B.M.S College of Engineering

P.O. Box No.: 1908 Bull Temple Road,

Bangalore-560 019

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Course – Analysis and Design of Algorithms

Course Code – 15IS4DCADA

AY 2018-19

Report on ADA Self-Study Assignment

TETRIS

Submitted to – Dr. Shambhavi B.R. and Dr.H.S.Guruprasad

Submitted by -

H.Anarghya 1BM17IS028

Irene Komal 1BM17IS033

Lalitashree R Hegde 1BM17IS039

Pannaga Sharma M L 1BM17IS053

Problem Statement:

Tetris:

Tetris is played on a 10 by 20 grid called the Matrix. Shapes called Tetrominos (7 types) are to be generated randomly. Player places them on the matrix. The primary way to score points in Tetris is to clear lines by manipulating the pieces so that they fill horizontal row within the Matrix.

Rules:

1. A tetromino cannot be placed outside the boundaries of the grid.
2. Rotations are possible for the shapes.
3. The shapes can be moved even after they land at the bottom briefly, but they will lock down as we stop the movements. At that point the next tetromino will begin to fall.
4. Scoring is mainly by clearing the horizontal rows once they are complete. It is also possible to score by placing a shape on the grid.

Data structure and Algorithm chosen:

Data structure used :

An array of structure with variables, i.e.,an array to store the shapes of the tetrominoes and an integer variable to store points assigned to each variable.

A 3 dimensional array of dimension 4x4x4 in which the first index of the array holds the possible rotations the shapes could have and the rest hold the shapes.

Array and structures are the most suitable data structures for the given set of problem because the memory to be utilized is already fixed as we already have the shapes and the points assigned to them beforehand. Hence static memory allocation is chosen.

Algorithm used:

Brute force, which is the basic algorithm ,is used as our design tool for implementing the game. We make use of this algorithm to check the boundary conditions and collision detection.

Brute force is a straightforward algorithm and since the basic operation in our program is to check for collisions and boundaries, using this algorithm makes it more understandable and easier to implement.

Source Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<signal.h>
#include<unistd.h>
char Matrix[21][12],fMatrix[21][12];
int score1=0;

typedef struct sh
{
   int block[4][4][4];
   int pts;
}shape;

shape Shape[7]= {

{{0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},25},

{{0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},50},

{{0,0,0,0,0,0,1,1,0,1,1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},50},

{{0,0,0,0,0,1,1,0,0,0,1,1,0,0,0,0,0,0,1,0,0,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},50},

{{0,0,0,0,0,1,1,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,
0,0,1,1,0,0,0,1,0,0,0,1,0,0,0,0,0},75},
```

```
{{0,0,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,1,0,0,1,1,0,0,0,0,0},75},


{{0,0,0,0,0,1,1,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,1,0,0,1,0,0,0,0,0,0,0,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,0,1,0,0,0,0,0},75}

};


void exit_handler()

{

    system("clear");

    printf("\nGAME OVER\nYOUR SCORE IS: %d\n",score1);

    exit(0);

}


void score()

{

    int i,j,k,l;

    for(i=19;i>=0;i--)

    {

        for(j=1;j<11;j++)

        {

            if(Matrix[i][j]=='\0')

                break;

        }

        if(j==11)

        {

        if(i==0)

        {

            printf("\nYOUR SCORE: %d \nGAME OVER",score1);

            exit(0);

        }
```

```
                score1+=100;
                for(k=i;k>0;k--)
                        {
                        for(l=1;l<11;l++)
                                fMatrix[k][l]=fMatrix[k-1][l];
                }
                }
        }
        for(j=1;j<11;j++)
        {
                if(Matrix[0][j]==1)
                {
                        printf("\nGAME OVER\nYOUR SCORE IS: %d\n",score1);
                        exit(0);
                }
        }
}

void initfMatrix()
{
        int i,j;
        char ch;
        for (i = 0; i < 21; i++)
                for (j = 0; j < 12; j++)
                        fMatrix[i][j] = (j == 0 || i == 20 || j == 11) ? '#' : '\0';
        system("clear");
        printf("\n\n\t\t\tWELCOME TO TETRIS\n");
        printf("\n\t\tInstructions:\n\t\t1.Enter 1 to rotate the piece once every time,0 to stop rotating.\n\t\t2.Enter L to move the piece left and enter R to move the piece right.\n\t\t\t Enter N to stop and move the piece downwards.\n\t\t3.Press Ctrl+C to quit.\n");
        printf("\n\t\tPress Enter key to continue..\n");
```

