BINGO

Submitted by:
Rutuja Jadhav
(14BCE0002)

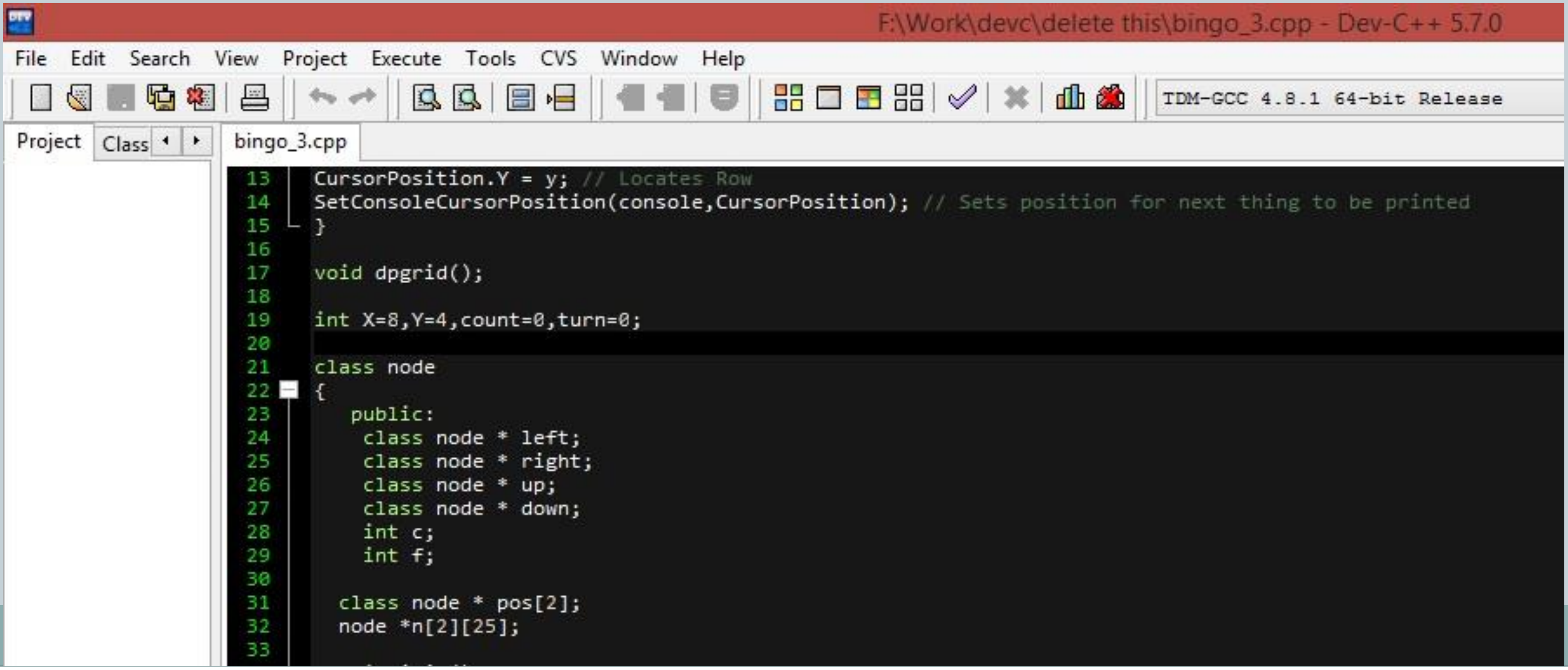
Submitted to:
Prof. Prabhakaran

Abstract

- This game is for two people playing. Each person can choose one number in the 5 * 5 box each turn. Players respectively call out a number between 1-25, which is then struck out by both the players. Straight or transverse or oblique five tag number can be a line. The one who reached five lines at first is the winner!

