****

** Global Academy of Technology**

**Department Of Computer Science and Engineering**

**Report**

**on**

**OPERATIONS RESEARCH ACTIVITY**

**VI Semester**

**Academic Year: 2018-2019**

**Group No.: 1**

**Title: North-West Corner Method In C-Programming**

|  |  |  |
| --- | --- | --- |
| **USN** | **Name** | **Signature** |
| **1GA16CS018** | **AMRIT RAJ** |  |
| **1GA16CS033** | **ASHUTOSH RANJAN** |  |
| **1GA16CS034** | **AYUSH DHAR** |  |
| **1GA16CS045** | **DHAIRYA ANAND** |  |
| **1GA16CS114** | **RAVI KANT** |  |
| **1GA16CS191** | **VISHAL KUMAR** |  |

**Guide:**

**VANISHREE M L**

Assistant Professor Dept of CSE

**Objective of the Project**

The transportation model is a special case of linear programming problems in which the objective is to minimize the total cost of transporting goods from the various supply origins to the different demand destinations. The model is often classified as a linear programming problem because the relationship between the variables transportation costs and the number of units shipped is assumed to be linear. There are important decision rules that must be applied to the transportation problem.

One method that can be used to obtain the initial basic feasible solution is the **Northwest Corner** **Method.** The advantage of the northwest corner rule is that it allows us to find an initial feasible solution to the transportation problem. It emphasizes finding an initial solution that satisfies all constraints without regard to the relative shipping costs of those orders.

**System Requirement Specification**

**Software Requirements Specification**

* Language used: C-Programming
* IDE/Compiler used: Dev C++
* OS used: Windows

**Hardware Requirements Specification**

* Processor : 2.5 GHz
* System : Intel i5
* Hard Disk : 1024 GB
* Monitor : 15 VGA color
* Mouse : Logitech
* Keyboard : 102 keys enhanced

**Source Code**

#include <stdio.h>

void main()

{

int c[20][20],i,j,m,n,dem[20],sup[20],sum=0;

printf("\*\*\*\*\*NORTH WEST CORNER METHOD\*\*\*\*\* \n");

printf("\nEnter the row:\n");

scanf("%d",&m);

printf("\nEnter the column:\n");

scanf("%d",&n);

printf("\nEnter the cost:\n");

for(i=0;i<m;i++){

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

scanf("%d",&c[i][j]);

}

printf("\nEnter the demand:\n");

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

scanf("%d",&dem[i]);

printf("\nEnter the supply:\n");

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

scanf("%d",&sup[i]);

printf("\nMatrix:\n");

for(i=0;i<m;i++){

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

printf(" %d ",c[i][j]);

printf("%d",sup[i]);

printf("\n");

}

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

printf("%d ",dem[j]);

for(i=0,j=0;(i<m&&j<n);){

if (sup[i]<dem[j]){

sum+=c[i][j]\*sup[i];

dem[j]-=sup[i];

i++;

}

else

if(sup[i]>dem[j])

{

sum+=c[i][j]\*dem[j];

sup[i]-=dem[j];

j++;

}

else

if(sup[i]==dem[j])

{

sum+=c[i][j]\*dem[j];

i++;

j++;

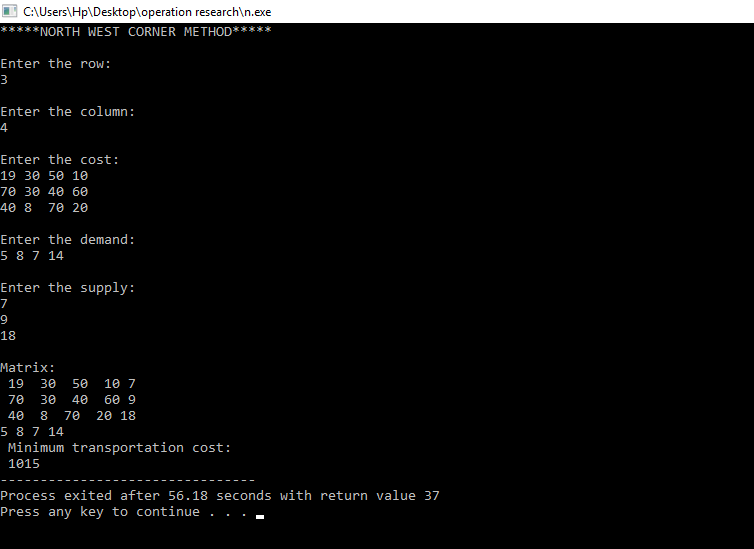
}

}

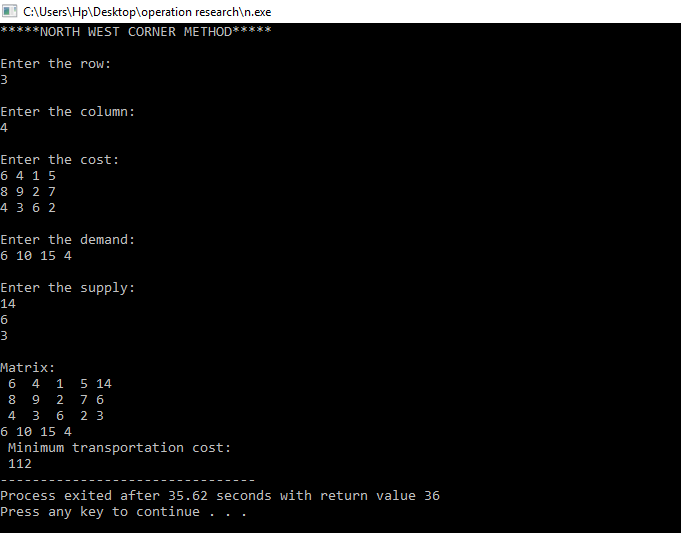
printf("\nMinimum transportation cost: \n %d",sum);

}

**Snapshots**



**Fig 1:** For Balanced Transportation Problem



**Fig 2:** For Unbalanced Transportation Problem