```
      ch=getchar();
}

void print_on_fboard(int shapeno,int rotno,int x,int y)
{
   int i,j;
   for(i=0;i<4;i++)
   {
      for(j=0;j<4;j++)
      {
         if(Shape[shapeno].block[rotno][i][j]==1)
            fMatrix[x+i][y+j]=(char)Shape[shapeno].block[rotno][i][j];
      }
   }
     score1+=Shape[shapeno].pts;
}

void displayBoard()
{
   int i,j;
   system("clear");
   for(i=0;i<21;i++)
   {
      printf("\n");
            for(j=0;j<12;j++)
                  printf("%c",(Matrix[i][j]==(char)1) ? 'X' :(Matrix[i][j]=='\0') ? '
':Matrix[i][j]);
   }
   printf("\n");
   printf("\nYOUR SCORE: %d\n",score1);
}
```

```
int checkposition(int shapeno,int rotno,int x,int y)
{
      int i,j,flag=0;
      for(i=0;i<4;i++)
   {
      for(j=0;j<4;j++)
       {
              if(Shape[shapeno].block[rotno][i][j]==1)
          {
              if(x+i==20||y+j==0||y+j==11)
                     return 0;
              else if(fMatrix[x+i][y+j]==1)
              return 2;
          }
       }
    }
    return 1;
}

void print_shape_on_board(int shapeno,int rotno,int x,int y)
{
      int i,j,flag=0;
      for (i = 0; i < 21; i++)
      for (j = 0; j < 12; j++)
              Matrix[i][j] =fMatrix[i][j];
   for(i=0;i<4;i++)
   {
      for(j=0;j<4;j++)
      {
```

```
            if(Shape[shapeno].block[rotno][i][j]==1)
                Matrix[x+i][y+j]=(char)Shape[shapeno].block[rotno][i][j];
        }
    }
}

int rotate_shape(int shno,int rotno)
{
    switch(shno)
    {
        case 0: printf("No rotations possible\n");
                break;
        case 1:
        case 2:
        case 3: rotno=((rotno)+1)%2;
                break;
        case 4:
        case 5:
        case 6: rotno=((rotno)+1)%4;
                break;
    }
    return rotno;
}

int main()
{
    int i,j,shno=-1,rno,py,px,flag=0,c;
    char ch;
    signal(SIGINT,exit_handler);
    initfMatrix();
```

