**using** System**;**

**using** System**.**Drawing**;**

**using** System**.**Linq**;**

**using** System**.**Threading**.**Tasks**;**

**using** System**.**Windows**.**Forms**;**

**namespace** TLvov\_TowersOfHanoi\_Ver2

**{**

**public** **partial** class frmTowersOfHanoi **:** Form

**{**

//Global Variable Assingment

int**[,]** disks**;**

int**[]** towerPointer**;**

string**[]** feedback **=** **{** ""**,** "You must place the disk onto a tower!"**,** "Disks cannot be placed on disks larger than themselves!"**,** "You did it! Good job!" **};**

SolidBrush**[]** p **=** **new** SolidBrush**[**7**]** **{** **new** SolidBrush**(**Color**.**Red**),** **new** SolidBrush**(**Color**.**Orange**),** **new** SolidBrush**(**Color**.**Gold**),** **new** SolidBrush**(**Color**.**YellowGreen**),** **new** SolidBrush**(**Color**.**DarkCyan**),** **new** SolidBrush**(**Color**.**RoyalBlue**),** **new** SolidBrush**(**Color**.**BlueViolet**)};**

int numOfTowers **=** 3**,** numOfDisks **=** 3**,** tower **=** 0**,** diskHeld**,** attempts **=** 0**,** feedbackInt**;**

bool movable **=** **true;**

**private** void picBoxTowerDisplay\_MouseDown**(object** sender**,** MouseEventArgs e**)**

**{**

**if** **(**movable **==** **true)**

**{**

MouseEventArgs me **=** **(**MouseEventArgs**)**e**;**

int x **=** me**.**X**;** //Gets x coordinate of cursor in relation to the picture box

**foreach** **(**int i **in** Enumerable**.**Range**(**1**,** numOfTowers **+** 1**))** //Finds tower that the x coordinate matches

**{**

**if** **(**x **<=** 67 **+** i **\*** **(**50 **+** numOfDisks **\*** 16**)** **-** **(**numOfDisks **\*** 8**)** **&** numOfTowers **>=** i**)**

**{**

tower **=** i**;**

**break;** //Stops searching once relevant tower has been found

**}**

**}**

**if** **(**tower **!=** 0**)**

**{**

**if** **(**towerPointer**[**tower **-** 1**]** **!=** 0**)**

**{**

PopStack**(**tower **-** 1**);** //"Pops" stack of relevant tower

Display**(**0**);** //Redraws the display

var bmp **=** picBoxTowerDisplay**.**Image**;**

**using** **(**var flagGraphics **=** Graphics**.**FromImage**(**bmp**))**

**{**

flagGraphics**.**FillRectangle**(**p**[**diskHeld **-** 1**],** me**.**X **-** 25**,** me**.**Y **-** 15**,** **(**diskHeld **+** 5**)** **\*** 10**,** 30**);** //Draws the disk where the cursor is (within the picture box)

picBoxTowerDisplay**.**Image **=** bmp**;**

**}**

**}**

**else**

**{**

tower **=** 0**;**

**}**

**}**

**}**

**}**

**private** void picBoxTowerDisplay\_MouseUp**(object** sender**,** MouseEventArgs e**)**

**{**

**if** **(**movable **==** **true)**

**{**

int num **=** 0**;**

**if** **(**tower **!=** 0**)** //Wont do anything unless you're actually holding a disk

**{**

MouseEventArgs me **=** **(**MouseEventArgs**)**e**;**

int x **=** me**.**X**;**

**foreach** **(**int i **in** Enumerable**.**Range**(**1**,** numOfTowers **+** 1**))** //Finds tower that the x coordinate matches

**{**

**if** **(**x **<=** 67 **+** i **\*** **(**50 **+** numOfDisks **\*** 16**)** **-** **(**numOfDisks **\*** 8**)** **&** numOfTowers **>=** i**)**

**{**

num **=** i**;**

**break;** //Stops searching once relevant tower has been found

**}**

**}**

**if** **(**num **!=** 0**)**

**{**

**if** **(**num **==** tower**)**

**{**

PushStack**(**tower **-** 1**);** //"Pushes" stack of relevant tower

attempts**--;** //Removes an attempt as it shouldn't count as one

feedbackInt **=** 0**;**

**}**

**else** **if** **(**towerPointer**[**num **-** 1**]** **==** 0**)**

**{**

PushStack**(**num **-** 1**);** //"Pushes" stack of relevant tower

feedbackInt **=** 0**;**

**}**

**else** **if** **(**diskHeld **<** disks**[**num **-** 1**,** towerPointer**[**num **-** 1**]** **-** 1**]** **||** towerPointer**[**num **-** 1**]** **==** 0**)**

