DSA PROJECT

Winter Semester 2017-18

ABSTRACT

The Three-Dimensional Bin Packing Problem (3BP) consists of allocating, without overlapping, a given set of three dimensional rectangular items to three-dimensional identical finite bins. The problem is NP hard in the strong sense, and finds many industrial applications. We in this process have made used to heuristic approach rather than greedy algorithm.

INTRODUCTION

Given a set of n three-dimensional rectangular items, each characterized by width w_j, height h_j and depth d_j and an unlimited number of identical three-dimensional rectangular containers (bins) having width 2.3m, height 3 m and depth 12 m, the Three-Dimensional Bin Packing Problem (3BP) consists of orthogonally packing, without overlapping, all the items into the minimum number of bins. It is assumed that the items have fixed orientation, i.e., they cannot be rotated. We will call base of an item (resp. bin) its $w_j * h_j$ (resp. W*H) side. We assume, without loss of generality, that all the input data are positive integers, and that $w_j < 2.3m$, $h_j < 3m$ and $d_j < 12m$

Three-dimensional packing problems have relevant practical interest in *industrial applications* such as, e.g., cutting of foam rubber in arm-chair production, container and pallet loading, and packaging design. Although 3BP is a simplified version of real-world problems, in many cases it arises as a sub problem.

In our project we have approached it with heuristic algorithm in this the solution yielded may not be the best or efficient with respect to the problem but it is one of many solutions possible for the problem while in greedy algorithm the output obtained is generally the more efficient and optimum for the problem

The algorithm starts by sorting the items by non-increasing height. The items are then partitioned into clusters, each characterized by a different height, as follows. The first (tallest) item j defines the first cluster with height h_j, which includes all items having height h_k satisfying $h_k <= bh_j$, where b (b belongs to 0-1) is a prefixed parameter. The tallest item s for which $h_s < bh_j$ defines the next cluster with height h_s, and so on. The items in each cluster are then sorted by non-increasing w_j*d_j value, and the item set is renumbered, by increasing cluster, following the new order.

Note that the value of b determines the way the items are partitioned into clusters. A value close to zero produces very few clusters and disregards the item heights, while a value close to one produces a very large number of clusters and disregards the area of the item bases. A good value should appropriately take into account both information. Through preliminary experimental evaluations we determined b ¼ 0:75 as the value producing, on average, the partitions leading to the best packing.

REAL TIME PROBLEM

During house shifting moving of goods from one place to other is quite a difficult task we are developing an algorithm to allocate the object to the vehicle used for this purpose whose dimensions are 2.3*3*12 m3 (w*h*d which are standard dimensions)

ALGORITHM

Input

-no. of boxes

-dimensions of boxes

PROCESS

1)we input the width, height and depth of boxes and store them in W, H, D array respectively

2)we calculate base area of each boxes as width*depth and store them in array name

3)we first sort height in decreasing order and store it in array

4)in array of base area it is sorted according to their highest in decreasing order i.e. the base area of highest block will be stored in the 1st index position of the array

5)now we created clusters of boxes according to their height each cluster contains boxes with height > 0.75*h (where h is height of the highest block)

6)the cluster created are stored in a matrix with first row containing the first cluster, second containing the second and so on if. the number of columns and rows of matrix created are predefined.

7)corresponding to each element in the cluster matrix another matrix is created which contains base area of the box at the same position

8)the base area of each cluster is then also sorted by doing this we follow *height first area second* approach, choosing the two best possible solutions

