9)
        k[i].weight = 6000;
    else if(k[i].no==10)
        k[i].weight = 2700;

}
```

```
    qsort((void*)k,n,sizeof(struct knapsackGreedy),comp);

    printf("Items you must put in your luggage are : \n");

    long long wsum=0;

    long long vsum=0;

    for(int i=0;i<n;i++)

    {

                if(wsum+k[i].weight <= W)

{

                            wsum+=k[i].weight;

                            vsum+=k[i].value;

                            if(k[i].no==1)

        printf("LAPTOP\n");

    else if(k[i].no==2)

        printf("FOOTBALL\n");

    else if(k[i].no==3)

        printf("BOOKS\n");

    else if(k[i].no==4)

        printf("PILLOWS AND BLANKETS\n");

    else if(k[i].no==5)

        printf("FOOD AND GROCERIES\n");

    else if(k[i].no==6)

        printf("SOFT DRINKS\n");

    else if(k[i].no==7)
```

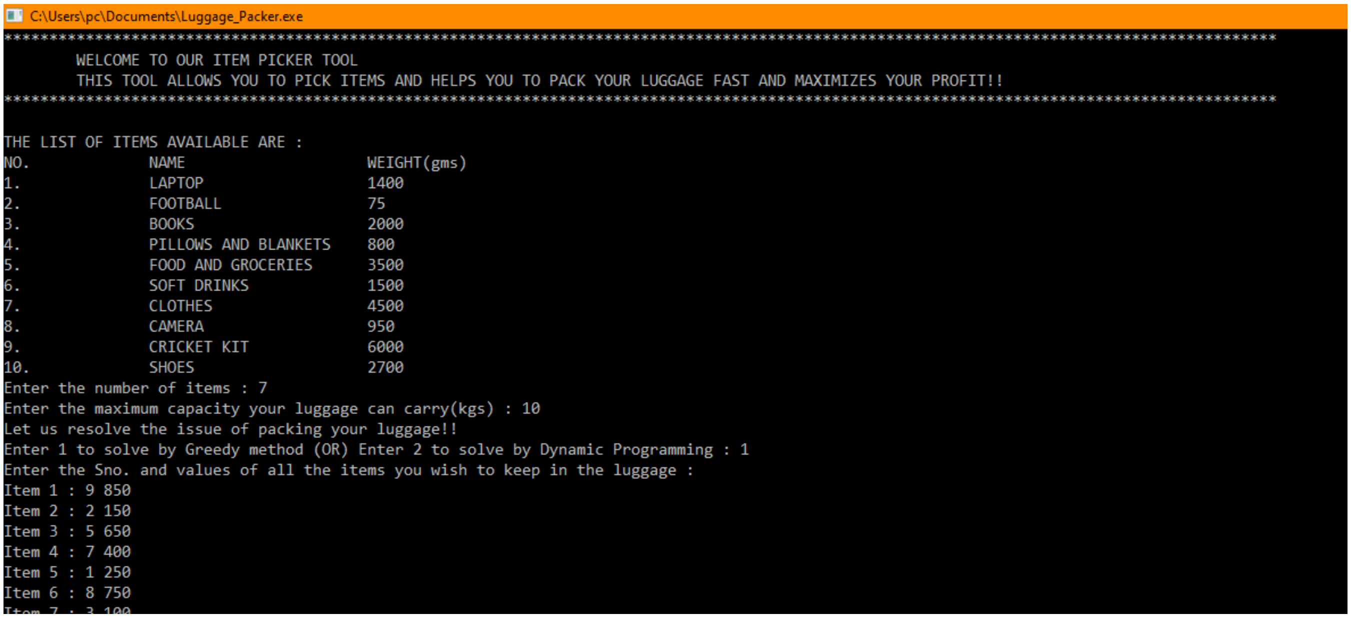
```
            printf("CLOTHES\n");
          else if(k[i].no==8)
            printf("CAMERA\n");
          else if(k[i].no==9)
            printf("CRICKET KIT\n");
          else if(k[i].no==10)
            printf("SHOES\n");