**{**

PushStack**(**num **-** 1**);** //"Pushes" stack of relevant tower

feedbackInt **=** 0**;**

**}**

**else**

**{**

PushStack**(**tower **-** 1**);** //"Pushes" stack of relevant tower

attempts**--;** //Removes an attempt as it shouldn't count as one

feedbackInt **=** 2**;**

**}**

**}**

**else**

**{**

PushStack**(**tower **-** 1**);** //"Pushes" stack of relevant tower

attempts**--;** //Removes an attempt as it shouldn't count as one

feedbackInt **=** 1**;**

**}**

Display**(**0**);** //Redraws the display

attempts**++;** //Increases the attempt counter

sldTowers**.**Enabled **=** **false;** //Disables sliders

sldDisks**.**Enabled **=** **false;** //

tower **=** 0**;**

**}**

**}**

**}**

**private** void picBoxTowerDisplay\_MouseMove**(object** sender**,** MouseEventArgs e**)**

**{**

MouseEventArgs me **=** **(**MouseEventArgs**)**e**;**

int x **=** me**.**X**;** //Gets x coordinate of cursor in relation to the picture box

int n **=** 0**;**

**if** **(**movable **==** **true)**

**{**

**foreach** **(**int i **in** Enumerable**.**Range**(**1**,** numOfTowers **+** 1**))** //Finds tower that the x coordinate matches

**{**

**if** **(**x **<=** 67 **+** i **\*** **(**50 **+** numOfDisks **\*** 16**)** **-** **(**numOfDisks **\*** 8**)** **&** numOfTowers **>=** i**)**

**{**

n **=** i**;**

**break;** //Stops searching once relevant tower has been found

**}**

**}**

Display**(**n**);** //Redraws the display, this time will show a darkened tower (the one the user is going to affect)

**}**

**if** **(**tower **!=** 0**)**

**{**

var bmp **=** picBoxTowerDisplay**.**Image**;**

**using** **(**var flagGraphics **=** Graphics**.**FromImage**(**bmp**))**

**{**

flagGraphics**.**FillRectangle**(**p**[**diskHeld **-** 1**],** me**.**X **-** 25**,** me**.**Y **-** 15**,** **(**diskHeld **+** 5**)** **\*** 10**,** 30**);** //Draws the disk where the cursor is (within the picture box)

picBoxTowerDisplay**.**Image **=** bmp**;**

**}**

**}**

**}**

**private** void PopStack**(**int tower**)**

**{**

towerPointer**[**tower**]--;** //Decreases the pointer by 1

diskHeld **=** disks**[**tower**,** towerPointer**[**tower**]];** //Sets the value of the disk held to the disk at relevant location

disks**[**tower**,** towerPointer**[**tower**]]** **=** 0**;** //Removes disk of relavent location from the tower

**}**

**private** **async** void PushStack**(**int tower**)**

**{**

movable **=** **false;** //Disables player action whilst disk pushing is happening

cmdRestart**.**Enabled **=** **false;**

**for** **(**int i **=** 0**;** i **<** **(**numOfDisks **-** towerPointer**[**tower**])** **\*** 40**;** i **=** i **+** 5**)** //Determines how many times the disk will be drawn

**{**

Display**(**0**);** //Redraws the display

var bmp **=** picBoxTowerDisplay**.**Image**;**

int x **=** tower**;**

**using** **(**var flagGraphics **=** Graphics**.**FromImage**(**bmp**))**

**{**

flagGraphics**.**FillRectangle**(**p**[**diskHeld **-** 1**],** 67 **+** tower **\*** **(**50 **+** numOfDisks **\*** 16**)** **-** diskHeld **\*** 5**,** 185 **+** i **-** **(**numOfDisks **-** 3**)** **\*** 40**,** **(**diskHeld **+** 5**)** **\*** 10**,** 30**);**

picBoxTowerDisplay**.**Image **=** bmp**;** //Draws a disk on top of display at relevant location, this repeats many times forming an animation

**}**

**await** Task**.**Delay**(**1**);** //One millisecond pause between each frame

**}**

Console**.**Beep**();** //Plays beep sound when disk "lands"

disks**[**tower**,** towerPointer**[**tower**]]** **=** diskHeld**;** //Sets the value of the relevant location in the tower to that of the disk held

towerPointer**[**tower**]++;** //Increases the pointer by 1

**if** **(**towerPointer**[**numOfTowers **-** 1**]** **==** numOfDisks**)** //Checks if the player has beaten the puzzle

