Software Engineering

Project 1

Software Maintenance

Team 14
Aditi Mavalankar (201201049)
Vanshika Srivastava (201201065)
Ramakrishna Vadla (201350850)

Project : Rasterization Polygon

Objective of this experiment is to understand the steps of filling a polygon in a 2D frame buffer. Given the vertices of the polygon, the experiment outputs a filled polygon. Here we have demonstrated the steps of Scan Line Poylgon Fill algorithm.

Milestone 1

- Test Cases for existing lab

The application was tested on the following parameters and a total of about 15 test cases were used.

  - Usability
  - Responsiveness
  - Intuitiveness of the GUI
  - Data type verification
  - Out of range input values, and others

Milestone 1

- Defects / Issues

A total of about 4 issues were identified

- ➢ The application doesn't run in as a stand alone. It gives access control exception, java.io.FilePermission error. Looks to be a applet error.
- ➢ The invalid input co-ordinates enter into the "Enter co-ordinates" text box doesn't give the error message but simply ignores it. It should give the proper error message
- ➢ When clicked on "Start Experiment with Default Values" it displays the "Enter" button behind "Next iteration" button. The "Enter" button should not be displayed.
- ➢ The page doesn't refresh when experiment starts with the Default Values

Milestone 1

- Quality metrics of the existing code base
  - ➢ Lines of code : 980
  - ➢ Cyclomatic complexity : 5
  - ➢ Coupling : Monolithic design
  - ➢ Cohesion : Low
  - ➢ Maintainability Index :
  - ➢ Halstead Complexity Measure Volume: = 593.6
  - ➢ $MI = 171 - 5.2 * \ln(V) - 0.23 * (G) - 16.2 * \ln(LOC) = 25.114$
  - ➢ Web page loads in moderate time.
  - ➢ Web page loads in moderate time.
  - ➢ Web page responds in moderate time. Could not test the experiment loading time due to bugs in the code.

Milestone 2

- Identifying
    - code smells
    - Anti-patterns
    - design flaws
        - Monolithic design