```
while(1)
{
        srand(time(0));
        shno=rand()%7;
        rno=0;
        px=0;
        py=4;
        if(checkposition(shno,rno,px,py)==2)
        {
                system("clear");
                printf("\nGAME OVER\nYOUR SCORE IS: %d\n",score1);
                exit(0);
        }
        print_shape_on_board(shno,rno,px,py);
        displayBoard();
        printf("\nDo you want to rotate it?(1/0): ");
        scanf("%d",&c);
        while(c!=0)
        {
           rno=rotate_shape(shno,rno);
           print_shape_on_board(shno,rno,px,py);
           displayBoard();
           printf("\nDo you want to rotate it?(1/0):");
           scanf("%d",&c);
        }
        print_shape_on_board(shno,rno,px,py);
        displayBoard();
        printf("\nMove left(L) or right(R) (Press N for no movement): ");
        scanf("%c",&ch);
        while(ch!='N')
```

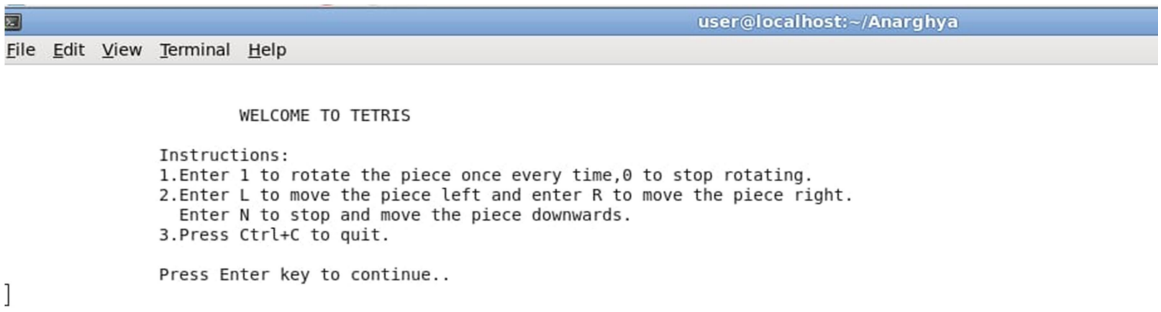
```
{
        px++;