Implementation

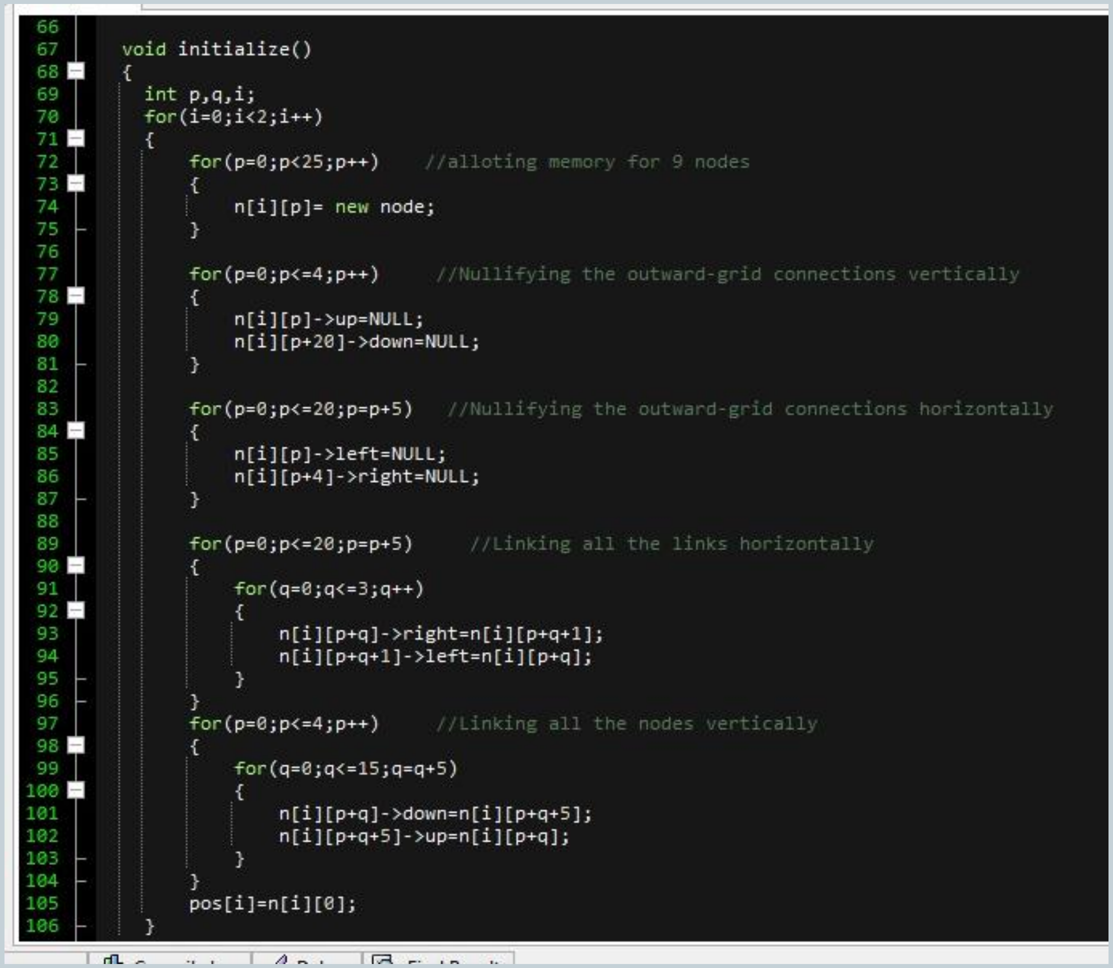
- The basic concept of the bingo here uses 2 Dimensional array of pointers to a class, in a Quadruple linked grid.


F:\Work\devc\delete this\bingo_3.cpp - Dev-C++ 5.7.0
File Edit Search View Project Execute Tools CVS Window Help
TDM-GCC 4.8.1 64-bit Release
Project Class
bingo_3.cpp

```
CursorPosition.Y = y; // Locates Row
SetConsoleCursorPosition(console,CursorPosition); // Sets position for next thing to be printed
}

void dpgrid();

int X=8,Y=4,count=0,turn=0;

class node
{
   public:
     class node * left;
     class node * right;
     class node * up;
     class node * down;
     int c;
     int f;

   class node * pos[2];
   node *n[2][25];
```

- The first dimension is used to differentiate between the two players, i.e
- n[0][1-25] will contain the grid of the first player, while n[1][1-25] will store the grid of the second player.
- The 4 class-type pointers, wiz. *up,*down,*left,*right point towards nodes, and are entwined in a complicated mesh.

