Mohini Akhare

```
package codatation;
import java.util.*;
public class Codation_AssignmentGame_of_life
{

          public final static int Dead=0; //for dead we puted 0
and final to make it constant
          public final static int Alive=1; //for alive we puted
1 and final to make it constant
           int [][] grid; //array of grid
           int rows;
           int columns;
           int k=1;
           Scanner sc=new Scanner(System.in);//scanner class to
take input from user

           public void set(int [][] grid,int ROWS,int COLUMNS)
           {
             this.rows=ROWS;
             this.columns=COLUMNS;
             this.grid=new int[rows][columns];
             for(int i=0;i<ROWS;i++)
               for(int j=0;j<COLUMNS;j++)
               this.grid[i][j]=grid[i][j];
           }

           public void get()
           {
             for(int i=0;i<rows;i++)
             {
               for(int j=0;j<columns;j++)
               System.out.print(grid[i][j]);

                 System.out.println();
             }
             System.out.println();
           }

             public void gameOfLife()
              {
                    for (int i = 0; i < rows; i++)
                   {
                              for (int j = 0; j < columns; j++)
```

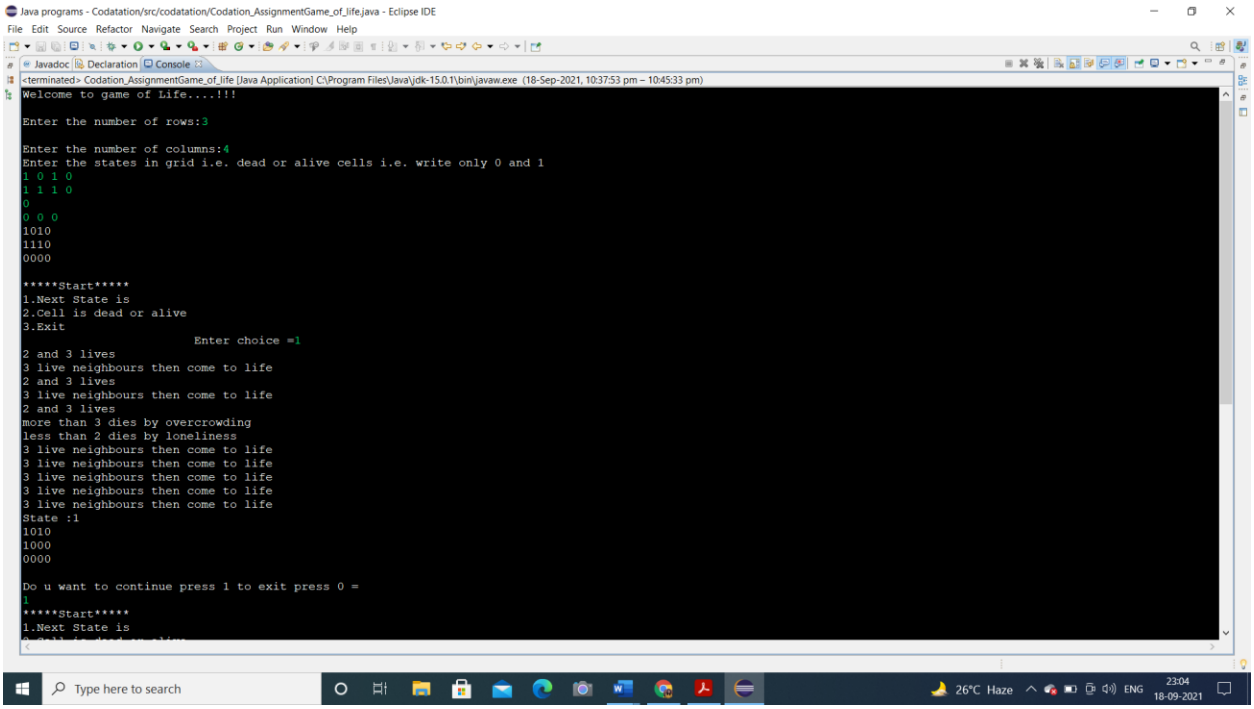
```
                    {
                        int Cell = grid[i][j];
                       n(i,j,Cell);

                    }
                }
            }

        public void n(int i,int j,int Cell)
        {
                int count=0;
                int a = i - 1;
                int b = i + 1;
                int c = j - 1;
                int d = j + 1;

                   if (a >= 0 && grid[a][j]==1)
                     count++;

                   if(b <rows && grid[b][j]==1)

                  count++;
                if(c >=0 && grid[i][c]==1)
                {
                  count++;

                }

                if(d <columns && grid[i][d]==1)
                {
                  count++;

                }

                //lower right side diagonal
                if(j>=0 && j<columns-1 && i>=0 && i<rows-
1)

                {
                   if(grid[b][d]==1)
                     count++;

                    // grid[b][d]=5;
                }

                //upper left side diagonal
                if(i>0 && i<rows && j>0 && j<columns)
                {
```

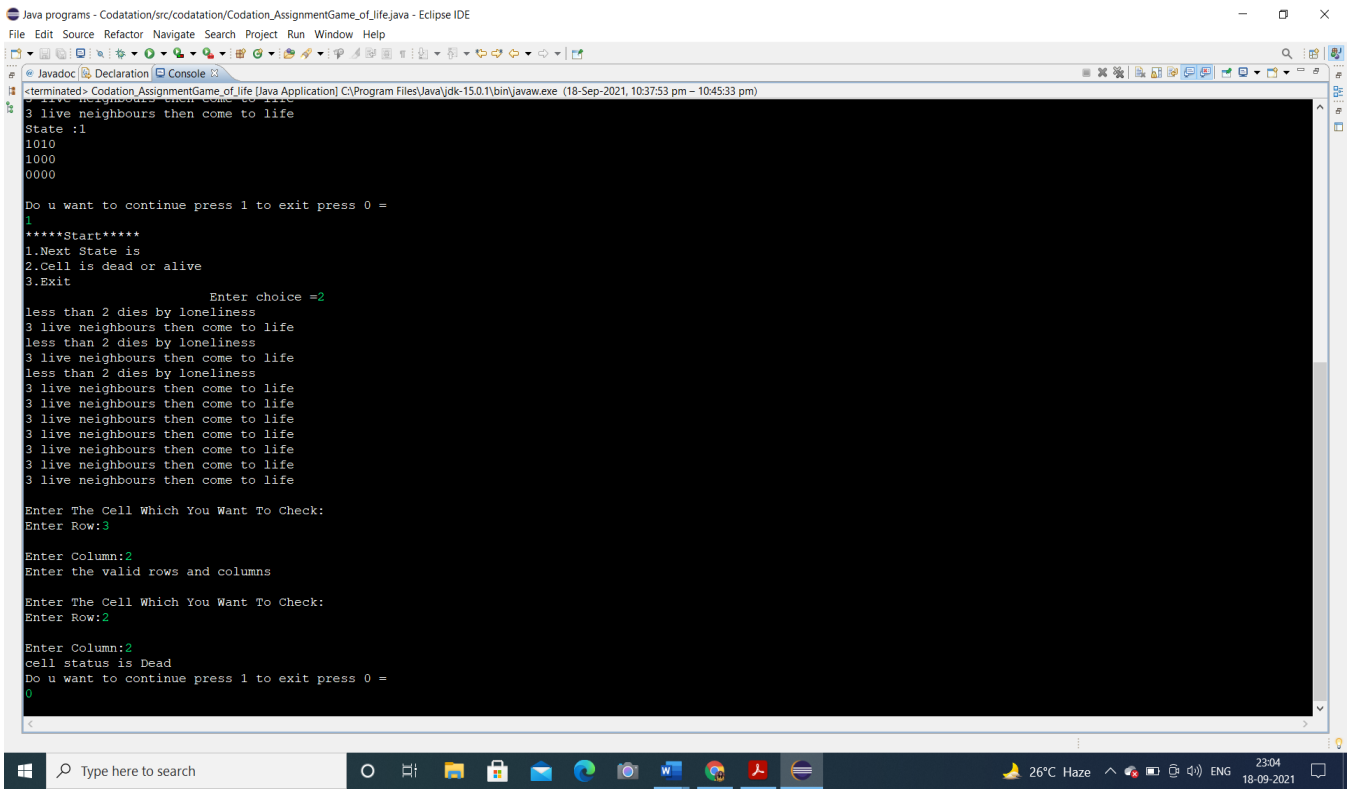
```
                    if(grid[a][c]==1)
                      count++;

                      //grid[a][c]=5;
                    }

                    //lower left side diagonal
                    if(j>0 && j<columns && i>=0 && i<rows-1)
                    {
                      if(grid[b][c]==1)
                        count++;

                        //grid[b][c]=5;
                    }

                    //upper side right diagonal
                    if(i>0 && i<rows && j>=0 && j<columns-1)
                    {
                      if(grid[a][d]==1)
                        count++;
                    }

                    if(Cell==1)