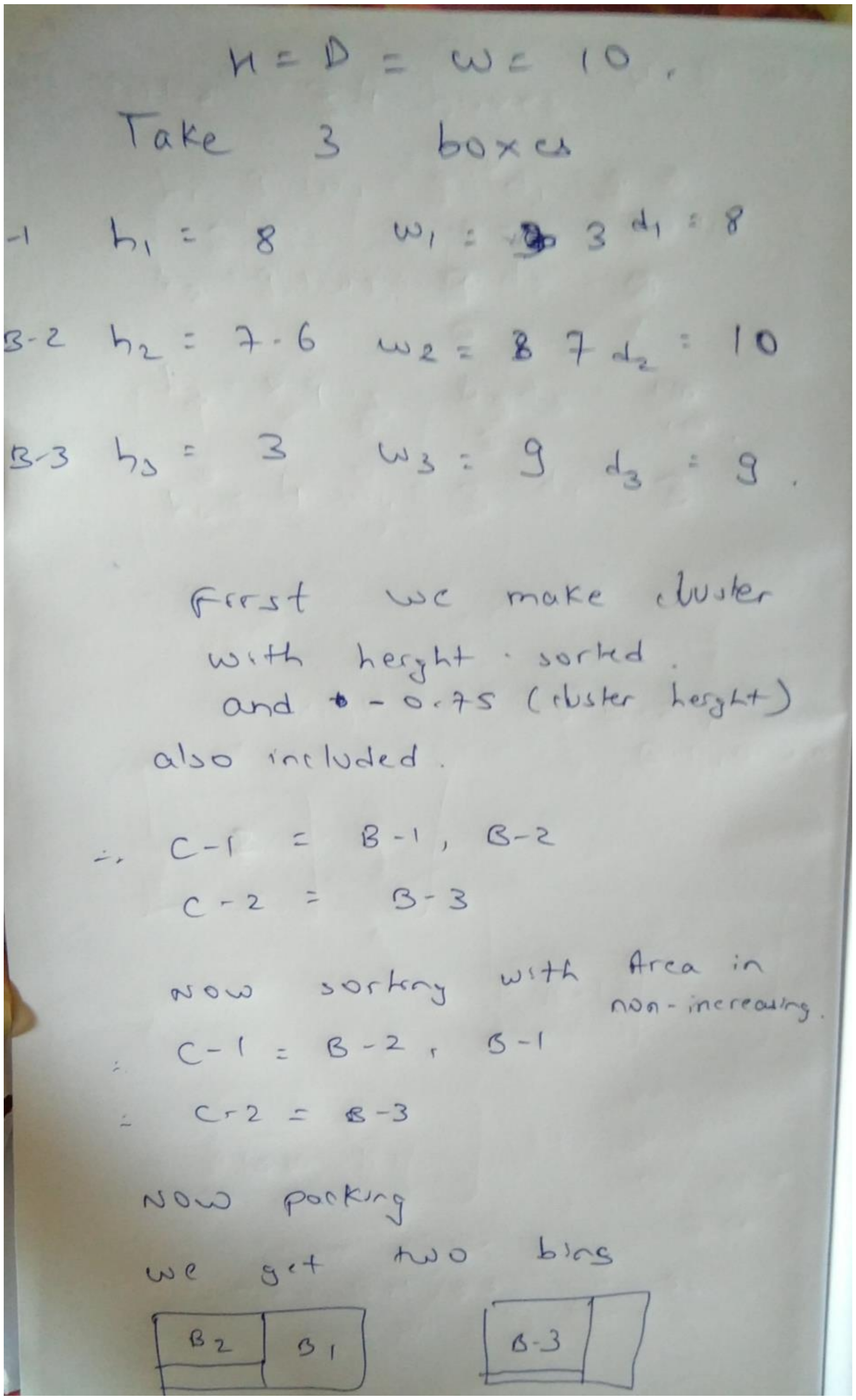
By organizing the objects first into cluster according to height and then according to the base area we have enhanced the efficiency of packing of boxes into the bin

9)now the bins are packed with the boxes.

10)If the first bin's base area is insufficient to accommodate another box (Total Area Used > 27.6 m^2), a new bin is used

11) now the same step is repeated until all the boxes are accommodated

SAMPLE QUESTION:



$H = D = W = 10.$

Take 3 boxes

-1 $h_1 = 8$ $w_1 = $ ~~8~~ 3 $d_1 = 8$

B-2 $h_2 = 7.6$ $w_2 = $ ~~8~~ 7 $d_2 = 10$

B-3 $h_3 = 3$ $w_3 = 9$ $d_3 = 9.$

First we make cluster with height sorted. and - 0.75 (cluster height) also included.

∴ C-1 = B-1, B-2

C-2 = B-3

Now sorting with Area in non-increasing.

∴ C-1 = B-2, B-1

∴ C-2 = B-3

Now packing we get two bins

BASE PAPER EXPLANATION:

Basically The Base Paper Shows the implementation in two Phases:

PHASE 1:

The algorithm starts by sorting the items by non-increasing height. The items are then partitioned into clusters, each characterized by a different height, as follows. The first (tallest) item j defines the first cluster with height hj, which includes all items having height hk satisfying hk<=b*hj, where b (b belongs to(0,1)   ) is a prefixed para-meter. The tallest item s for which hs < bhj defines the next cluster with height hs, and so on. The items in each cluster are then sorted by non-increasing wjdj value, and the item set is renumbered, by increasing cluster, following the new order.

PHASE 2:

The items are directly sorted and re-numbered by non-increasing wjdj value. These items are now packed within the bins. The best solution is finally selected.