- Java to JavaScript conversion
- Integration and Testing
- New Quality Metrics

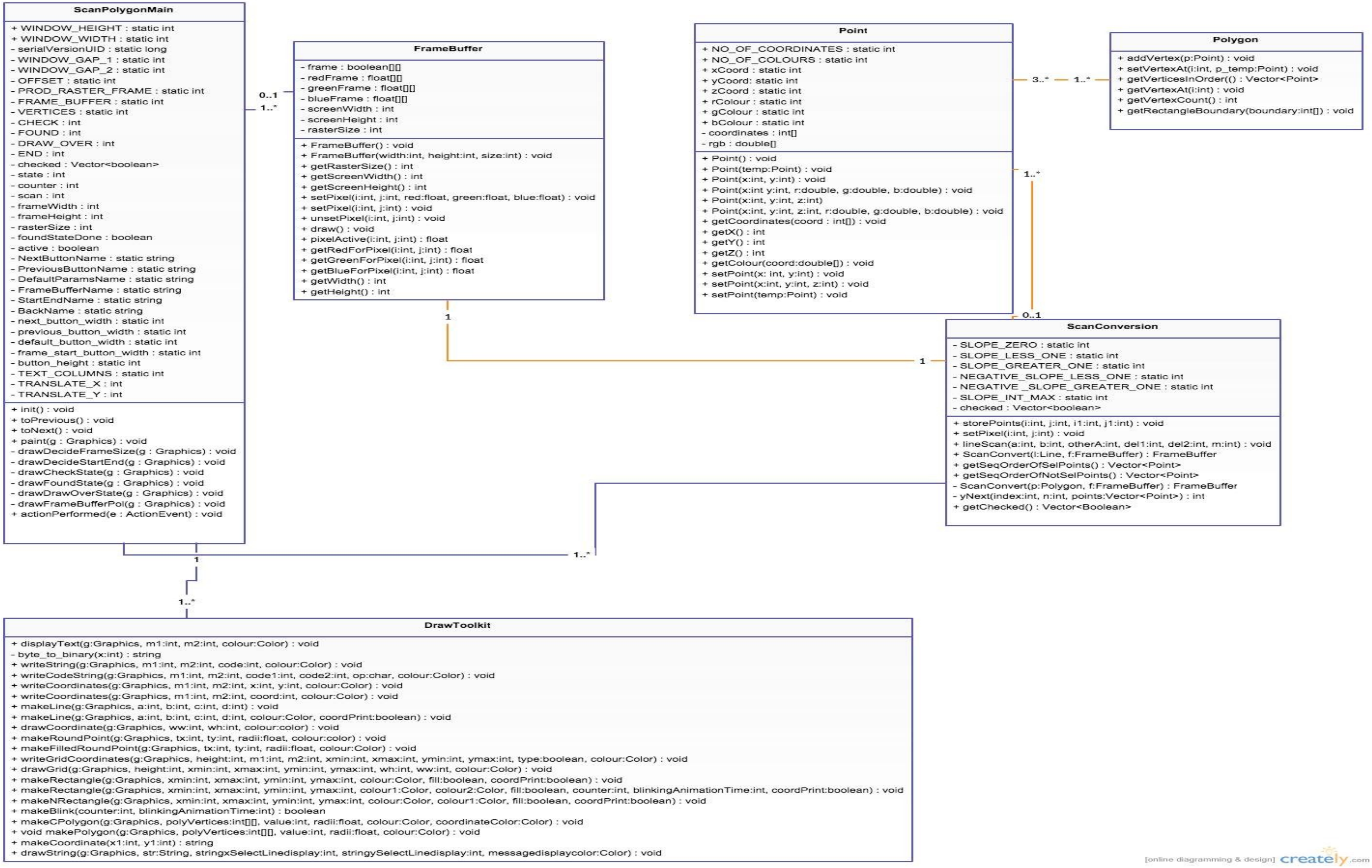
Milestone 2

- Code Smells
  - Unnecessary comments
  - Long Parameter List
  - High conditional complexity
  - Combinatorial Explosion
  - Large Class
  - Very uncommunicative names
  - Inconsistent names
  - Data Clumps
  - Inappropriate Intimacy
  - Middle man

Milestone 2

- Anti-Patterns
  - The Blob
    - Single controller class
  - Lava Flow
    - Large commented-out code with no explanations
    - Lot’s of TODOs
  - Golden Hammer