Creating and Linking all nodes together

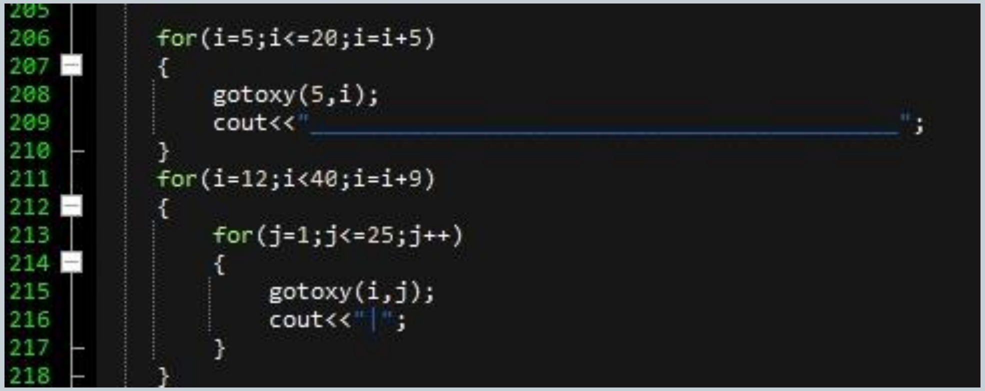


```
void initialize()
{
    int p,q,i;
    for(i=0;i<2;i++)
    {
        for(p=0;p<25;p++)    //alloting memory for 9 nodes
        {
            n[i][p]= new node;
        }

        for(p=0;p<=4;p++)     //Nullifying the outward-grid connections vertically
        {
            n[i][p]->up=NULL;
            n[i][p+20]->down=NULL;
        }

        for(p=0;p<=20;p=p+5)   //Nullifying the outward-grid connections horizontally
        {
            n[i][p]->left=NULL;
            n[i][p+4]->right=NULL;
        }

        for(p=0;p<=20;p=p+5)     //Linking all the links horizontally
        {
            for(q=0;q<=3;q++)
            {
                n[i][p+q]->right=n[i][p+q+1];
                n[i][p+q+1]->left=n[i][p+q];
            }
        }
        for(p=0;p<=4;p++)     //Linking all the nodes vertically
        {
            for(q=0;q<=15;q=q+5)
            {
                n[i][p+q]->down=n[i][p+q+5];
                n[i][p+q+5]->up=n[i][p+q];
            }
        }
        pos[i]=n[i][0];
    }
```

- 25 nodes are created and assigned memory to. Pointers to those nodes are then, in a loop, linked with each other. For eg.
- The right pointer of the first node points towards the second node, while the left pointer of the second node points towards the first node.
- The outward connections, like up pointer of the first node, are made to point towards NULL.

