Aufgabe:	1	2	3	Σ
Punktzahl:	/20	/20	/20	/60

Klausur: **Computergraphik I (CG-1)** **Probeklausur**	Prüfer: **Kolb**	Datum/Zeit:
Semester:	Prüfungsdauer: **60 Minuten**	Max. Punktzahl: **60**

Hilfsmittel:

- Schreibgeräte (kein Bleistift, keine Farbstifte (rot und grün))
- Lineal, Geo-Dreieck
- Nicht-programmierbarer Taschenrechner

Bitte beachten:

- Tragen Sie auf jedes Blatt Ihren **Namen** und Ihre **Matr.-Nr.** in die entsprechenden Felder ein.
- Benutzen Sie nur das ausgeteilte **Papier** – kein eigenes Papier. **Notizen** sind auf den Rückseiten der Klausurblätter zulässig und werden nicht als Lösungsbestandteil gewertet. Die Mitnahme von Klausurbögen aus dem Raum ist nicht gestattet.
- **Die Klammerung** der Klausurbögen darf **nicht gelöst** werden.
- Schreiben Sie **deutlich**! Doppelte, unleserliche oder mehrdeutige Lösungen sind ungültig.
- Erscheint Ihnen eine Aufgabenstellung mehrdeutig, melden Sie sich bitte durch Handzeichen bei einer Aufsichtsperson.
- Ein **Täuschungsversuch** (Abschreiben, Nutzung unerlaubter Hilfmittel, Kommunikation mit anderen Klausurteilnehmern, etc.) führt umgehend zum Ausschluss und Nichtbestehen. Es erfolgt keine Vorwarnung.

Bitte ausfüllen:

Name:	**Matrikel-Nr.**:

Aufgabe 1 Linien und Polygon Rasterisierung (20 Punkte)

1.1. Gegeben ist eine Strecke $\overline{\mathbf{P}_1\mathbf{P}_2}$ in Raster-Koordinaten:

$$\mathbf{P}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}, \quad \mathbf{P}_2 = \begin{pmatrix} 0 \\ 4 \end{pmatrix}$$

Welche Anpassung ist durchzuführen, so dass die Einschränkungen des Bresenham-Algorithmus erfüllt sind? Ermitteln Sie Punkte $\mathbf{Q}_1, \mathbf{Q}_2$, die durch die gewählte Anpassung aus $\mathbf{P}_1, \mathbf{P}_2$ entstehen.

1.2. Gegeben sind zwei Punkte:

$$\mathbf{Q}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \text{ und } \mathbf{Q}_2 = \begin{pmatrix} 4 \\ 1 \end{pmatrix}.$$

Zeigen Sie das für die Strecke $\overline{\mathbf{Q}_1\mathbf{Q}_2}$ die Bedingungen für den Bresenham-Algorithmus erfüllt sind. Berechnen Sie anschließend die Rasterpunkte die der Bresenham-Algorithmus für die Strecke $\overline{\mathbf{Q}_1\mathbf{Q}_2}$ liefert. Verwenden Sie zur Durchführung des Algorithmus die folgende Tabelle:

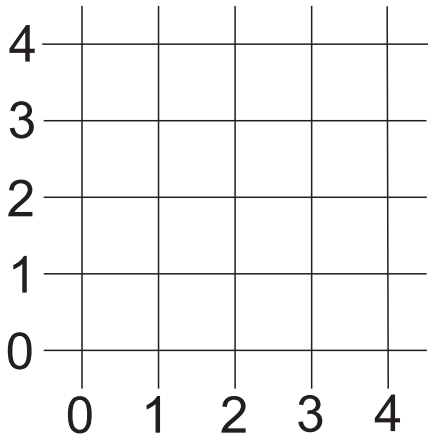
x	**y**	**d**

Hinweis: $d_{init} = 2\Delta y - \Delta x,\ inc_1 = 2\Delta y,\ inc_2 = 2(\Delta y - \Delta x)$.

1.3. Die Kantendaten eines Polygons sind in der nachfolgenden Edge-Table gegeben. Die Edge-Table spiegelt den Anfangszustand des Scanline-Algorithmus wider. Für die Steigung der Polygonkanten gilt $\frac{\Delta x}{\Delta y} = dx + \frac{rx}{\Delta y}$ und zusätzlich gilt $x(y) = x + \frac{k}{\Delta y}$. Zeichnen Sie nun das Polygon in das Koordinatensystem ein.

EDGE-TABLE

ID	x	k	y_{min}	y_{max}	dx	rx	Δy
AB	2	0	0	4	-1	1	2
AD	2	0	0	4	0	1	2
CB	2	0	2	4	-1	0	2
CD	2	0	2	4	1	0	2



1.4. Führen Sie nun die Rasterisierung für die Edge Table aus Aufgabe 1.**??** mit Hilfe des Scanline-Algorithmus durch. Vervollständigen Sie die Active Edge-Table in der folgenden Vorlage, indem Sie den Algorithmus durchführen. Geben Sie anschließend auch die Pixel-Intervalle für alle Scanlines die einen Schnittpunkt mit dem Polygon aufweisen an.

