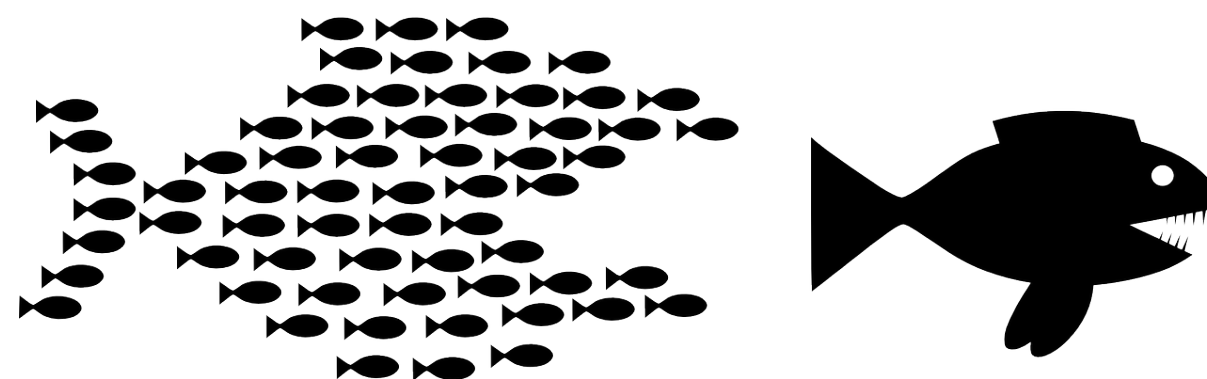




# Computergraphik I

## Organisatorisches

Organize!