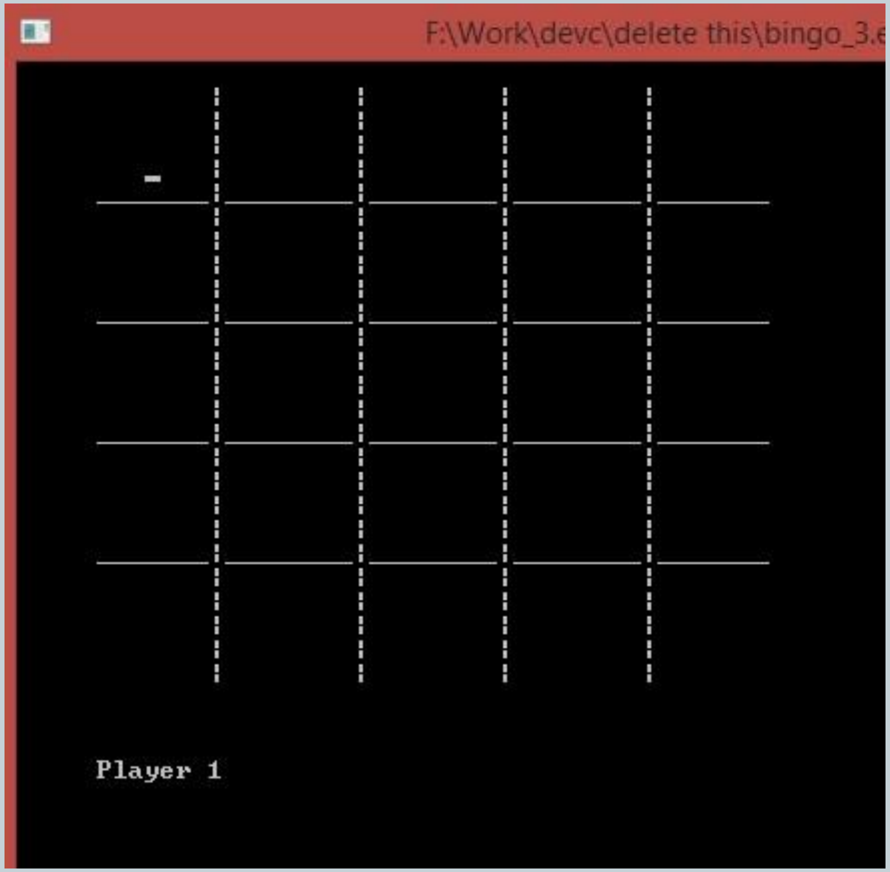
Display

- A 5x5 grid is created using | and ___ in a loop.



```
for(i=5;i<=20;i=i+5)
{
    gotoxy(5,i);
    cout<<"____________________________________________";
}
for(i=12;i<40;i=i+9)
{
    for(j=1;j<=25;j++)
    {
        gotoxy(i,j);
        cout<<"|";
    }
}
```

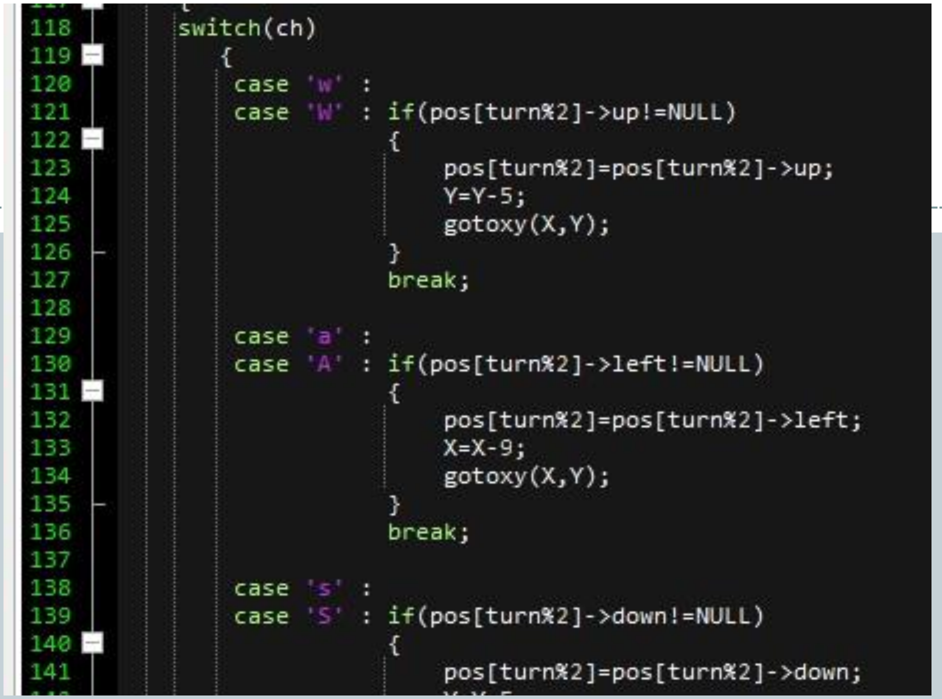
The resultant grid looks like



```
Player 1
```

Traversal through the Grid

- The traversal of the cursor is done with an inbuilt function called gotoxy(). In a loop, character input is taken, which is then used to move the cursor by changing co-ordinate values. A position pointer simultaneously points towards the node where the cursor is.



```
        switch(ch)
            {
                case 'w' :
                case 'W' : if(pos[turn%2]->up!=NULL)
                           {
                               pos[turn%2]=pos[turn%2]->up;
                               Y=Y-5;
                               gotoxy(X,Y);
                           }
                           break;

                case 'a' :
                case 'A' : if(pos[turn%2]->left!=NULL)
                           {
                               pos[turn%2]=pos[turn%2]->left;
                               X=X-9;
                               gotoxy(X,Y);
                           }
                           break;

                case 's' :
                case 'S' : if(pos[turn%2]->down!=NULL)
                           {
                               pos[turn%2]=pos[turn%2]->down;
```

- For eg, if the cursor is travelling to the node above it, the position pointer now points towards the node above the current node.