                    {
                      if(count<2)
                      {
                    grid[i][j]=0;
                    System.out.println("less than 2 dies by
loneliness");
                      }

                      if(count>3)
                      {
                    grid[i][j]=0;
                    System.out.println("more than 3 dies by
overcrowding");
                      }

                    if(count==2|| count==3)
                      {
                        grid[i][j]=grid[i][j];
                        System.out.println("2 and 3 lives ");
                      }
                    }

                    if(Cell==0)
                    {
```

```
                        if(count==3)
                        grid[i][j]=1;
                        System.out.println("3 live neighbours
then come to life");
                    }

          }

            public void printGrid()
            {
                  if(k<rows)
                  System.out.println("State :"+k++);

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

                  for(int j=0;j<columns;j++)
                    System.out.print(grid[i][j]);

                  System.out.println();
                }

              }


            public void PrintCell()
             {
                 int counte = 0;
               System.out.print("\nEnter The Cell Which You Want
To Check: ");
                System.out.print("\nEnter Row:");
                int r=sc.nextInt();
                System.out.print("\nEnter Column:");
                int c=sc.nextInt();
                if(r<rows && c<columns)
                {
                for(int i=0;i<rows;i++)
                {
                  for(int j=0;j<columns;j++)
                    if(grid[r][c]==0)
                      counte=0;
                    else
                      counte=1;
                }
                  if(counte==0)
                    System.out.print("cell status is Dead");
                  if(counte==1)
```

```
                    System.out.print("cell status is Alive");

                }
                else
                {
                  System.out.println("Enter the valid rows and
columns");
                  PrintCell();
                }
              }
             public void getStates()
              {
                int n;

               do {

                 System.out.println("*****Start*****");
                  System.out.println("1.Next State is");
                  System.out.println("2.Cell is dead or alive");
                  System.out.println("3.Exit");
                  System.out.print("\t\t\tEnter choice =");
                  int ch=sc.nextInt();

                  switch(ch)
                  {
                  case 1:
                   {
                   gameOfLife();
                   printGrid();
                   break;
                   }

                  case 2:
                   {
                   gameOfLife();
                   PrintCell();
                   break;
                   }

                  case 3:
                   {