        if(checkposition(shno,rno,px,py)==0)
        {
                px--;
        if(px==20)
                flag=1;
        }
        else if(checkposition(shno,rno,px,py)==2)
   {
        px--;
        flag=1;
   }
        else
        {
        print_shape_on_board(shno,rno,px,py);
        displayBoard();
        }
        flag=0;
   switch(ch)
   {
      case 'R':
      {
        py++;
        if(checkposition(shno,rno,px,py)==0)
                py--;
        else if(checkposition(shno,rno,px,py)==2)
        {
                py--;
           flag=1;
```

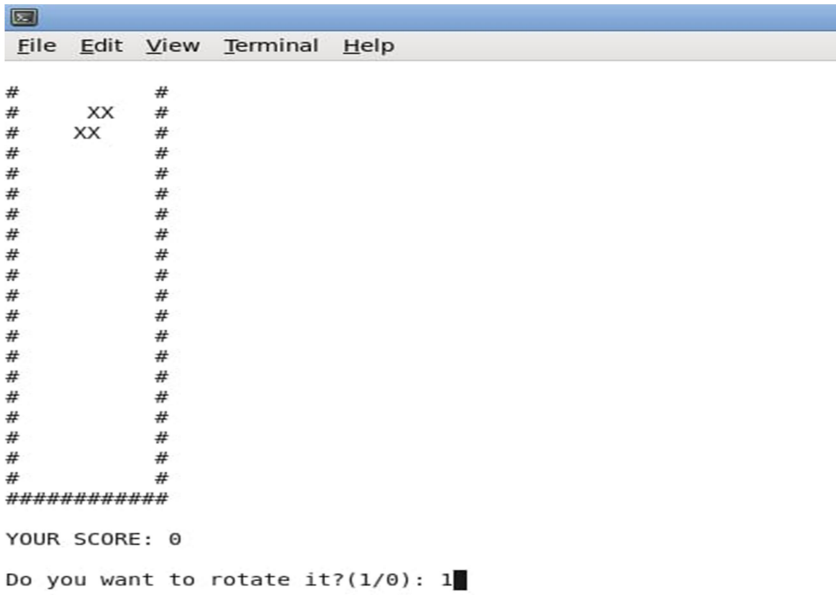
```
        }
        else
        {
                print_shape_on_board(shno,rno,px,py);
           displayBoard();
        }
        break;
    }
    case 'L':
    {
        py--;
        if(checkposition(shno,rno,px,py)==0)
                py++;
        else if(checkposition(shno,rno,px,py)==2)
        {
                py++;
           flag=1;
