Problem 1 (25%). Once again, you are given the StoopgesGUI (SGUI) library. The following are three SGUI event service routines and one utility function:

MouseService(TbuttonEnum WhichButton, TactionEnum WhatAction, int x, int y);
// TbuttonEnum: can be LeftMouseButton, or RightMouseButton
// TactionEnum: can be ButtonDown - mouse button went down
//                ButtonUp      - mouse button went up
//                MotionWithButtonDown - mouse moving with button down
//                or MotionWithButtonUp - mouse moving without any button depressed
//  x and y are the current mouse position

KeyboardService(unsigned char InputCharacter);
// InputCharacter: use’s input character

SGUIRedrawNeeded();
// Informs SGUI that a redraw is necessary

RedrawService();
// SGUI will call this function when redraw is necessary
// redraw can either caused by system events, or
// redraws can caused by someone calling: SGUIRedrawNeeded

With the above service and utility function, you are asked to implement an application that supports the definition/manipulation of a circle and input a string from the user. Here are the specifications:

(i) If the circle is not initialized, or is completely defined, left mouse button down defines the center of the circle and starts the Circle Definition State (CDS).
(ii) While in CDS, you must echo the circle with radius defined by the center to the current mouse position.
(iii) The left mouse button goes down while a circle is in CDS will completely define the circle and thus exists the CDS.
(iv) After the circle is completely defined, a user must be able to define a new center position for the circle with a subsequent left mouse button down event and the circle should appear at the new center position immediately.
(v) If your application is not already in String Input State (SIS), right mouse button down starts the String Input State (SIS).
(vi) While in SIS, all circle manipulation operations are suspended, i.e. the user cannot change the state of the circle even if the circle is in CDS. However, you must still echo the current state of the circle by redrawing it when necessary.
(vii) While in SIS, you are to collect all characters a user types. You will exist SIS if there are more characters than the predefined MaxStringLength, or the left mouse button goes down. At this point, you must print the collected characters to the console by calling cerr.
(viii) Note that if a left mouse button down event ends SIS then it should not cause the circle to enter or exist CDS. Remember that circle operations are suspended while in SIS.

You may find the following SGUI library functions useful:

void DrawCircle(int centerX, int centerY, double radius);
// draws a circle at (centerX, centerY) with radius

void EraseScreen();
// erase everything on the screen.
You are told that you have access to the following global variables:

\[
\begin{align*}
\text{int} & \quad \text{CenterX, CenterY}; \quad \text{// initialized to -1} \\
\text{double} & \quad \text{Radius}; \quad \text{// initialized to -1} \\
\text{int} & \quad \text{StringLength}; \quad \text{// initialized to 0 (zero)} \\
\text{char} & \quad \text{InputString[MaxStringSize];} \quad \text{// MaxStringSize is defined somewhere}
\end{align*}
\]

You can define any other global/local variables you find necessary. Please show the implementation of

\[
(4\%) \quad \text{RedrawService()},
\]

\[
(7\%) \quad \text{KeyboardService()}, \quad \text{and}
\]

\[
(14\%) \quad \text{MouseMove()}
\]

routines that would support the specified functionality. Please show the details of how you would compute the radius of the circle. When the state of the drawable is out-of-date, you must call \text{SGUIRedrawNeeded()} to trigger redraw. Please make sure you only make the \text{SGUIRedrawNeeded()} requests when necessary. Excessive redraw requests will result in inefficient implementation and will result in points lost.

As we have discussed in lectures, the scaling, moving, and rotation matrices are:

\[
\begin{align*}
S(s_x, s_y) &= \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad & M(m_x, m_y) &= \begin{bmatrix} 1 & 0 & m_x \\ 0 & 1 & m_y \\ 0 & 0 & 1 \end{bmatrix} \\
R(\theta) &= \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{align*}
\]

It is ok for your answers to contain \(S(x, y)\), or \(M(x, y)\), or \(R(\theta)\).

**Problem 2. (35%)**. For the following diagrams:

\[T_1\] and \(T_2\) are transformation matrices that will transform objects to and from the left and right diagrams. You do not need compute the values for \((x_1, y_1)\) in the diagram on the right, when necessary, just use \((x_i, y_i)\) in your solution.