Scanline	ID	x	k	LR: pixel(x)

Scanline	ID	x	k	LR: pixel(x)

Für die Laufvariablen x und k gilt:

$$k + rx < \Delta y \; : \; x \leftarrow x + dx \; ; \; k \leftarrow k + rx$$
$$k + rx \geq \Delta y \; : \; x \leftarrow x + dx + 1 \; ; \; k \leftarrow k + rx - \Delta y$$

mit $Initial : k = 0$.

Aufgabe 2 Raycasting, Hierarchien (20 Punkte)

Beim Raycasting wird für jeden Bildpunkt ein Strahl mit den Objekten der Szene geschnitten. Im Folgenden soll ein Point-in-Polygon-Test durchgeführt werden, um zu prüfen, ob der Strahl

$$\mathbf{G}(\alpha) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \alpha \cdot \begin{pmatrix} -4 \\ 1 \\ 3 \end{pmatrix},\ \alpha \geq 0$$

ein Polygon $\mathcal{P}$ mit den folgenden 5 Eckpunkten trifft.

$$\mathbf{P}_1 = \begin{pmatrix} 0 \\ 1 \\ 2 \end{pmatrix},\ \mathbf{P}_2 = \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix},\ \mathbf{P}_3 = \begin{pmatrix} -2 \\ 0 \\ -1 \end{pmatrix},\ \mathbf{P}_4 = \begin{pmatrix} -2 \\ -0.5 \\ -0.5 \end{pmatrix},\ \mathbf{P}_5 = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

2.1. Bestimmen Sie den Schnittpunkt $\mathbf{S}$ der Polygonebene mit dem Strahl $\mathbf{G}(\alpha)$. Stellen sie hierfür zunächst die Ebenengleichung der von $\mathcal{P}$ definierten Ebene auf.

2.2. Nachfolgend sei ein weiterer Schnittpunkt $\mathbf{S} = (-1, 0.5, 0.5)^T$ gegeben. Projizieren sie das Polygon $\mathcal{P}$ sowie den Schnittpunkt $\mathbf{S}$ zweidimensional auf eine Ebene. Geben sie die projizierten Punkte $\mathbf{P}'_i$ und $\mathbf{S}'$ an und begründen sie ihre Ebenen-Wahl.

Die Ergebnis-Punkte $\mathbf{P}'_i$ und $\mathbf{S}'$ liegen anschließend im (x', y')-Koordinaten-System vor.

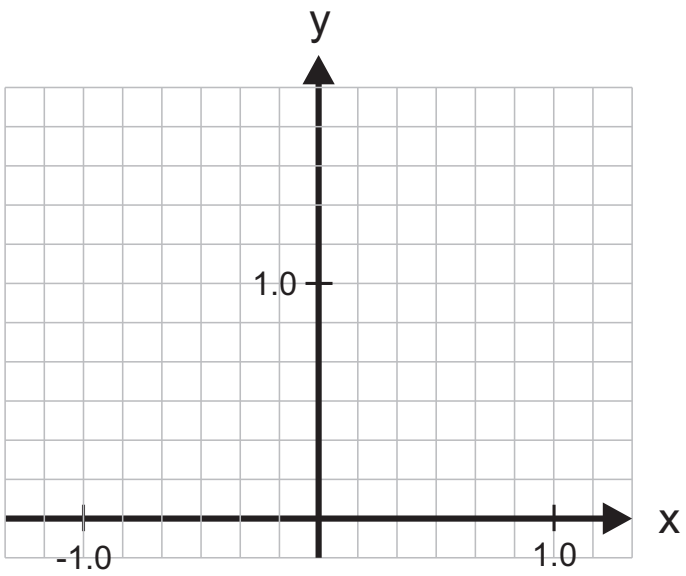
2.3. Verschieben sie anschließend die projizierten Punkte, so dass der Schnittpunkt $\mathbf{S}'$ im Ursprung liegt. Führen sie für das verschobene Polygon mit den Punkte $\mathbf{P}_i''$ für die Kanten $\overline{\mathbf{P}_i''\mathbf{P}_{i+1}''}$ $(i = 1, \ldots, 5,\ \mathbf{P}_6'' = \mathbf{P}_1'')$ die Trivial-Reject-Prüfung entlang der x'-Achse durch.

2.4. Bestimmen sie für die verbleibende(n) Kante(n) die Schnittpunkte mit der x'-Achse und überprüfen sie, ob $\mathbf{S}$ inner- oder außerhalb des Polygons liegt.