        }
        else
        {
                print_shape_on_board(shno,rno,px,py);
           displayBoard();
        }
        break;
    }
        default: break;
}
if(flag==1)
                break;
        printf("\n>>>");
```

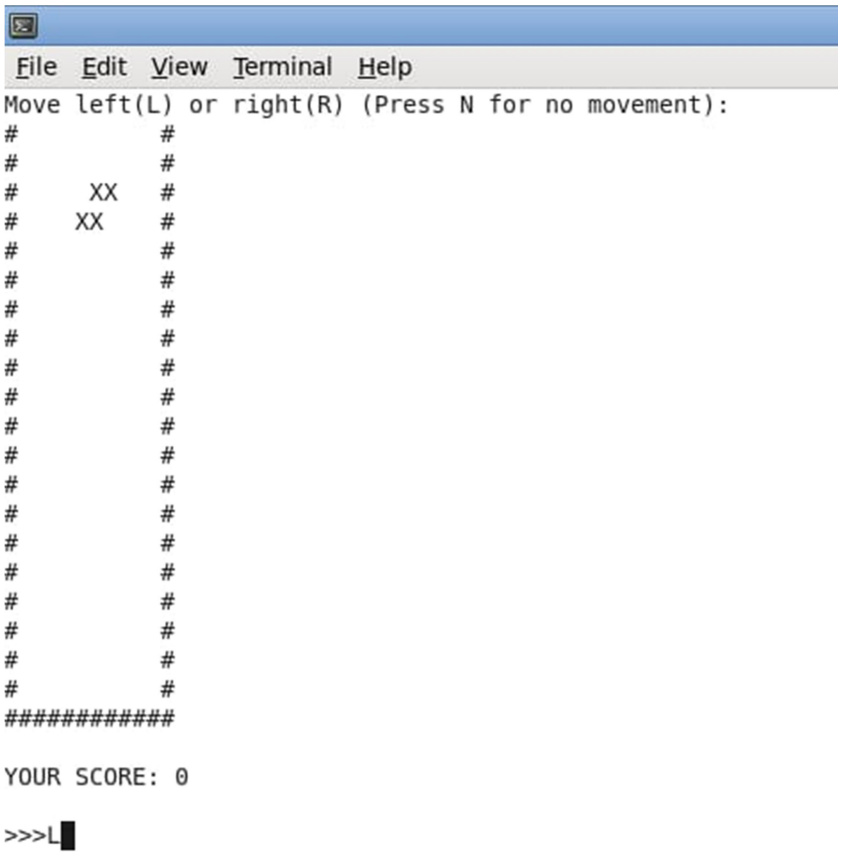
```
    scanf("%c",&ch);
}
while(1)
{
        px++;
   if(checkposition(shno,rno,px,py)==0)
        {
            px--;
        break;
        }
        else if(checkposition(shno,rno,px,py)==2)
   {
        px--;
        break;
   }
        else
        {
        print_shape_on_board(shno,rno,px,py);
        displayBoard();
        }
}
print_on_fboard(shno,rno,px,py);
displayBoard();
score();
for (i = 0; i < 21; i++)
{
        printf("\n");
   for (j = 0; j < 12; j++)
        printf("%c",fMatrix[i][j]);
 }
```

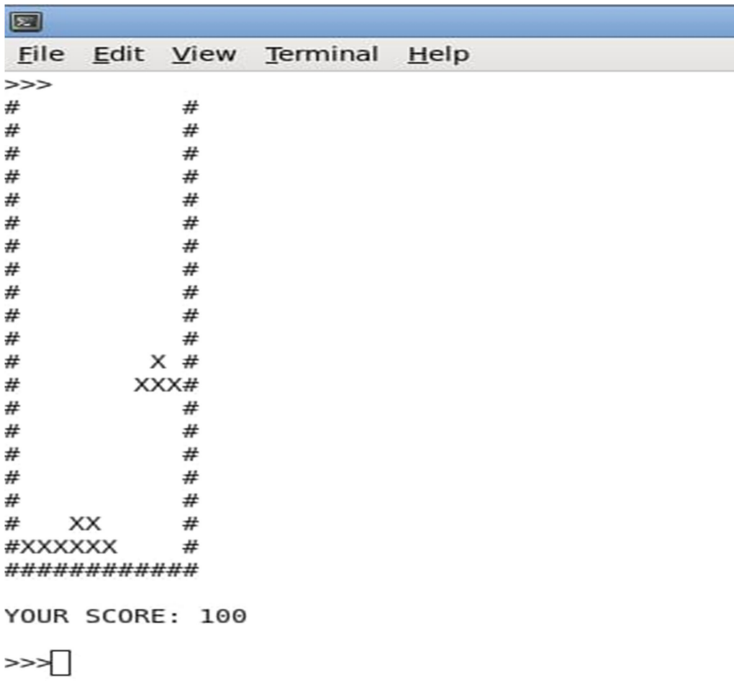
```
    }
  return 0;
}
```

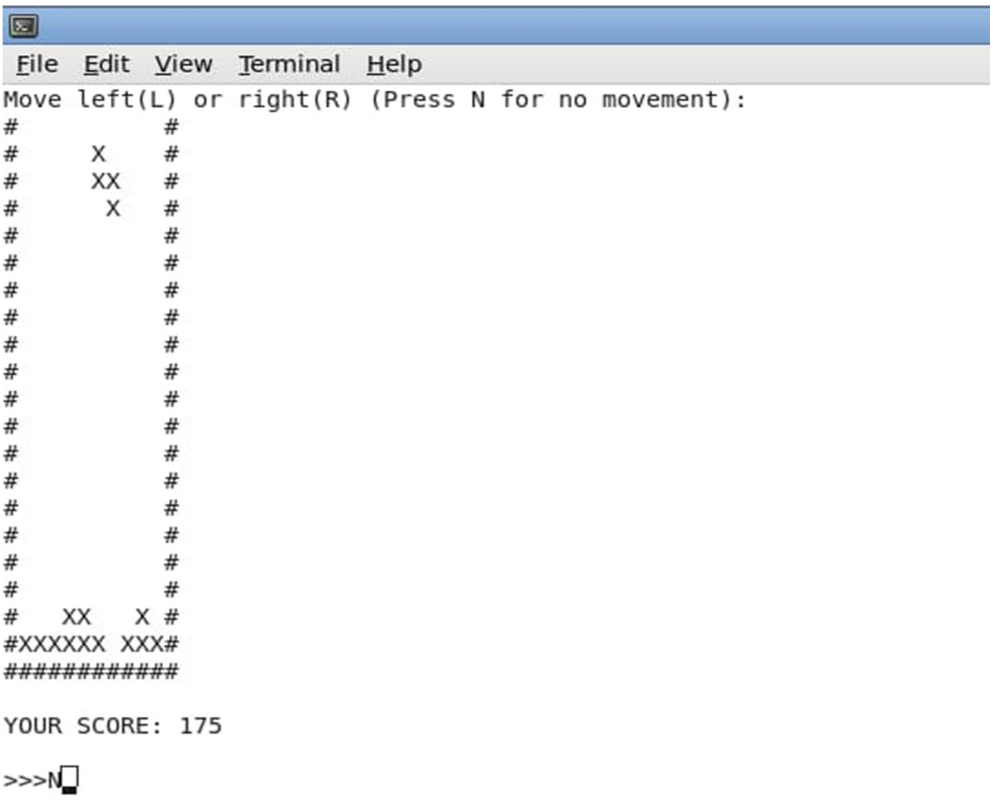
Output:

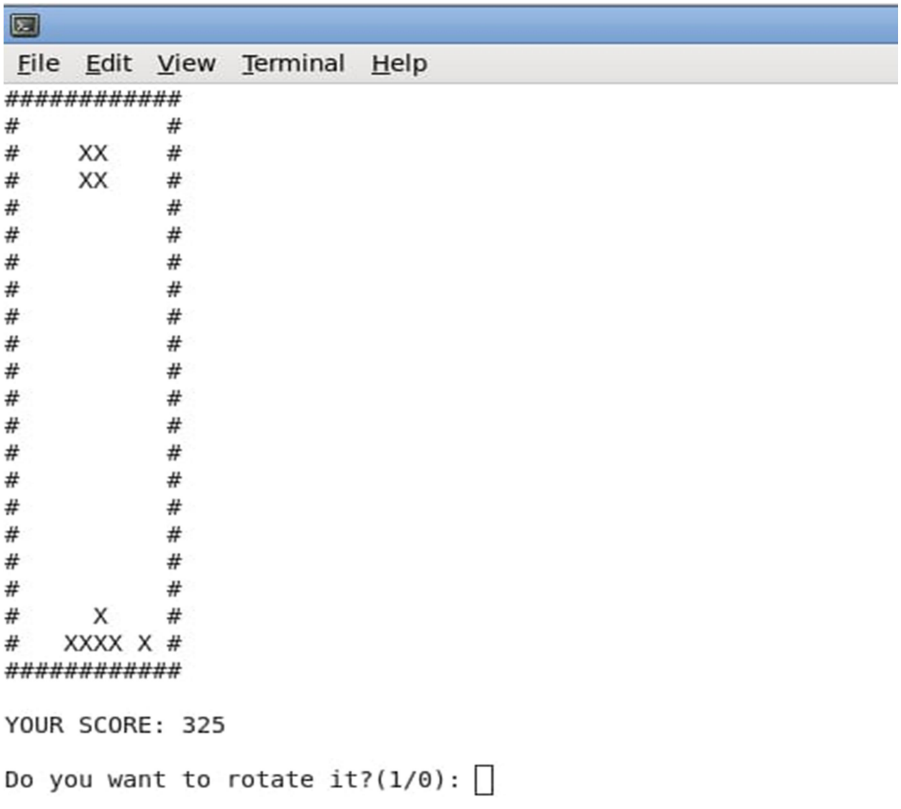


