```
#include <iostream>
#include<ctime>
#include<fstream>

using namespace std;
const int m=500,n=500;
int comparisonArray[m][n];
int tempX,tempY;


class Point
{
private:
        int x1,y1;

public:
        Point(){}

        Point(int x,int y)
        {
                x1=x;
                y1=y;
        }
        int getX()
        {
                return x1;
        }

        int getY()
        {

                return y1;
        }
};

bool isAttached(int x, int y){
        bool isAttached = false;
        int x_min = x - 1;
        int y_min = y - 1;
        int x_max = x + 1;
        int y_max = y + 1;
        if (x_max > m - 1) {x_max = m - 1;}
        if (y_max > m - 1) {y_max = m - 1;}
        if (x_min < 0) {
            x_min = 0;
    }
        if (y_min < 0)
        {
            y_min = 0;
    }

        for (int i = x_min; i <= x_max; i++){
                for (int j = y_min; j <= y_max; j++ ){
                        if (comparisonArray[i][j] == 1){
                                isAttached = true;
                        }
                }
        }
        return isAttached;
}


bool comparison(int row,int column)
{
        if(comparisonArray[row+1][column]==1)
        {
                return true;
        }
        else if(comparisonArray[row-1][column]==1)
        {
                return true;
        }
        else if(comparisonArray[row][column+1]==1)
        {
                return  true;
        }
        else if(comparisonArray[row][column-1]==1)
        {
                return  true;
        }
        else if(comparisonArray[row+1][column-1]==1)
        {
                return  true;
        }

        else if(comparisonArray[row-1][column+1]==1)
        {
                return  true;
        }
        else if(comparisonArray[row-1][column-1]==1)
        {
                return  true;
        }
        else if(comparisonArray[row+1][column+1]==1)
        {
                return  true;
        }
        else
        {
                return  false;

        }


};

int update(int x)
{
        int max=1;
        int min=-1;

        int randNum = rand()%(max-min + 1) + min;
        int result;
        result =randNum+x;
        if (result<0)
        {
                return 0;
        }
        else if(result>m-1)
        {
                return m-1;
        }
        else
                return result;
}



void compute(Point p)
```

```
{

        tempX=p.getX();
                tempY=p.getY();
                int flag=0;

                while(flag!=1){

                        if(isAttached(tempX,tempY)==true)
                        {
                                comparisonArray[tempX][tempY]=1;
                                flag=1;
                        }
                        else
                        {
                                tempX=update(tempX);
                                tempY=update(tempY);

                        }

                }

        }



int main()
{

        int number_of_particles=10000;
        for(int i=0;i<m;i++){
                for(int j=0;j<n;j++){
                        comparisonArray[i][j]=0;
                }
                cout<<"\n";
        }

        int centerX=(m/2);
        int centerY=(n/2);
        comparisonArray[centerX][centerY]=1;

        int random_integer = 1+rand()%m;
        srand(time(NULL));
    int random_int = rand()%m;
    cout<<random_int<<endl;


    Point p(random_int,0);
    Point q(0,random_int);
    Point r(random_int,m-1);
    Point s(m-1,random_int);




        for(int i=0;i<number_of_particles;i++){

                compute(p);
                compute(q);
                compute(r);
                compute(s);
        }

        for(int i=0;i<m;i++){
                for(int j=0;j<n;j++){
                        cout<<comparisonArray[i][j]<<"\t";
                }
                cout<<"\n";
        }

        ofstream myfile;
    myfile.open ("matrix_output.csv",fstream::out);
    for (int i = 0; i < m ; i++) {
        for (int j = 0; j < m; j++){
            myfile << comparisonArray[i][j];
            if (j < m - 1) {
                myfile << ",";
            }
        }
        myfile << endl;

    }
    myfile.close();
    return 0;


}
```