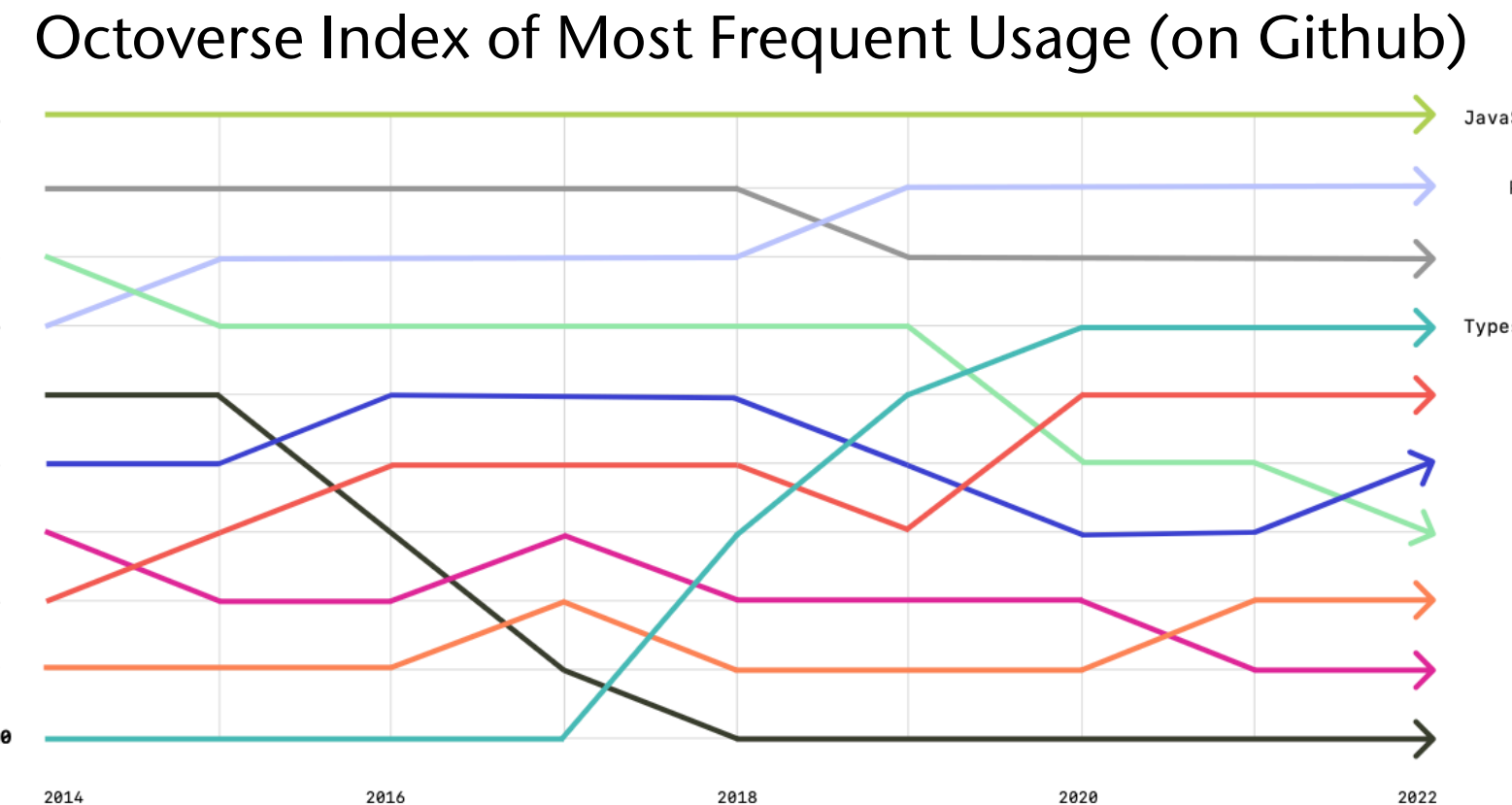
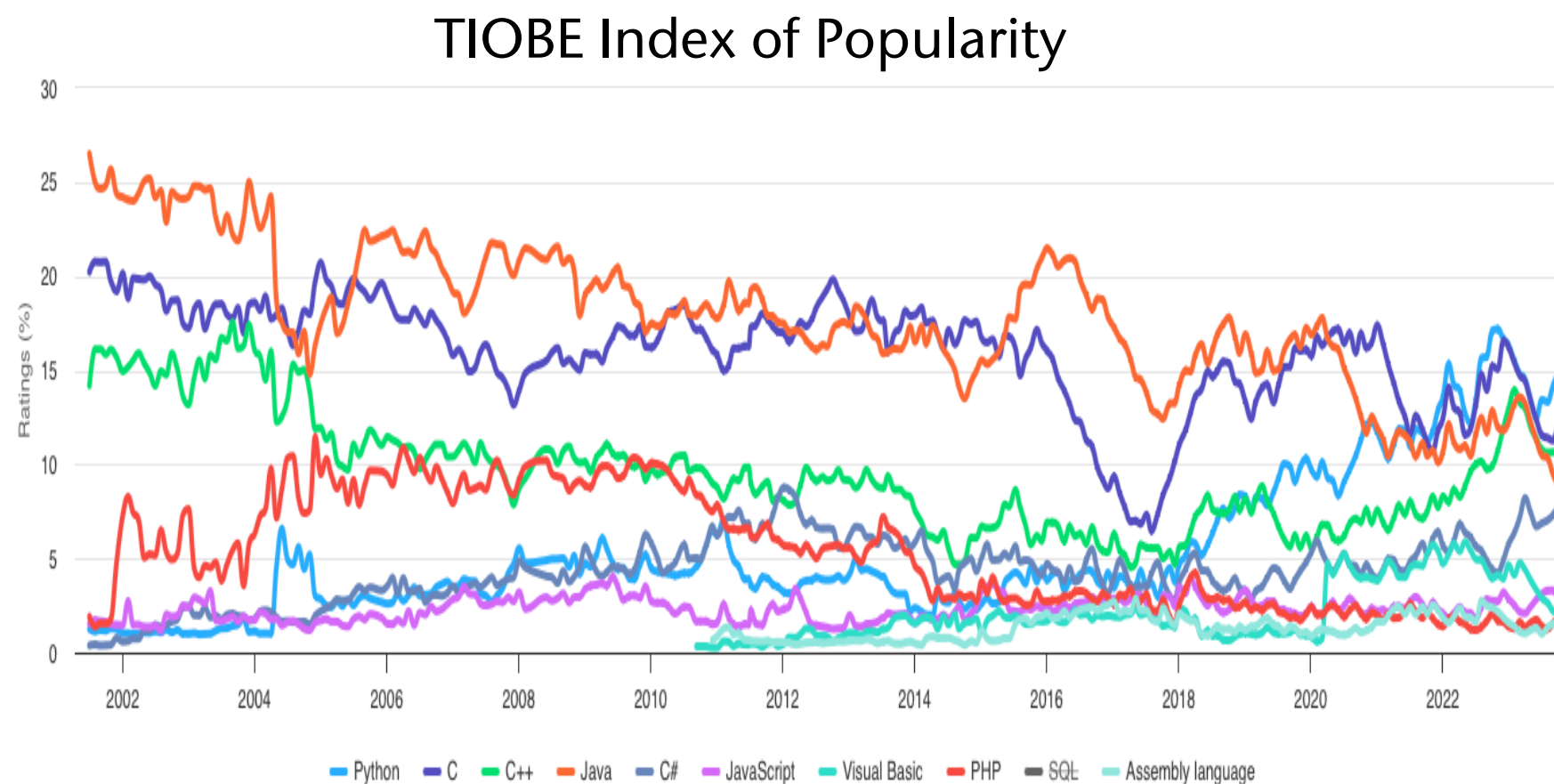