UML Class Diagram for Java Code



```
ScanPolygonMain
+ WINDOW_HEIGHT : static int
+ WINDOW_WIDTH : static int
- serialVersionUID : static long
- WINDOW_GAP_1 : static int
- WINDOW_GAP_2 : static int
- OFFSET : static int
- PROD_RASTER_FRAME : static int
- FRAME_BUFFER : static int
- VERTICES : static int
- CHECK : int
- FOUND : int
- DRAW_OVER : int
- END : int
- checked : Vector<boolean>
- state : int
- counter : int
- scan : int
- frameWidth : int
- frameHeight : int
- rasterSize : int
- foundStateDone : boolean
- active : boolean
- NextButtonName : static string
- PreviousButtonName : static string
- DefaultParamsName : static string
- FrameBufferName : static string
- StartEndName : static string
- BackName : static string
- next_button_width : static int
- previous_button_width : static int
- default_button_width : static int
- frame_start_button_width : static int
- button_height : static int
- TEXT_COLUMNS : static int
- TRANSLATE_X : int
- TRANSLATE_Y : int
+ init() : void
+ toPrevious() : void
+ toNext() : void
+ paint(g : Graphics) : void
- drawDecideFrameSize(g : Graphics) : void
- drawDecideStartEnd(g : Graphics) : void
- drawCheckState(g : Graphics) : void
- drawFoundState(g : Graphics) : void
- drawDrawOverState(g : Graphics) : void
- drawFrameBufferPol(g : Graphics) : void
+ actionPerformed(e : ActionEvent) : void
```

```
FrameBuffer
- frame : boolean[][]
- redFrame : float[][]
- greenFrame : float[][]
- blueFrame : float[][]
- screenWidth : int
- screenHeight : int
- rasterSize : int
+ FrameBuffer() : void
+ FrameBuffer(width:int, height:int, size:int) : void
+ getRasterSize() : int
+ getScreenWidth() : int
+ getScreenHeight() : int
+ setPixel(i:int, j:int, red:float, green:float, blue:float) : void
+ setPixel(i:int, j:int) : void
+ unsetPixel(i:int, j:int) : void
+ draw() : void
+ pixelActive(i:int, j:int) : float
+ getRedForPixel(i:int, j:int) : float
+ getGreenForPixel(i:int, j:int) : float
+ getBlueForPixel(i:int, j:int) : float
+ getWidth() : int
+ getHeight() : int
```

```
Point
+ NO_OF_COORDINATES : static int
+ NO_OF_COLOURS : static int
+ xCoord : static int
+ yCoord: static int
+ zCoord : static int
+ rColour : static int
+ gColour : static int
+ bColour : static int
- coordinates : int[]
- rgb : double[]
+ Point() : void
+ Point(temp:Point) : void
+ Point(x:int, y:int) : void
+ Point(x:int y:int, r:double, g:double, b:double) : void
+ Point(x:int, y:int, z:int)
+ Point(x:int, y:int, z:int, r:double, g:double, b:double) : void
+ getCoordinates(coord : int[]) : void
+ getX() : int
+ getY() : int
+ getZ() : int
+ getColour(coord:double[]) : void
+ setPoint(x: int, y:int) : void
+ setPoint(x:int, y:int, z:int) : void
+ setPoint(temp:Point) : void
```

```
Polygon
+ addVertex(p:Point) : void
+ setVertexAt(i:int, p_temp:Point) : void
+ getVerticesInOrder(() : Vector<Point>
+ getVertexAt(i:int) : void
+ getVertexCount() : int
+ getRectangleBoundary(boundary:int[]) : void
```

```
ScanConversion
- SLOPE_ZERO : static int
- SLOPE_LESS_ONE : static int
- SLOPE_GREATER_ONE : static int
- NEGATIVE_SLOPE_LESS_ONE : static int
- NEGATIVE _SLOPE_GREATER_ONE : static int
- SLOPE_INT_MAX : static int
- checked : Vector<boolean>
+ storePoints(i:int, j:int, i1:int, j1:int) : void
+ setPixel(i:int, j:int) : void
+ lineScan(a:int, b:int, otherA:int, del1:int, del2:int, m:int) : void
+ ScanConvert(l:Line, f:FrameBuffer) : FrameBuffer
+ getSeqOrderOfSelPoints() : Vector<Point>
+ getSeqOrderOfNotSelPoints() : Vector<Point>
- ScanConvert(p:Polygon, f:FrameBuffer) : FrameBuffer
- yNext(index:int, n:int, points:Vector<Point>) : int
+ getChecked() : Vector<Boolean>
```

```
DrawToolkit
+ displayText(g:Graphics, m1:int, m2:int, colour:Color) : void
- byte_to_binary(x:int) : string
+ writeString(g:Graphics, m1:int, m2:int, code:int, colour:Color) : void
+ writeCodeString(g:Graphics, m1:int, m2:int, code1:int, code2:int, op:char, colour:Color) : void
+ writeCoordinates(g:Graphics, m1:int, m2:int, x:int, y:int, colour:Color) : void
+ writeCoordinates(g:Graphics, m1:int, m2:int, coord:int, colour:Color) : void
+ makeLine(g:Graphics, a:int, b:int, c:int, d:int) : void
+ makeLine(g:Graphics, a:int, b:int, c:int, d:int, colour:Color, coordPrint:boolean) : void
+ drawCoordinate(g:Graphics, ww:int, wh:int, colour:color) : void
+ makeRoundPoint(g:Graphics, tx:int, ty:int, radii:float, colour:color) : void
+ makeFilledRoundPoint(g:Graphics, tx:int, ty:int, radii:float, colour:Color) : void
+ writeGridCoordinates(g:Graphics, height:int, m1:int, m2:int, xmin:int, xmax:int, ymin:int, ymax:int, type:boolean, colour:Color) : void
+ drawGrid(g:Graphics, height:int, xmin:int, xmax:int, ymin:int, ymax:int, wh:int, ww:int, colour:Color) : void
+ makeRectangle(g:Graphics, xmin:int, xmax:int, ymin:int, ymax:int, colour:Color, fill:boolean, coordPrint:boolean) : void
+ makeRectangle(g:Graphics, xmin:int, xmax:int, ymin:int, ymax:int, colour1:Color, colour2:Color, fill:boolean, counter:int, blinkingAnimationTime:int, coordPrint:boolean) : void
+ makeNRectangle(g:Graphics, xmin:int, xmax:int, ymin:int, ymax:int, colour:Color, colour1:Color, fill:boolean, coordPrint:boolean) : void
+ makeBlink(counter:int, blinkingAnimationTime:int) : boolean
+ makeCPolygon(g:Graphics, polyVertices:int[][], value:int, radii:float, colour:Color, coordinateColor:Color) : void
+ void makePolygon(g:Graphics, polyVertices:int[][], value:int, radii:float, colour:Color) : void
+ makeCoordinate(x1:int, y1:int) : string
+ drawString(g:Graphics, str:String, stringxSelectLinedisplay:int, stringySelectLinedisplay:int, messagedisplaycolor:Color) : void
```

[online diagramming & design] creately.com

Thank You