                   System.exit(0);
                   break;
                   }
                  }
                  System.out.println("\t\t\t\nDo u want to
continue press 1 to exit press 0 = ");
```

```
                n=sc.nextInt();
               }while(n!=0);

            }

          public static void main(String[] args)
          {
            Scanner sc2=new Scanner(System.in);
            System.out.println("Welcome to game of
Life....!!!");
            System.out.print("\nEnter the number of rows:");
            int rows1=sc2.nextInt();

            System.out.print("\nEnter the number of
columns:");
            int columns1=sc2.nextInt();

            int [][] grid=new int[rows1][columns1];

            System.out.println("Enter the states in grid i.e.
dead or alive cells i.e. write only 0 and 1");

            for(int i=0;i<rows1;i++)
              for(int j=0;j<columns1;j++)
                grid[i][j]=sc2.nextInt();

            Codation_AssignmentGame_of_life  gm=new
Codation_AssignmentGame_of_life();
            gm.set(grid, rows1, columns1);
            gm.get();
            gm.getStates();

          }

}
```


Java programs - Codatation/src/codatation/Codation_AssignmentGame_of_life.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Javadoc Declaration Console
<terminated> Codation_AssignmentGame_of_life [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (18-Sep-2021, 10:37:53 pm – 10:45:33 pm)
Welcome to game of Life....!!!
Enter the number of rows:3
Enter the number of columns:4
Enter the states in grid i.e. dead or alive cells i.e. write only 0 and 1
1 0 1 0
1 1 1 0
0
0 0 0
1010
1110
0000
*****Start*****
1.Next State is
2.Cell is dead or alive
3.Exit
Enter choice =1
2 and 3 lives
3 live neighbours then come to life
2 and 3 lives
3 live neighbours then come to life
2 and 3 lives
more than 3 dies by overcrowding
less than 2 dies by loneliness
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
State :1
1010
1000
0000
Do u want to continue press 1 to exit press 0 =
1
*****Start*****
1.Next State is
Type here to search
26°C Haze
ENG
23:04
18-09-2021


Java programs - Codatation/src/codatation/Codation_AssignmentGame_of_life.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Javadoc Declaration Console
<terminated> Codation_AssignmentGame_of_life [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (18-Sep-2021, 10:37:53 pm – 10:45:33 pm)
3 live neighbours then come to life
State :1
1010
1000
0000
Do u want to continue press 1 to exit press 0 =
1
*****Start*****
1.Next State is
2.Cell is dead or alive
3.Exit
Enter choice =2
less than 2 dies by loneliness
3 live neighbours then come to life
less than 2 dies by loneliness
3 live neighbours then come to life
less than 2 dies by loneliness
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
3 live neighbours then come to life
Enter The Cell Which You Want To Check:
Enter Row:3
Enter Column:2
Enter the valid rows and columns
Enter The Cell Which You Want To Check:
Enter Row:2
Enter Column:2
cell status is Dead
Do u want to continue press 1 to exit press 0 =
0
Type here to search
26°C Haze
ENG
23:04
18-09-2021