CODE :

```
#include<stdio.h>
int main()
{
int n,i;
int key =0;
int j,x=0;
float a,b;
printf("\t\t\t\tWelcome To ABS PACKERS AND MOVERS.\n\n");
printf("\t\t\t\tRespecting Fragility since 1999.\n\n");
printf("Please Read the Details Before Proceeding Further.\n\n");
printf("Maximum Permissible Dimension Are As Follows: \n\t WIDTH -
2.3m. \n\t DEPTH - 12m \n\t HEIGHT - 3m.\n\n");
printf("Please Enter the number of Boxes to be Transported : ");
scanf("%d",&n);
float depth[n];
float height[n];
float width[n];
for(i=0;i<n;i++)
{
printf("\nEnter the Width in metre of Box No-%d : ",i+1);
```

```
scanf("%f",&width[i]);
if(width[i]>2.3 || width[i]<0)

{
    printf("\n\tEntered value was not in the Specified Range.\n");
    printf("\n\tPlease Re-Enter the Width in metre of Box No-%d : ",i+1);
    scanf("%f",&width[i]);
}

printf("\nEnter the Depth in metre of Box No-%d  : ",i+1);
scanf("%f",&depth[i]);
if(depth[i]>12 || depth[i]<0)

{
   printf("\n\tEntered value was not in the Specified Range.\n");
     printf("\n\tPlease Re-Enter the Depth in metre of Box No-%d  : ",i+1);
scanf("%f",&depth[i]);
}

printf("\nEnter the Height in metre of Box No-%d : ",i+1);
scanf("%f",&height[i]);
if(height[i]>3 || height[i]<0)
```

```
{
printf("\n\tEntered value was not in the Specified Range.\n");
     printf("\n\tPlease Re-Enter the Height in metre of Box No-%d  :
",i+1);
    scanf("%f",&height[i]);
}
printf("\n\t\t\t\t\tDimensions Of The Box Have Been Stored.\n");
}
float heightsort[n];
for(i=0;i<n;i++)

{
heightsort[i]=height[i];
}


float basearea[n];
for(i=0;i<n;i++)

{
   basearea[i]=depth[i]*width[i];
}
```

```
float temp3,temp4;
        for (i = 0 ; i < ( n - 1 ); i++){
                for (j= 0 ; j < n - i - 1; j++){
                        if(heightsort[j] < heightsort[j+1]){
                        temp3=heightsort[j];
                        heightsort[j]   = heightsort[j+1];
                        heightsort[j+1] = temp3;
                        temp4=basearea[j];
                        basearea[j]   = basearea[j+1];
                        basearea[j+1] = temp4;
                        }
                }
        }
printf("\n\n\nBase Area Of Each Box: \n");
for(i=0;i<n;i++)

{
   printf("\n%f\n",basearea[i]);
}

float cluster[n][n];
float clusterarea[n][n];
```

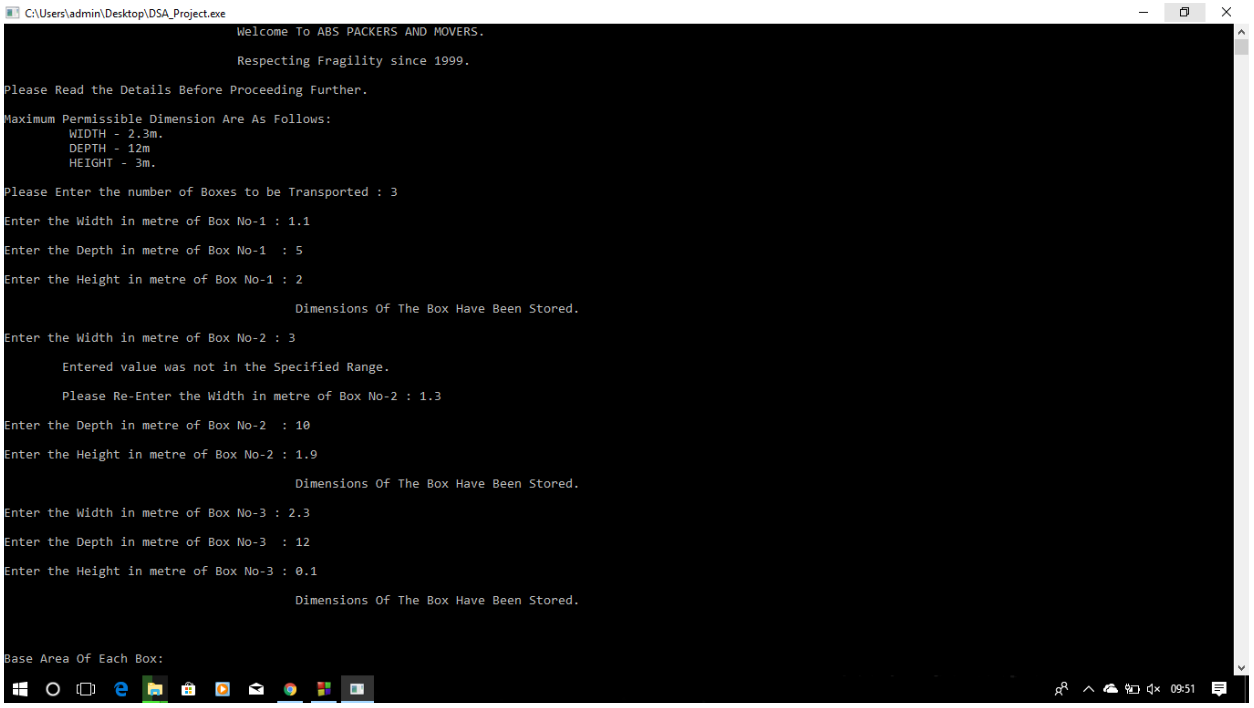
```
for(i=0;i<n;i++)
 {
     for(j=0;j<n;j++)
{
    cluster[i][j]=0;
    clusterarea[i][j]=0;
}
 }
int sm=0;
for(i=0;i<n;i++)
    {
        a = heightsort[key];
        for(j=key;j<n;j++)
        {
            if(heightsort[j]>=(0.75*a))
            {
                cluster[i][j-x] = heightsort[j];
                clusterarea[i][j-x]= basearea[sm];
                sm++;
            }
            else
            {
                x=j;
                key=j;
                break;
            }
            if(j==(n-1))
            {
                i=n-1;
            }
        }
    }
    printf("\n\n\nCluster Containing Height Elements Of Same Height or
Greater 0.75*Layer Height.\n");
  for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("\t%f",cluster[i][j]);
        }
        printf("\n");
    }

printf("\n\n\nCluster Wise Area Initial\n");
```

```
 for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("\t%f",clusterarea[i][j]);
        }
        printf("\n");
    }
int k;
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < n; ++j)
        {
            for (k =(j + 1); k < n; ++k)
            {
                if (clusterarea[i][j] < clusterarea[i][k])
                {
                    a = clusterarea[i][j];
                    clusterarea[i][j] = clusterarea[i][k];
                    clusterarea[i][k] = a;
                }
            }
        }
    }

    printf("\n\n\nCluster Wise Base Area Sorted : \n");
    for(i=0;i<n;i++)

    {
        for(j=0;j<n;j++)
        {
            printf("\t%f",clusterarea[i][j]);
        }
        printf("\n");
    }

    int bincount=1;
    float sum;
    for(i=0;i<n;i++)

    {
        for(j=0;j<n;j++)
        {
            if (clusterarea[i][j]>0)
```