G. Zachmann  
University of Bremen, Germany  
[cgvr.cs.uni-bremen.de](http://cgvr.cs.uni-bremen.de)

# Webseite zur Computergraphik

- Alle **wichtigen Informationen** zur VL stellen wir Ihnen auch im **Internet** zur Verfügung :  
<http://cgvr.cs.uni-bremen.de/>  
→ "Teaching" → "Computergraphik"
- Folien & Übungsblätter
- Literaturhinweise, Online-Doku
- Evtl. aktuelle Meldungen
- Bitte anmelden in StudIP! (wegen Rundmails)
- Chat: <https://discord.gg/YGUZFxf> → "Introduction to Computer Graphics"
  - Achtung: ich selbst bin **nicht** in der Gruppe!

- Ein wenig Mathematik (Trigonometrie, Rechnen mit Vektoren und Matrizen)
- Für die Programmieraufgaben: ein wenig C/C++ (gute Gelegenheit, dieses wieder aufzufrischen)



# Modus der Vorlesung / Übung

- Vorlesung: Dienstag 10 ct — 14 Uhr
  - Pause?
- Übungs-/Tutoriums-Slots:
  - Montag 8 Uhr ct
  - Donnerstag 14 Uhr ct
- Abgabe der Aufgaben: sonntags 23:59
  - Über git (bitte mit Tutor abklären)
- Sprechstunde der Tutoren (in Person): Mittwoch 15-16 Uhr

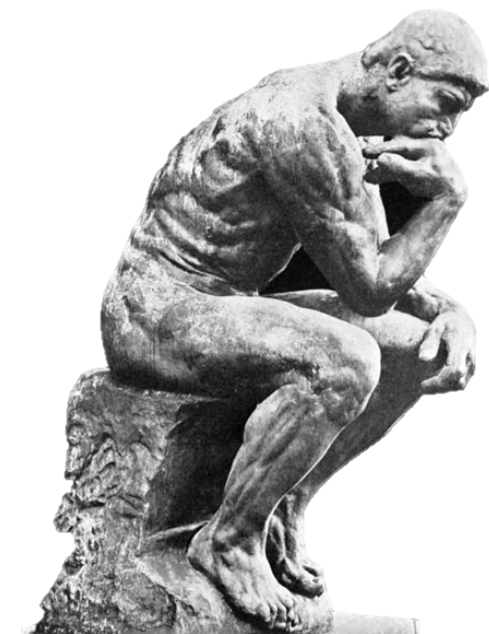
# Ziel der Vorlesung

“The mind is not a vessel to be filled,  
but a fire to be kindled.”

