using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Parking\_Cars

{

class Program

{

static void Main(string[] args)

{

int[] size = new int[2];

char[,] cars;

int[,] carsDistance;

//Get dimensions from user

GetDimensions("Enter an even number of collumns up to 10: ", out size[0]);

GetDimensions("Enter an even number of rows up to 10: ", out size[1]);

Console.Clear();

GenerateCars(size, out cars);

GetCarsDistance(size, cars, out carsDistance);

DrawCarPark(size[0], size[1], cars, carsDistance);

Console.ReadLine();

ShuffleCars(size, cars, carsDistance);

Console.ReadLine();

}

static void GenerateCars(int[] size, out char[,] cars)

{

char[] building = { 'A', 'B', 'C' };

Random rand = new Random();

cars = new char[size[0], size[1]];

for (int a = 0; a < size[0]; a++)

{

for (int b = 0; b < size[1]; b++)

{

//Selects random building for car from list

cars[a, b] = building[rand.Next(0,3)];

}

}

}

static void GetDimensions(string prompt, out int dimension)

{

dimension = 11;

//Keep on getting input until dimension is even and is max 10

while (dimension > 10 || dimension % 2 == 1 || dimension < 0)

{

Console.Clear();

Console.Write(prompt);

dimension = Convert.ToInt32(Console.ReadLine());

}

}

static void GetCarsDistance(int[] size, char[,] cars, out int[,] carsDistance)

{

carsDistance = new int[size[0], size[1]];

for (int a = 0; a < size[0]; a++)

{

for (int b = 0; b < size[1]; b++)

{

carsDistance[a, b] = CalculateDistance(a,b,cars[a, b], size);

}

}

}

static int CalculateDistance(int a, int b, char c, int[] size)

{

int distance;

//Depending on what building the car is, calculate distance

switch (c)

{

case 'A':

{

if (b < Math.Floor((decimal)(size[1] - 1) / 2))

distance = a - b + (size[1] - 2) / 2;

else if (b > Math.Ceiling((decimal)(size[1] - 1) / 2))

distance = a + b - size[1] / 2;

else

distance = a;

return distance;

}

case 'B':

{

if (a < Math.Floor((decimal)(size[0] - 1) / 2))

distance = b - a + (size[0] - 2) / 2;

else if (a > Math.Ceiling((decimal)(size[0] - 1) / 2))

distance = b + a - size[0] / 2;

else

distance = b;

return distance;

}

case 'C':

{

if (b < Math.Floor((decimal)(size[1] - 1) / 2))

distance = size[0] - 1 - a - b + (size[1] - 2) / 2;

else if (b > Math.Ceiling((decimal)(size[1] - 1) / 2))

distance = size[0] - 1 - a + b - size[1] / 2;

else

distance = size[0] - 1 - a;

return distance;

}

}

return 0;

}

static void ShuffleCars(int[] size, char[,] cars, int[,] carsDistance)

{

bool changesWereMade = true;

while (changesWereMade)

{

changesWereMade = false;

//For every single car, check that by swapping with any other car it doesnt make a change, if it does, make the change

for (int a = 0; a < size[0]; a++)

{

for (int b = 0; b < size[1]; b++)

{

for (int c = a; c < size[0]; c++)

{

for (int d = b; d < size[1]; d++)

{

//If the cars are not from the same building, and swapping them changes the combined distance, Swap the cars

if (cars[a, b] != cars[c, d] && carsDistance[a, b] + carsDistance[c, d] >= CalculateDistance(c, d, cars[a, b], size) + CalculateDistance(a, b, cars[c, d], size))

{

if (carsDistance[a, b] + carsDistance[c, d] > CalculateDistance(c, d, cars[a, b], size) + CalculateDistance(a, b, cars[c, d], size))

changesWereMade = true;

char chr = cars[a, b];

//Swaps the cars

cars[a, b] = cars[c, d];

cars[c, d] = chr;

//Recalculates The Distance

carsDistance[a, b] = CalculateDistance(a, b, cars[a, b], size);

carsDistance[c, d] = CalculateDistance(c, d, cars[c, d], size);

DrawCarPark(size[0], size[1], cars, carsDistance);

}

}

}

}

}

}

}

static void DrawCarPark(int collumns, int rows, char[,] cars, int[,] carsDistance)

{

Console.Clear();

//For every row in the carpark

for (int a = 0; a < rows; a++)

{

//Draw the roof of each collumn

DrawLine(collumns, "+-----", '+');

//Draw the car's buidling

for (int b = 0; b < collumns; b++)

Console.Write("| " + cars[b, a] + " ");

Console.WriteLine('|');

//Draw the car's distance

for (int b = 0; b < collumns; b++)

Console.Write("| " + String.Format("{0:d2}", carsDistance[b, a]) + " ");

Console.WriteLine('|');

}

//Draw the "floor" of the carpark

DrawLine(collumns, "+-----", '+');

}

static void DrawLine(int collumns, string line, char finish)

{

for (int a = 0; a < collumns; a++)

{

Console.Write(line);

}

Console.WriteLine(finish);

}

}

}