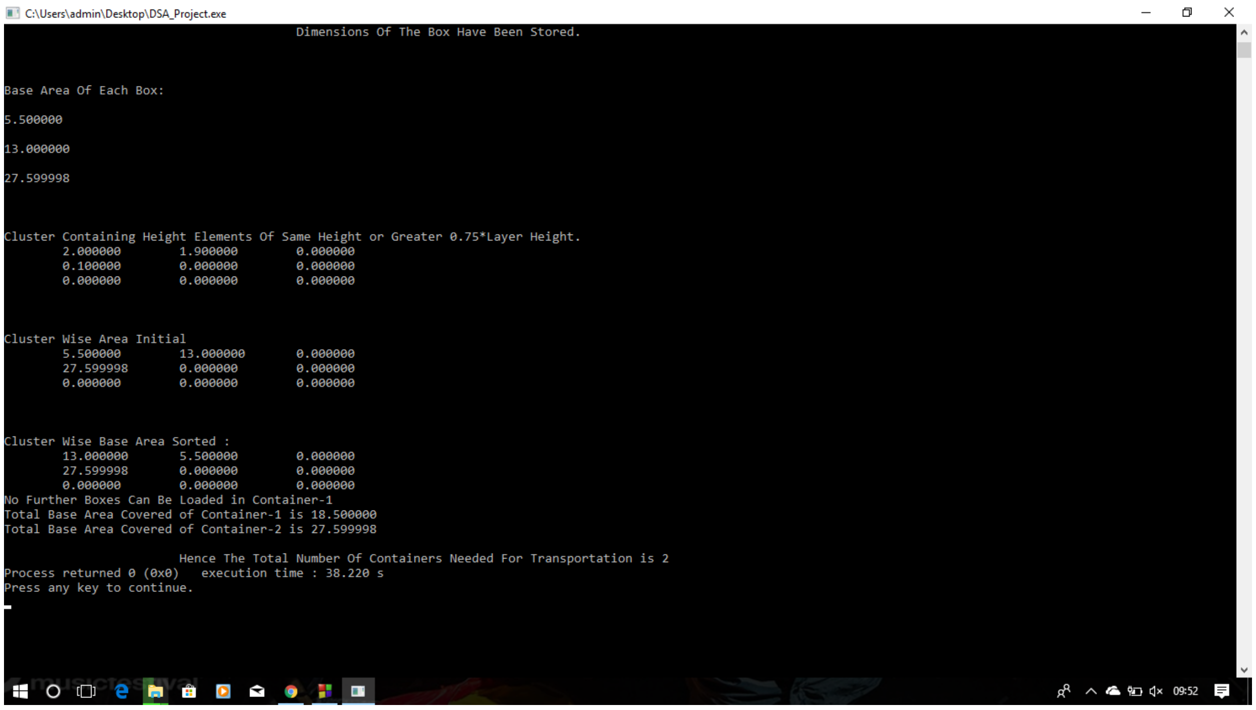
```
            {

              sum=sum+clusterarea[i][j];
              if(sum>27.6)

              {
                   sum=sum-clusterarea[i][j];
                  printf("No Further Boxes Can Be Loaded in Container-
%d",bincount);
                  printf("\nTotal Base Area Covered of Container-%d is
%f",bincount,sum);
                  sum=clusterarea[i][j];
                  bincount++;
              }
            }
        }
    }
printf("\nTotal Base Area Covered of Container-%d is %f",bincount,sum);

printf("\n\n\t\t\tHence The Total Number Of Containers Needed For
Transportation is %d",bincount);
}
```

OUTPUT :

