Fragen:

1. Erstellen einer Konsol-Applikation

1. Was sehen Sie nach der Ausführung des Programmes im Graphik-Fenster?

Ein rotes Viereck ist zu sehen.

2. Sind die Anweisungen glBegin und glEnd unbedingt notwendig? Was passiert, wenn man sie weglässt?

Ja, diese Anweisungen sind notwendig, da ohne diese kein Viereck zu sehen ist.  glBegin und glEnd umschließen eine Liste von Eckpunkten. Bei glBegin markiert dabei den Anfang der Liste und  der Typ des Primitivs wird in Klammern dahinter angegeben. glEnd markiert das Ende der Liste.

3. An welchen Positionen im Programm müssen diese glColor4f-Anweisungen stehen?

Nach den einzelnen Punktangaben. Zum Beispiel nach glVertex3f(-0.5, 0.5, -0.5);

4. Welche Position im Code ist am besten zum Aufruf von glClear geeignet und warum?

Direkt am Anfang der RenderScene() //Zeichenfunktion.

5. In welcher Reihenfolge müssen glClear und glClearColor aufgerufen werden?

Zuerst muss glClear und dann glClearColor aufgerufen werden. Zuerst wird der Hintergrund gelöscht. Dann eine neue Hintergrundfarbe gesetzt.

Zuerst muss glClearColor und dann glClear aufgerufen werden. In glClearColor wird die Farbe definiert, die in glClear benutzt werden.

6. Was passiert, wenn Sie in einer Animation glClear(GL\_COLOR\_BUFFER\_BIT); weglassen? (Ihre Vermutung können Sie aber erst in Teil 4 überprüfen.)

Es gibt keinen Hintergrund bzw. es wird nur der Default-Hintergrund angezeigt.

7. Welche der beiden Flächen sehen Sie?

Nur das grüne Viereck ist zu sehen.

8. Erzeugen Sie die Flächen mal in einer anderen Reihenfolge: Was fällt Ihnen auf und warum ist das so?

Es handelt sich um eine 2D-Darstellung. Da die x- und Y-Werte identisch sind, sieht man nur das Viereck, welches zuletzt gezeichnet wurde.

9. a) Entspricht die Ansicht der Flächen nun Ihren Erwartungen?

Ich hätte erwartet, das grüne Viereck noch zu sehen hinter dem rot, blauen Viereck. Es war aber nur das rot, blaue Viereck zu sehen, egal ob zuerst das grüne oder rot-blaue Viereck gezeichnet wurde. Das kommt daher, da das grüne und das rot-blaue Viereck identisch sind bis auf die Z-Werte. Das rot-blaue Viereck ist näher an der Kamera und überdeckt somit das grüne Viereck.

b) Beschreiben Sie kurz, wie der Z-Buffer funktioniert.

Der Wert beschreibt den Abstand vom aktuellen Standpunkt der Kamera. Werden negative Werte gesetzt, dann liegt die Fläche hinter der Kamera.

2. OpenGL-Kamera

10. Welche Fläche sehen Sie und warum sehen Sie gerade diese Fläche?

Man sieht die Rückseite. Die Farben (Schwarz, Grün, Cyan und Blau) sprechen für die Rückseite.

11. Probieren Sie (ohne Translation des Würfels) die folgenden Kamerapositionen aus und dokumentieren Sie die dazu verwendeten gluLookAt-Aufrufe:

a) Betrachtung der Szene von vorne oben; (Kameraposition: (0., 1., 1.). Sind die Parameterwerte in glOrtho richtig gesetzt? Falls NEIN, was stimmt nicht?

Tiefe der Frustrum stimmt nicht, zu klein.

b) Betrachtung der Szene direkt von rechts; Kameraposition: (1., 0., 0.). Wie lauten die kompletten Aufrufe von glortho und gluLookAt?

gluLookAt(1., 0., 0., 0., 0., 0., 0., 1., 0.);

glOrtho(-1., 1., -1., 1., 0., 1.);

c) Betrachtung der Szene von rechts oben: wie lautet die Kameraposition: (?, ?, ?).

gluLookAt(1., 1., 0., 0., 0., 0., 0., 1., 0.);

3. OpenGL-Transformationen - anhand eines Szenengraphs

12. Ist es möglich, den Rotate-Befehl vom „oberen“ Teil in die beiden Äste des Szenegraphs zu verlagern? (Wenn JA, wohin und wie? Wenn NEIN, warum nicht?)

4. OpenGL-Animationen

13. Vergessen Sie bitte nicht die in Frage 6 angesprochene Nicht-Ausführung des glClear-Befehls für den Bild-Hintergrund. Der Tiefenpuffer muss trotzdem gelöscht werden:  glClear ( GL\_DEPTH\_BUFFER\_BIT );