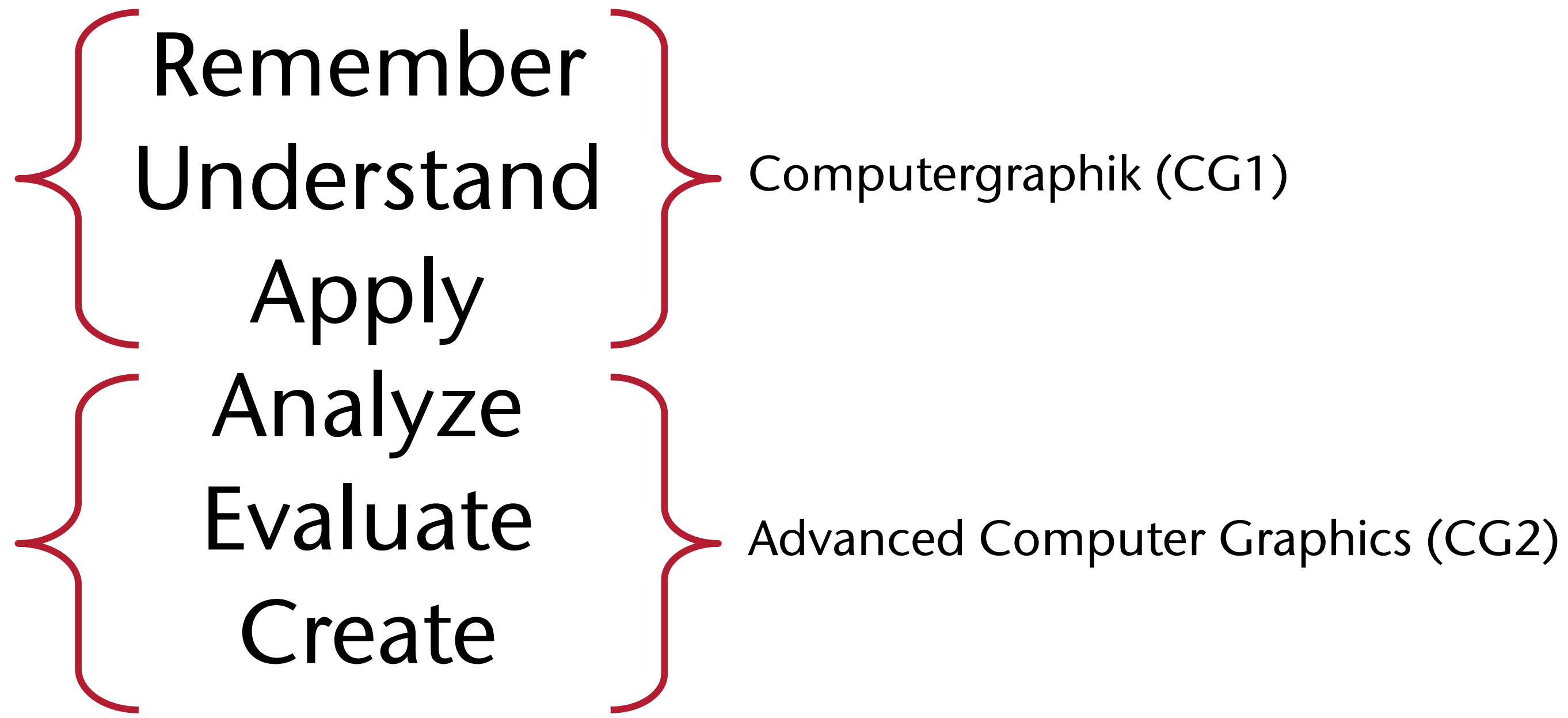
– *Plutarch*

# Ziele der Vorlesung

- Praxis: Sei in der Lage, einfache interaktive 3D Graphikprogramme zu schreiben (in OpenGL)
- Theorie: Verstehe den mathematischen Hintergrund und einige grundlegende Prinzipien moderner 3D Graphiksysteme
- Diese Vorlesung behandelt *nicht* Graphikprogramme/Modeler wie Blender, 3DStudio Max, Cinema4D, AfterEffects, Photoshop, ...



# Cognitive Processes





# Die Übungen

- Frequenz: 1 Blatt pro Woche
- Seit ChatGPT → nur noch Programmier-Aufgaben
- Praktische Aufgaben = Programmieren in C++ und OpenGL
  - Eigentlich nur "C mit Klassen"
  - Bedarf für C/C++ Refresher?
- In 4er-Gruppen ! (zur Not auch 3-er oder 5-er)
- Sprechstunde der Tutoren: Freitag nachmittag (genaue Uhrzeit TBD)

"**Vorstellung** ersetzt erst dann das **Handeln**,  
wenn jene von diesem ausreichend **Erkenntnisse** gewonnen hat."