TEST CASE -1

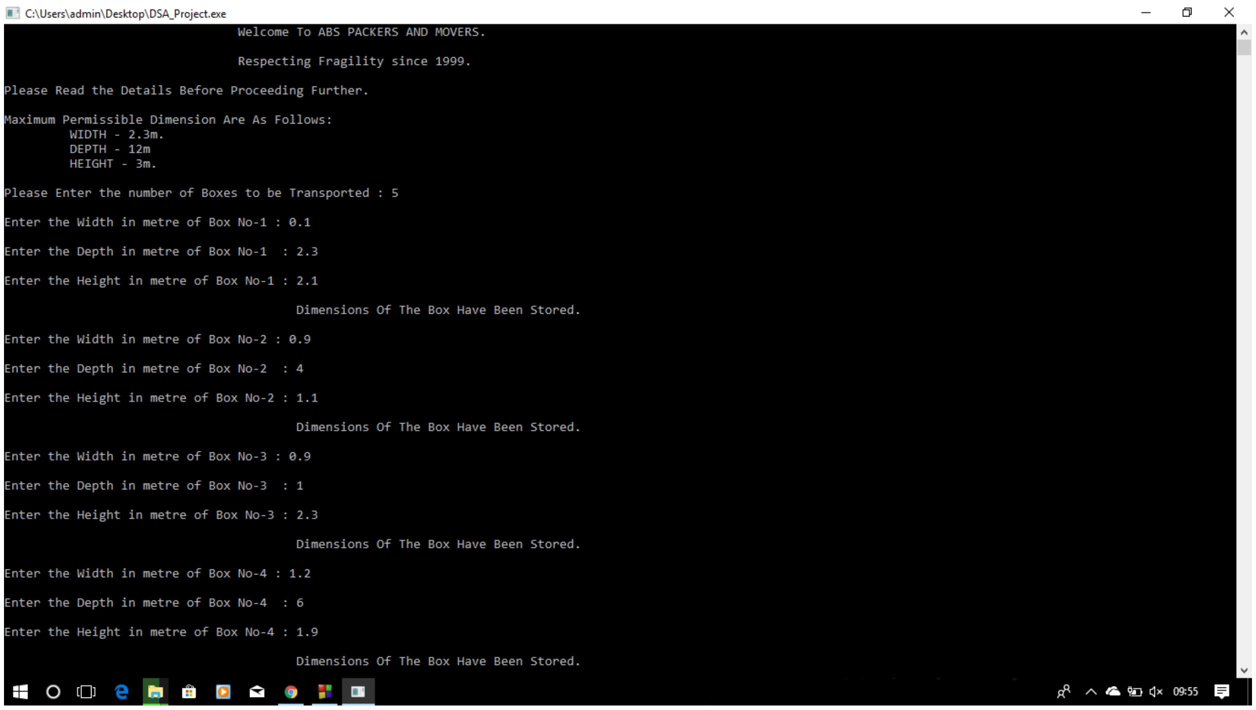


```
C:\Users\admin\Desktop\DSA_Project.exe
                                Welcome To ABS PACKERS AND MOVERS.

                                Respecting Fragility since 1999.

Please Read the Details Before Proceeding Further.

Maximum Permissible Dimension Are As Follows:
         WIDTH - 2.3m.
         DEPTH - 12m
         HEIGHT - 3m.

Please Enter the number of Boxes to be Transported : 3

Enter the Width in metre of Box No-1 : 1.1

Enter the Depth in metre of Box No-1  : 5

Enter the Height in metre of Box No-1 : 2

                                        Dimensions Of The Box Have Been Stored.

Enter the Width in metre of Box No-2 : 3

        Entered value was not in the Specified Range.

        Please Re-Enter the Width in metre of Box No-2 : 1.3

Enter the Depth in metre of Box No-2  : 10

Enter the Height in metre of Box No-2 : 1.9

                                        Dimensions Of The Box Have Been Stored.

Enter the Width in metre of Box No-3 : 2.3

Enter the Depth in metre of Box No-3  : 12

Enter the Height in metre of Box No-3 : 0.1

                                        Dimensions Of The Box Have Been Stored.


Base Area Of Each Box:
```