**Problem 2a (12%)** Compute \(T_1\) and \(T_2\).

Note: for answers to parts b, c, and d, besides the provided \text{DrawUnitSquare()} you cannot use any other OpenGL primitives (you cannot call \text{glBegin()}). Also, please clearly show how you manipulate and utilize the OpenGL Matrix Stack.
Problem 2b (8%) You are given:

```c
void DrawUnitSquare() {
    double sq[4][2] = {{0, 0}, {0, 1}, {1,1}, {1,0} };
    SetupPatternAndColorCorrectly();
    // this will set up the correct pattern and color for us
    glBegin(GL_POLYGON);
    for (int i=0; i<4; i++)
        glVertex2d(sq[i][0], sq[i][1]);
    glEnd();
}
```

Remembering that in OpenGL, if you want to Rotate, then, Translate, and performs Scale last you would …

```c
// glScale(…)
// glTranslate(…)
// glRotate(…)
// glVertex(…)
```

// NOTE: I AM NOT suggesting the above transformation sequence is correct or incorrect
// I am just reminding you how OpenGL works …

Show how you would utilize `DrawUnitSquare()` to implement `DrawOrgShape()` to draw the shape in the diagram on the left.

Problem 2c (5%) Show how you would implement `DrawFinalShape()` by calling `DrawOrgShape()` to draw the shape in the diagram on the right.

Problem 2d (10%) Show how you would utilize only `DrawUnitSquare()` together with OpenGL transformation and matrix manipulation functions to implement `DrawFinalShape()` to draw shape in the diagram on the right. You cannot call your `DrawOrgShape()` function from Problem 2b.
**Problem 3 (40%).** You are told that the `DrawRoom()` function draws the following figure:

![Room Diagram](image)

Given the OpenGL `glViewport()` and `gluOrtho2D()` functions:

```c
glViewport(lowerLeft, lowerRight, width, height)
gluOrtho2D(left, right, bottom, top)
```

and that the output drawable size is 100x100.

**Problem 3a (10%)** With the given OpenGL code:

```c
gMatrixMode(GL_PROJCTION);
gLoadIdentity();
gViewport(10, 20, 70, 80);
gluOrtho2D(5, 12, 6, 14);
gMatrixMode(GL_MODELVIEW);
gLoadIdentity();
DrawRoom();
```

Please sketch what you would expect to see on the drawable. Please clearly indicate the relative position (location and size) of the viewport in the drawable. Clearly show the objects you would expect to see in the viewport. In this case you do **not** need to indicate the corner positions of the visible objects in the viewport.

**Problem 3b (20%)** With the following OpenGL code:

```c
gMatrixMode(GL_PROJCTION);
gLoadIdentity();
gViewport(0, 0, 100, 100);
gluOrtho2D(0, 10, 0, 10);
gMatrixMode(GL_MODELVIEW);
gLoadIdentity();
gTranslated(-3, -4);
DrawRoom();
```

i. **(8%)** What is the content of the top of GL_PROJCTION matrix stack when `DrawRoom()` function is called? Please express your answer in terms of \( S(x, y) \), \( M(x, y) \), and \( R(\theta) \).

ii. **(2%)** What is the content of the top of GL_MODELVIEW matrix stack when `DrawRoom()` function is called? Please express your answer in terms of \( S(x, y) \), \( M(x, y) \), and \( R(\theta) \).
Problem 3(b) (continue)

iii. (10%) Please sketch what you would expect to see on the drawable. Clearly indicate all the coordinate positions of the visible corners in Device Coordinate space.

Problem 3c. (10%) For the following commands:

```gl
glMatrixMode(GL_PROJCTION);
glLoadIdentity();
glViewport(0, 0, 100, 100);
gluOrtho2D(2.5, 7.5, 2.5, 7.5);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
DrawRoom();
```

A Window is defined in the World Coordinate with 5x5 window size and ranges:

\[ 2.5 \leq x \leq 7.5 \]
\[ 2.5 \leq y \leq 7.5 \]

Assuming you have access to two valuator objects, HorizontalValuator(), and VerticalValuator(). Under user’s control, these two valuators each are capable of returning a value between 0 and 1. Explain how you would modify the above commands to use the values returned by the two valuators to achieve scrolling in the world coordinate with the 5x5 window size, for ranges:

\[ 0.0 \leq x \leq 10.0 \]
\[ 0.0 \leq y \leq 10.0 \]