Queen

```
import java.util.Scanner;
class nq
{
   private int[] result;
   private boolean[] column;
   private boolean[] rd;
   private boolean[] ld;
   private int n;
    nq(int n)
    {
       this.n=n;
       column=new boolean[n];
       rd=new boolean[2*n-1];
       ld=new boolean[2*n-1];
       result=new int[n];
    }

    public boolean nSolve()
    {
       return branchbound(0);
    }
    private boolean branchbound(int row)
    {
       if(row==n){
       printsolution();
       return true;
}
       boolean res=false;
       for(int col=0;col<n;col++)
       {
```

```
    if(isSafe(row,col))
    {
        placequeen(row,col);
        res=branchbound(row+1)|| res;
        removequeen(row,col);
}
    }

   return res;
}
private boolean isSafe(int row,int col){
        return !column[col] &&!ld[row-col+n-1]&& !rd[row+col];
                }
private void placequeen(int row,int col)
{
     result[row]=col;
     column[col]=true;
     ld[row-col+n-1]=true;
     rd[row+col]=true;
}
private void removequeen(int row,int col)
{
     column[col]=false;
     ld[row-col+n-1]=false;
     rd[row+col]=false;
}
private void printsolution()
{
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
```

```
            {
               if(result[i]==j)
               {
                  System.out.print("Q");
            }
               else{
                  System.out.print(".");
               }


         }
            System.out.println();
         }
         System.out.println();
      }
      public static void main(String args[])
      {
         int n=4;
         nq ns=new nq(n);
         if(!ns.nSolve())
            System.out.println("Cannot be placed");
      }




   }
```