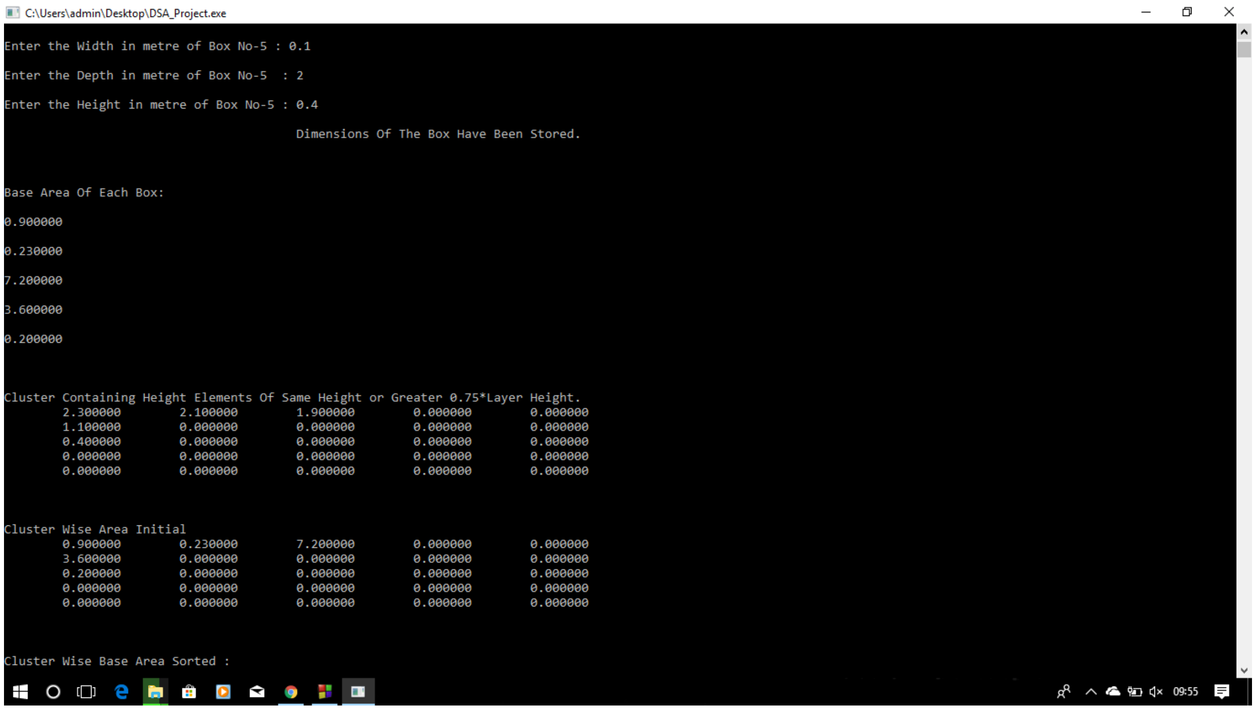
```
C:\Users\admin\Desktop\DSA_Project.exe
                                        Dimensions Of The Box Have Been Stored.


Base Area Of Each Box:

5.500000

13.000000

27.599998



Cluster Containing Height Elements Of Same Height or Greater 0.75*Layer Height.
        2.000000        1.900000        0.000000
        0.100000        0.000000        0.000000
        0.000000        0.000000        0.000000



Cluster Wise Area Initial
        5.500000        13.000000       0.000000
        27.599998       0.000000        0.000000
        0.000000        0.000000        0.000000



Cluster Wise Base Area Sorted :
        13.000000       5.500000        0.000000
        27.599998       0.000000        0.000000
        0.000000        0.000000        0.000000
No Further Boxes Can Be Loaded in Container-1
Total Base Area Covered of Container-1 is 18.500000
Total Base Area Covered of Container-2 is 27.599998

                        Hence The Total Number Of Containers Needed For Transportation is 2
Process returned 0 (0x0)   execution time : 38.220 s
Press any key to continue.
```

TEST CASE – 2



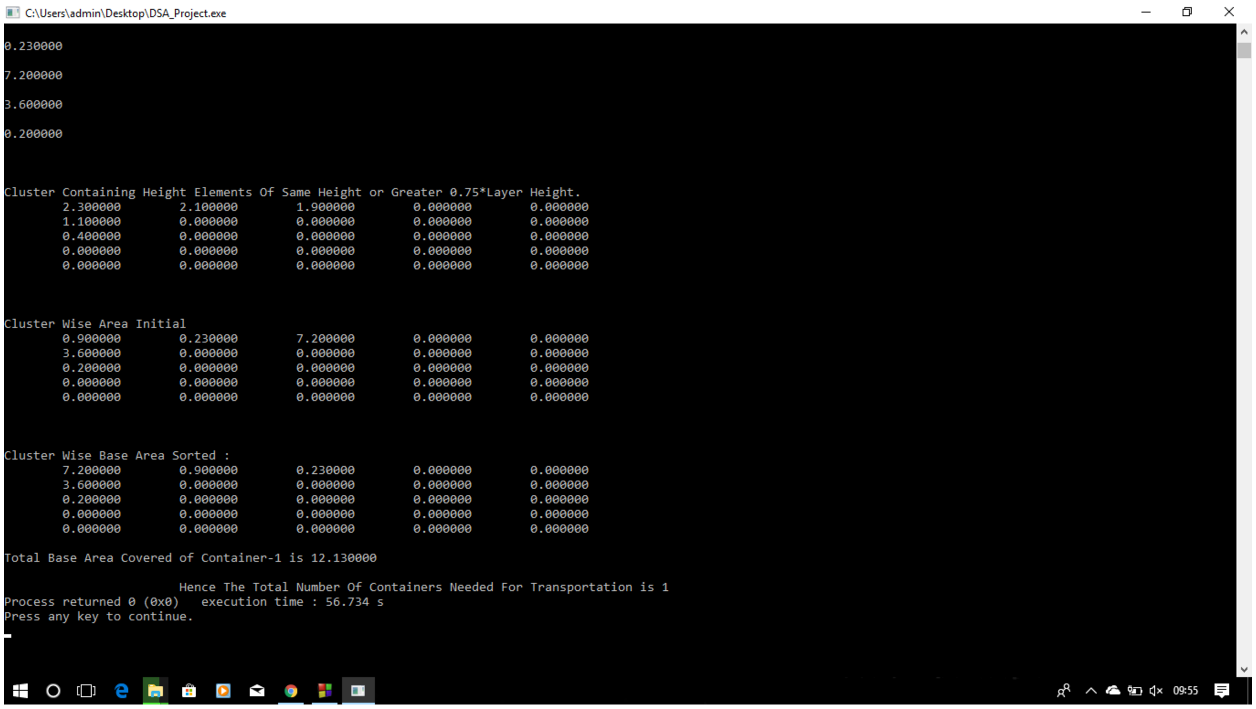
C:\Users\admin\Desktop\DSA_Project.exe

```
                                Welcome To ABS PACKERS AND MOVERS.

                                Respecting Fragility since 1999.

Please Read the Details Before Proceeding Further.

Maximum Permissible Dimension Are As Follows:
         WIDTH - 2.3m.
         DEPTH - 12m
         HEIGHT - 3m.

Please Enter the number of Boxes to be Transported : 5

Enter the Width in metre of Box No-1 : 0.1

Enter the Depth in metre of Box No-1  : 2.3

Enter the Height in metre of Box No-1 : 2.1

                                        Dimensions Of The Box Have Been Stored.

Enter the Width in metre of Box No-2 : 0.9

Enter the Depth in metre of Box No-2  : 4

Enter the Height in metre of Box No-2 : 1.1

                                        Dimensions Of The Box Have Been Stored.

Enter the Width in metre of Box No-3 : 0.9

Enter the Depth in metre of Box No-3  : 1

Enter the Height in metre of Box No-3 : 2.3

                                        Dimensions Of The Box Have Been Stored.

Enter the Width in metre of Box No-4 : 1.2

Enter the Depth in metre of Box No-4  : 6

Enter the Height in metre of Box No-4 : 1.9

                                        Dimensions Of The Box Have Been Stored.
```