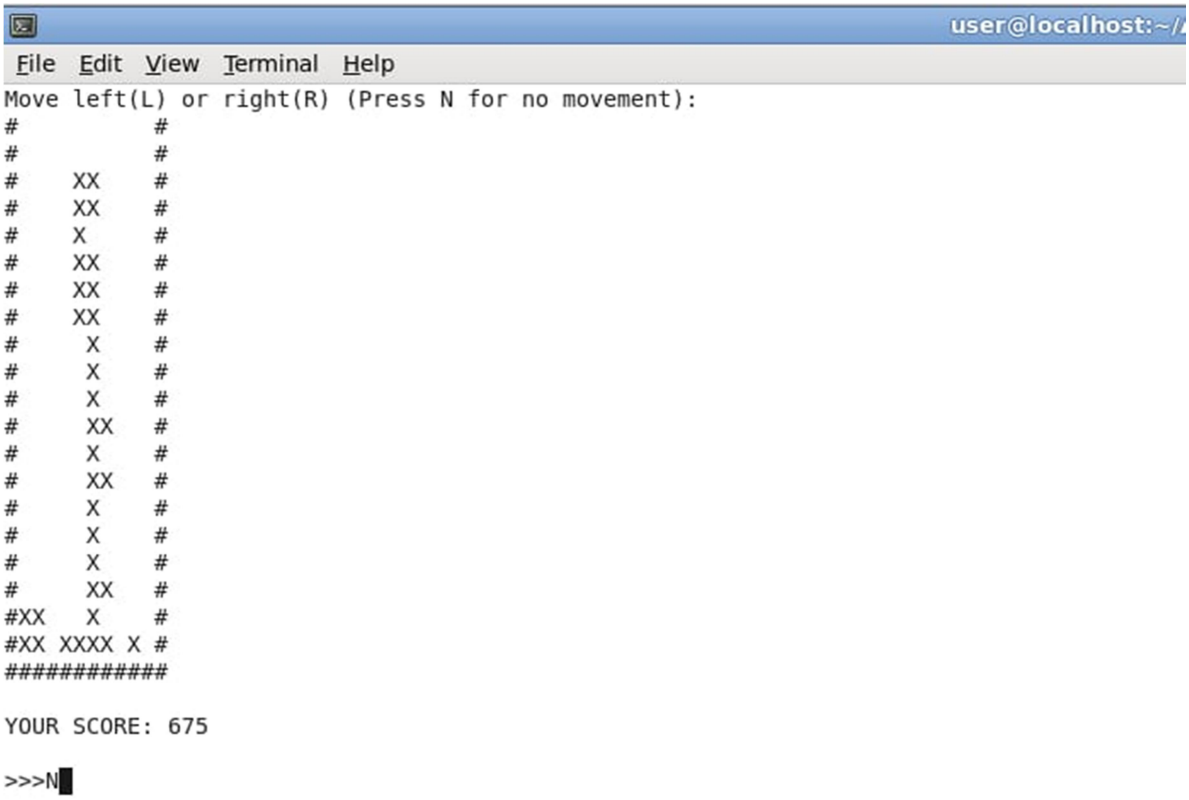


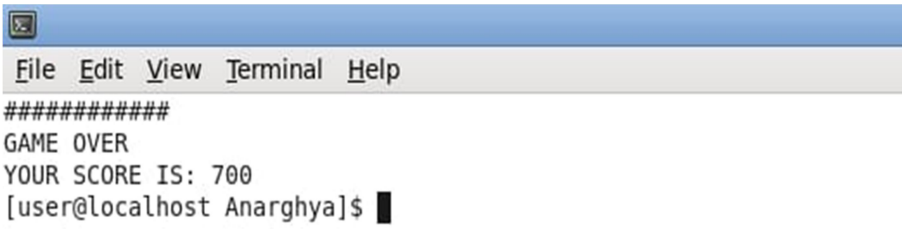

File Edit View Terminal Help
Move left(L) or right(R) (Press N for no movement):
YOUR SCORE: 0
>>>L


File Edit View Terminal Help
>>>
YOUR SCORE: 100
>>>


File Edit View Terminal Help
Move left(L) or right(R) (Press N for no movement):
#   #
X
XX
X
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
XX   X
#XXXXXX XXX#
############
YOUR SCORE: 175
>>>N


File Edit View Terminal Help
############
#   #
XX
XX
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
#   #
X
XXXX X
############
YOUR SCORE: 325
Do you want to rotate it?(1/0):


user@localhost:~/
File Edit View Terminal Help
Move left(L) or right(R) (Press N for no movement):
#   #
#   #
XX
XX
X
XX
XX
XX
X
X
X
XX
X
XX
X
X
X
XX
#XX   X   #
#XX XXXX X #
############
YOUR SCORE: 675
>>>N


File Edit View Terminal Help
############
GAME OVER
YOUR SCORE IS: 700
[user@localhost Anarghya]$