**{**

movable **=** **false;**

cmdRestart**.**Enabled **=** **false;**

feedbackInt **=** 3**;**

Display**(**0**);**

**}**

**else**

**{**

movable **=** **true;** //If player has not beaten the puzzle, they are now allowed to make another action

cmdRestart**.**Enabled **=** **true;**

**}**

**}**

**private** void sldTowers\_ValueChanged**(object** sender**,** EventArgs e**)**

**{**

numOfTowers **=** sldTowers**.**Value**;** //Changes the amount of towers

Load**();**

**}**

**private** void sldDisks\_ValueChanged**(object** sender**,** EventArgs e**)**

**{**

numOfDisks **=** sldDisks**.**Value**;** //Changes the amount of disks

Load**();**

**}**

**private** **async** void cmdRestart\_Click**(object** sender**,** EventArgs e**)**

**{**

attempts **=** 0**;** //Resets all important values and reloads the program, waits for a millisecond and then plays a beep noise

feedbackInt **=** 0**;**

movable **=** **true;**

Load**();**

sldDisks**.**Enabled **=** **true;**

sldTowers**.**Enabled **=** **true;**

**await** Task**.**Delay**(**1**);**

Console**.**Beep**();**

**}**

**private** **new** void Load**()**

**{**

int**[,]** setDisks **=** **new** int**[**numOfTowers**,** numOfDisks**];** //Creates a new array with dimensions based on the amounts of towers and disks

**foreach** **(**int i **in** Enumerable**.**Range**(**0**,** numOfTowers**))**

**{**

**foreach** **(**int k **in** Enumerable**.**Range**(**0**,** numOfDisks**))**

**{**

**if** **(**i **==** 0**)**

**{**

setDisks**[**i**,** k**]** **=** numOfDisks **-** k**;** //Sets all elements of the first tower to their relevant values

**}**

**else**

**{**

setDisks**[**i**,** k**]** **=** 0**;** //Sets all other elements to zero

**}**

**}**

**}**

disks **=** setDisks**;** //Sets the global array to the new array

int**[]** setTowerPointer **=** **new** int**[**numOfTowers**];** //Creates a new array of pointers based on the amount of towers

**foreach** **(**int i **in** Enumerable**.**Range**(**0**,** numOfTowers**))**

**{**

setTowerPointer**[**i**]** **=** 0**;** //Sets all elements of the array to zero

**}**

setTowerPointer**[**0**]** **=** numOfDisks**;** //Sets the pointer of the first tower to the amount of disks

towerPointer **=** setTowerPointer**;** //Sets the global array to the new array

Display**(**0**);** //Redraws the display

**}**

**public** void Display**(**int tower**)**

**{**

Controls**.**Add**(**picBoxTowerDisplay**);**

Bitmap flag **=** **new** Bitmap**(**picBoxTowerDisplay**.**Width**,** picBoxTowerDisplay**.**Height**);**

Graphics flagGraphics **=** Graphics**.**FromImage**(**flag**);**

SolidBrush p1 **=** **new** SolidBrush**(**Color**.**DarkGray**);**

SolidBrush p2 **=** **new** SolidBrush**(**Color**.**DimGray**);**

Rectangle**[]** rectsTowers **=** **new** Rectangle**[**numOfTowers**];**

**foreach** **(**int i **in** Enumerable**.**Range**(**0**,** numOfTowers**))**

**{**

rectsTowers**[**i**]** **=** **new** Rectangle**(**80 **+** i **\*** **(**50 **+** numOfDisks **\*** 16**),** 330 **-** 40 **\*** numOfDisks**,** 25**,** numOfDisks **\*** 40**);**

**}**

flagGraphics**.**FillRectangles**(**p1**,** rectsTowers**);** //Draws all towers

**if** **(**tower **!=** 0**)**

**{**

flagGraphics**.**FillRectangle**(**p2**,** 80 **+** **(**tower **-** 1**)** **\*** **(**50 **+** numOfDisks **\*** 16**),** 330 **-** 40 **\*** numOfDisks**,** 25**,** numOfDisks **\*** 40**);** //Draws darker tower if player is about to affect said tower

**}**

**foreach** **(**int i **in** Enumerable**.**Range**(**0**,** numOfTowers**))**

**{**

**foreach** **(**int k **in** Enumerable**.**Range**(**0**,** numOfDisks**))**

**{**

**if** **(**disks**[**i**,** k**]** **!=** 0**)**

**{**

flagGraphics**.**FillRectangle**(**p**[**disks**[**i**,** k**]** **-** 1**],** 67 **+** i **\*** **(**50 **+** numOfDisks **\*** 16**)** **-** disks**[**i**,** k**]** **\*** 5**,** 300 **-** k **\*** 40**,** **(**disks**[**i**,** k**]** **+** 5**)** **\*** 10**,** 30**);** //Draws all disks in relevant colours and locations

**}**

**}**

**}**

picBoxTowerDisplay**.**Image **=** flag**;**

string newLine **=** Environment**.**NewLine**;**

txtGameInfo**.**Text **=** "Bring all disks from the first tower to the last tower!" **+** newLine **+** feedback**[**feedbackInt**]** **+** newLine **+** newLine **+** "Number of towers = " **+** numOfTowers **+** newLine **+** "Number of disks = " **+** numOfDisks **+** newLine **+** newLine **+** newLine **+** "Attempts: " **+** attempts**;**

**}**

**public** frmTowersOfHanoi**()**

**{**

InitializeComponent**();**

Load**();**

**}**

**}**

**}**