C:\Users\admin\Desktop\DSA_Project.exe

```
Enter the Width in metre of Box No-5 : 0.1

Enter the Depth in metre of Box No-5  : 2

Enter the Height in metre of Box No-5 : 0.4

                                        Dimensions Of The Box Have Been Stored.


Base Area Of Each Box:

0.900000

0.230000

7.200000

3.600000

0.200000


Cluster Containing Height Elements Of Same Height or Greater 0.75*Layer Height.
        2.300000        2.100000        1.900000        0.000000        0.000000
        1.100000        0.000000        0.000000        0.000000        0.000000
        0.400000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000


Cluster Wise Area Initial
        0.900000        0.230000        7.200000        0.000000        0.000000
        3.600000        0.000000        0.000000        0.000000        0.000000
        0.200000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000


Cluster Wise Base Area Sorted :
```



C:\Users\admin\Desktop\DSA_Project.exe

```
0.230000

7.200000

3.600000

0.200000



Cluster Containing Height Elements Of Same Height or Greater 0.75*Layer Height.
        2.300000        2.100000        1.900000        0.000000        0.000000
        1.100000        0.000000        0.000000        0.000000        0.000000
        0.400000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000



Cluster Wise Area Initial
        0.900000        0.230000        7.200000        0.000000        0.000000
        3.600000        0.000000        0.000000        0.000000        0.000000
        0.200000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000



Cluster Wise Base Area Sorted :
        7.200000        0.900000        0.230000        0.000000        0.000000
        3.600000        0.000000        0.000000        0.000000        0.000000
        0.200000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000
        0.000000        0.000000        0.000000        0.000000        0.000000

Total Base Area Covered of Container-1 is 12.130000

                        Hence The Total Number Of Containers Needed For Transportation is 1
Process returned 0 (0x0)   execution time : 56.734 s
Press any key to continue.
```

ANALYSIS:

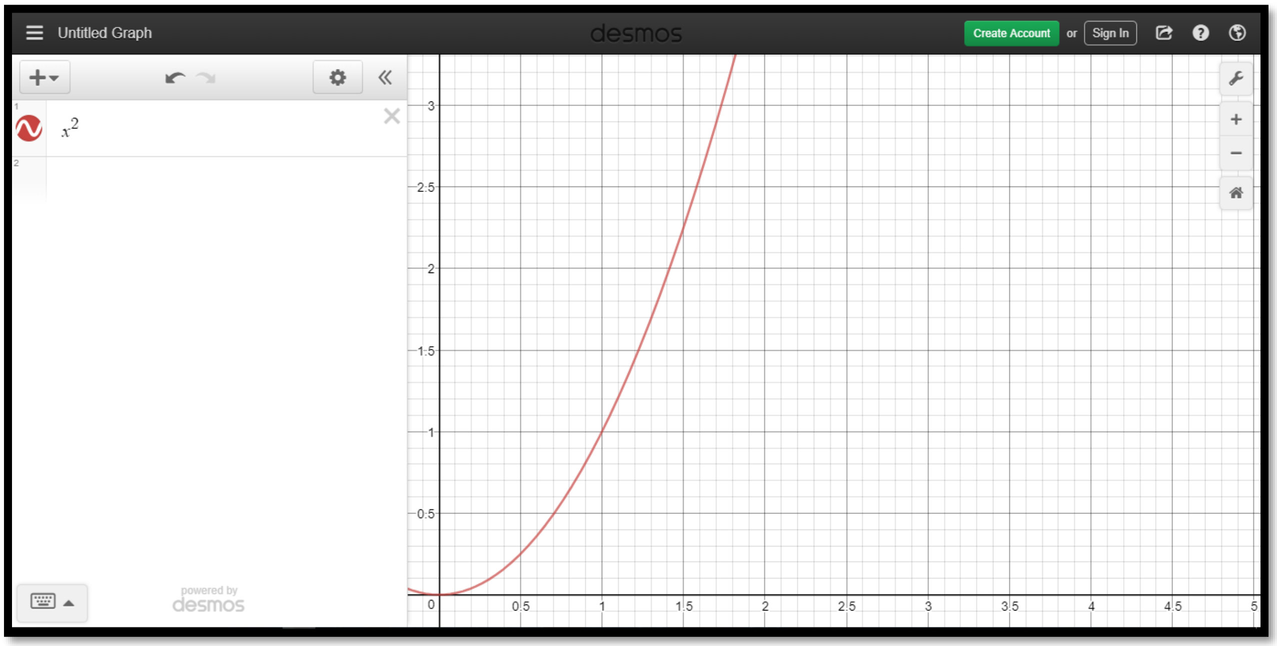
In our project, we used 3-Dimensional Bin Packing algorithm to solve one of the most common problem encountered during the shifting of homes i.e. the minimization of trucks required for moving the stuff.