        }
                /*else
        {
                        vsum+=k[i].value*(double)(W-wsum)/(double)(k[i].weight);
          break;
        }*/
    }
    return vsum;
}

int main()
{

printf("****************************************************************
****************************************************************************
*********\n");
   printf("\tWELCOME TO OUR ITEM PICKER TOOL\n");
   printf("\tTHIS TOOL ALLOWS YOU TO PICK ITEMS AND HELPS YOU TO
PACK YOUR LUGGAGE FAST AND MAXIMIZES YOUR PROFIT!!\n");

printf("***********************************************************************
****************************************************************************
*********\n\n");
   printf("THE LIST OF ITEMS AVAILABLE ARE : \n");
   //Weight in grams
   printf("NO.\t\tNAME\t\t\tWEIGHT(gms)\n");
   printf("1.\t\tLAPTOP\t\t\t1400\n");
   printf("2.\t\tFOOTBALL\t\t75\n");
   printf("3.\t\tBOOKS\t\t\t2000\n");
   printf("4.\t\tPILLOWS AND BLANKETS    800\n");
   printf("5.\t\tFOOD AND GROCERIES      3500\n");
   printf("6.\t\tSOFT DRINKS\t\t1500\n");
   printf("7.\t\tCLOTHES\t\t\t4500\n");
   printf("8.\t\tCAMERA\t\t\t950\n");
   printf("9.\t\tCRICKET KIT\t\t6000\n");
   printf("10.\t\tSHOES\t\t\t2700\n");


   int choice;
```

```
    int again=0;
    while(again==0)
    {
        printf("Enter the number of items : ");
        scanf("%d",&n);
        int num[n],val[n],wt[n];
        printf("Enter the maximum capacity your luggage can carry(kgs) : ");
        scanf("%d",&W);
        W = 1000*W;
        printf("Let us resolve the issue of packing your luggage!!\n");
        printf("Enter 1 to solve by Greedy method (OR) Enter 2 to solve by Dynamic Programming : ");
        scanf("%d",&choice);
        if(choice==1)
        {
            int res = GreedyKnapsack();
            printf("Maximum profit that is possible by putting all items in the Knapsack is : %d\n",res);
        }
        if(choice==2)
        {
            printf("Enter the Snos. and values of all the items you wish to put in the bag :\n");
```

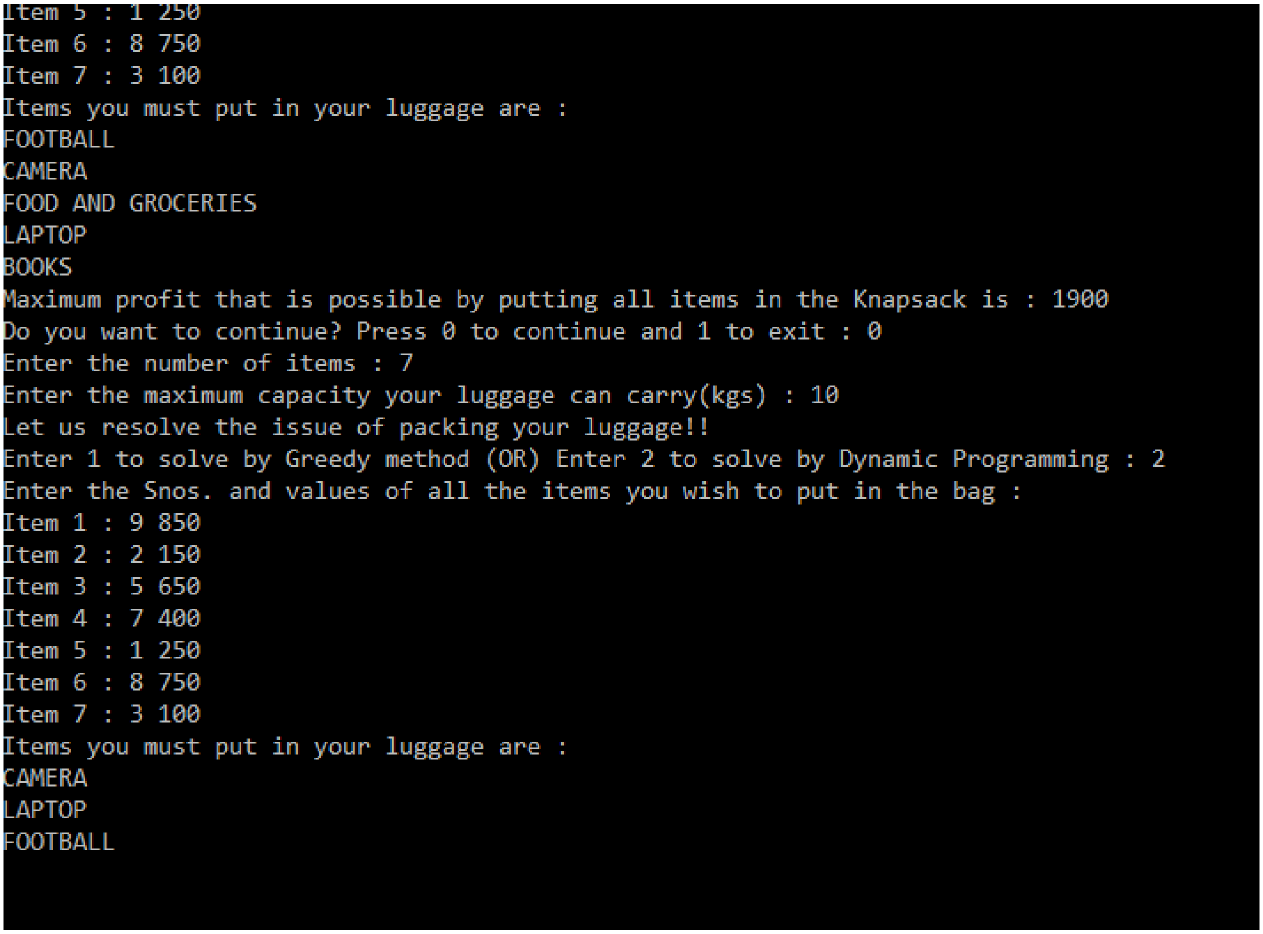
```
for(int i=0;i<n;i++)
{
    printf("Item %d : ",i+1);
    scanf("%d %d",&num[i],&val[i]);
    if(num[i]==1)
        wt[i] = 1400;
    else if(num[i]==2)
        wt[i] = 75;
    else if(num[i]==3)
        wt[i] = 2000;
    else if(num[i]==4)
        wt[i] = 800;
    else if(num[i]==5)
        wt[i] = 3500;
    else if(num[i]==6)
        wt[i] = 1500;
    else if(num[i]==7)
        wt[i] = 4500;
    else if(num[i]==8)
        wt[i] = 950;
    else if(num[i]==9)
        wt[i] = 6000;
    else if(num[i]==10)
```