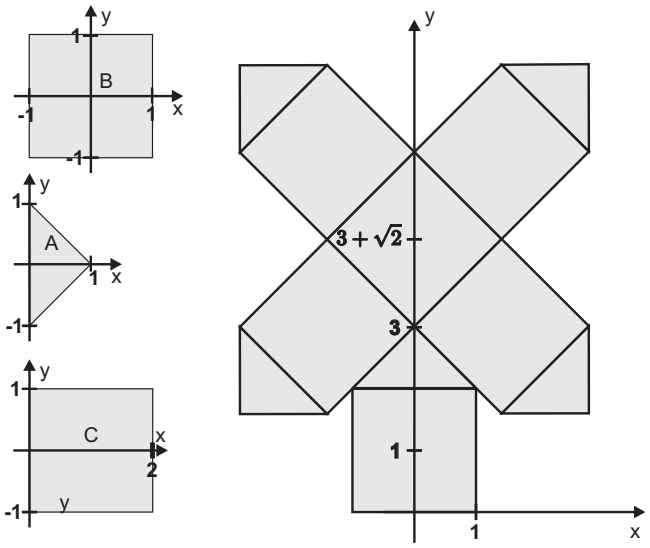
Aufgabe 3 Transformationen, Hierarchien (20 Punkte)

3.1. Zeichnen Sie das Ergebnis der OpenGL-Aufrufe in das (Welt-)Koordinatensystem ein. Kennzeichnen Sie die einzelnen Primitive in der Zeichnung.

```
glTranslatef(-0.5f,0.5f,0);
glPushMatrix();
    glRotatef(-90,0,0,1);
    glTranslatef(0,0.5f,0);
    glScalef(0.5f,1.0f,1.0f);
    // PRIMITIVE 0
    glBegin(GL_TRIANGLES);
        glColor3f(1,0,0);
        glVertex3f(0,0,0);
        glVertex3f(-1,1,0);
        glVertex3f(-2,0,0);
    glEnd();
glPopMatrix();
glPushMatrix();
    glTranslatef(0.5,0.5,0);
    glPushMatrix();
        glRotatef(-180.0f,0,0,1);
        glScalef(1,0.5f,1);
        // PRIMITIVE 1
        glBegin(GL_TRIANGLES);
            glVertex3f(1,0,0);
            glVertex3f(0,1,0);
            glVertex3f(0,-1,0);
        glEnd();
    glPopMatrix();
    glRotatef(-90.0f,0,0,1);
    // PRIMITIVE 2
    glBegin(GL_LINES);
        glVertex3f(-0.5,-1,0);
        glVertex3f(-0.5,1,0);
        glVertex3f(-0.5,1,0);
        glVertex3f(0.5,1,0);
        glVertex3f(0.5,1,0);
        glVertex3f(0.5,-1,0);
        glVertex3f(0.5,-1,0);
        glVertex3f(-0.5,-1,0);
    glEnd();
glPopMatrix();
```

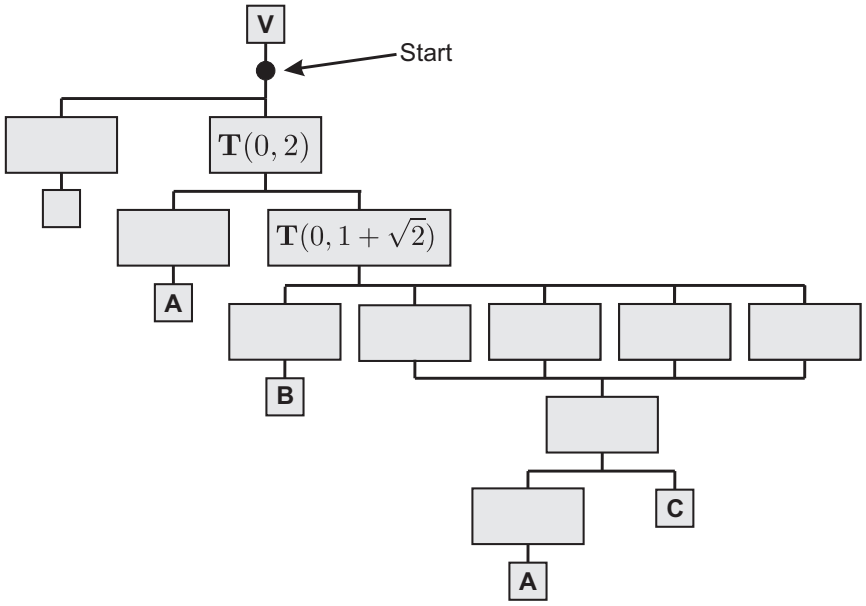


3.2. In dieser Teilaufgabe sind die Objekte A, B und C gegeben, mit deren Hilfe das Windrad als **Tangram**-Objekt konstruiert werden soll (siehe Abbildung).



Ergänzen Sie die folgende **Transformationshierarchie**, so dass genau das vorgegebene Tangram gebildet wird. Die Knoten des Hierarchie-Graphen haben die folgende Bedeutung:

V	Viewing-Transformation (hier: $V = Id$)
$T_{(x,y)}$	Translation um den Vektor (x, y)
R_ϕ	Rotation um den Winkel ϕ gegen den Uhrzeigersinn
$S_{a,b}$	Skalierung um a, b in x- bzw. y-Richtung
A, B, C	Zeichnen des Objektes A, B bzw. C



Hinweise: Objekte dürfen mehrfach verwendet werden. Achten Sie darauf, dass in der Hierarchie die Transformationen bzgl. des **lokalen** Koordinatensystem (Objektkoordinaten) zu interpretieren sind!