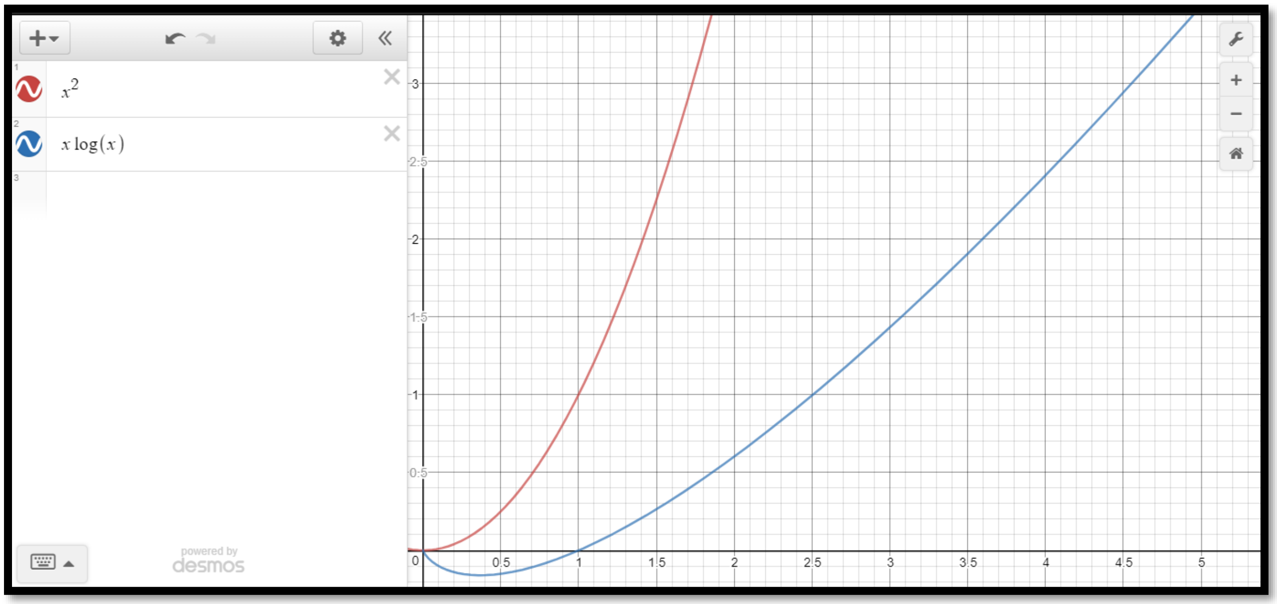
This minimization is achieved by taking into consideration all three dimensions i.e., length, breadth and the height of the container as well as the boxes to be packed. The process starts by sorting the boxes height-wise in descending order and distributing them in clusters of similar height computer by a constant factor. The boxes in the respective clusters are then again sorted out on the basis of base area.

Then the boxes are put in the trucks by maximizing the base area of the truck on the basis of clusters. The number of the containers (trucks) required is the minimum number of trucks required for shifting the boxes.

The algorithm used is a Heuristic algorithm, which means that the solution obtained by this algorithm may not be an accurate solution but the best possible approximate solution, which is a requirement to our situation as there is not a single possible solution of shifting the boxes but the way we used is an approximate but one of the best possible solution.

GRAPHS:





CONCLUSION:

In conclusion of our project, we would like to mention that our hypothesis was based on the fact that the maximum volume of a container can be occupied if the items are first sorted on the basis of one dimension and then on the basis of other two (combined). We also understood that the time complexity of our algorithm was n^2. We would like to mention that or hypothesis came out to be true as we successfully could manage to minimize the number of trucks used for the shifting process by using our algorithm.

THANK YOU

GROUP MEMBERS-

1) Rajat Rathi (17BCE0900)
2) Patel Yash Vijaykumar (17BCE0335)
3) Shubham Gupta (17BCE0199)
4) Anuraj Srivastava (17BCE2006)