```
                wt[i] = 2700;


            }

            int ans = DPKnapsack(wt,val);

            printf("Maximum profit that is possible by putting all items in the Knapsack is
: %d\n",ans);

        }

        printf("Do you want to continue? Press 0 to continue and 1 to exit : ");

        scanf("%d",&again);

        if(again==1)

        {

            printf("THANK YOU FOR USING OUR TOOL!!");

            return 0;

        }

    }

}
```

OUTPUT OF THE PROGRAM



```
***********************************************************************************************************************************************
        WELCOME TO OUR ITEM PICKER TOOL
        THIS TOOL ALLOWS YOU TO PICK ITEMS AND HELPS YOU TO PACK YOUR LUGGAGE FAST AND MAXIMIZES YOUR PROFIT!!
***********************************************************************************************************************************************

THE LIST OF ITEMS AVAILABLE ARE :
NO.             NAME                    WEIGHT(gms)
1.              LAPTOP                  1400
2.              FOOTBALL                75
3.              BOOKS                   2000
4.              PILLOWS AND BLANKETS    800
5.              FOOD AND GROCERIES      3500
6.              SOFT DRINKS             1500
7.              CLOTHES                 4500
8.              CAMERA                  950
9.              CRICKET KIT             6000
10.             SHOES                   2700
Enter the number of items : 7
Enter the maximum capacity your luggage can carry(kgs) : 10
Let us resolve the issue of packing your luggage!!
Enter 1 to solve by Greedy method (OR) Enter 2 to solve by Dynamic Programming : 1
Enter the Sno. and values of all the items you wish to keep in the luggage :
Item 1 : 9 850
Item 2 : 2 150
Item 3 : 5 650
Item 4 : 7 400
Item 5 : 1 250
Item 6 : 8 750
Item 7 : 3 100
```



```
Item 5 : 1 250
Item 6 : 8 750
Item 7 : 3 100
Items you must put in your luggage are :
FOOTBALL
CAMERA
FOOD AND GROCERIES
LAPTOP
BOOKS
Maximum profit that is possible by putting all items in the Knapsack is : 1900
Do you want to continue? Press 0 to continue and 1 to exit : 0
Enter the number of items : 7
Enter the maximum capacity your luggage can carry(kgs) : 10
Let us resolve the issue of packing your luggage!!
Enter 1 to solve by Greedy method (OR) Enter 2 to solve by Dynamic Programming : 2
Enter the Snos. and values of all the items you wish to put in the bag :
Item 1 : 9 850
Item 2 : 2 150
Item 3 : 5 650
Item 4 : 7 400
Item 5 : 1 250
Item 6 : 8 750
Item 7 : 3 100
Items you must put in your luggage are :
CAMERA
LAPTOP
FOOTBALL
```

CONCLUSIONS

The main aim of the project is to assist the User to pack their luggage efficiently, which was achieved.

We would like to extend this project by including the branch and bound approach to solve the problem.

REFERENCES

- https://www.geeksforgeeks.com
- https://stackoverflow.com/
- https://www.youtube.com
- Ellis Horowitz, Sartaj Sahani, Sanguthevar Rajasekaran,“ Fundamentals of computer Algorithms”, Second edition (2008),Universities Press.