[Piaget]



# Persönliche Vorab-Installationen

- Bitte folgende Software schon vorab auf dem eigenen Laptop installieren, falls nicht schon vorhanden
- Ein IDE eurer Wahl, z.B.
  - Mac: XCode ( <https://developer.apple.com/> oder App Store)
  - Linux: Kdevelop (?), ist wahrscheinlich schon alles drauf
  - Windows: Visual Studio
  - Minimalistisch: einfacher ASCII-Editor (vim/emacs) + Compiler
  -

# Die Prüfung

- Notenspiegel: 95% → 1.0 , 40% → 4.0

## 1. Wer die Übungsaufgaben gemacht hat:

$$\text{Gesamtnote} = \min \left\{ \frac{1}{2} \cdot \text{Übungsaufgaben} + \frac{1}{2} \cdot \text{Klausur}, \text{Klausur} \right\}$$

- Voraussetzung: beide Noten  $\geq 4.0$  (Allgemeiner Teil der Bachelorprüfungsordnungen der Universität Bremen, 2010)

## 2. Wer die Übungsaufgaben **nicht** gemacht hat:

$$\text{Gesamtnote} = \text{Klausurnote}$$

# Prüfungsmodalitäten

- Klausur ist "open notes":
  - Kein Taschenrechner, kein Handy, keine Smartwatch, etc.
  - Erlaubt ist "persönlicher Spickzettel": 3 DIN-A4-Blätter mit beliebigen Notizen, persönlich in eigener(!) Handschrift geschrieben

- **Bewertungskriterien der Programmier-Aufgaben:**

1. Gute Variablen- und Funktionsnamen

2. Genügend in-line Kommentare

3. Dokumentation der Funktion und deren Parameter (in/out, pre-/post-condition, was tut die Funktion, ...)

4. Funktionalität (Aufgabe vollständig gelöst? Bug-frei? ...)

Documentation: minimum

```
// Convert HSV color space to RGB
// input: h must be in [0,360); s,v must be in [0,1]
// output: r,g,b (out) will be in [0,1]
// Warning: no parameter range checks are done!
__device__
void HSVtoRGB( float *r, float *g, float *b,
               float h, float s, float v )
{
```

Documentation: better

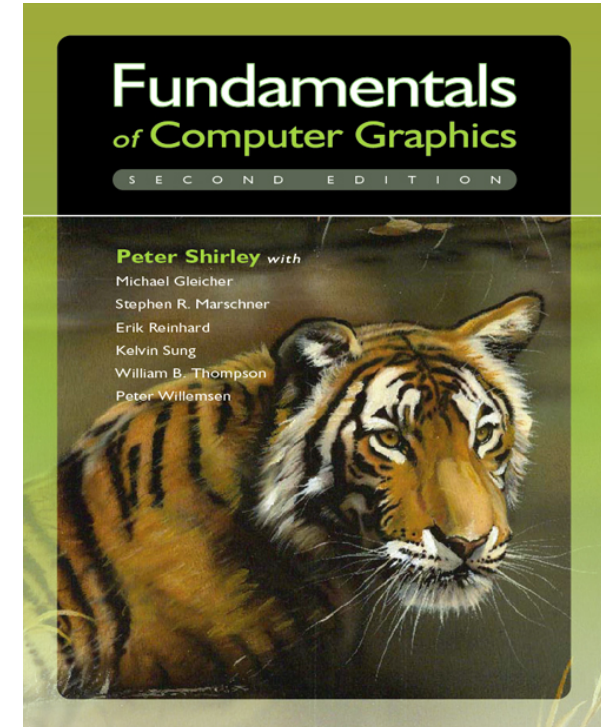
```
/** Compute point nearest to q and on an edge of SIG
*
* @param q           the current query point
* @param points      the point cloud
* @param delaunay    delaunay diagram of the point cloud
* @param pstar       NN of q (out)
* @param pstar2      neighbor of pstar (in SIG) (out)
* @param phat        point closest to q on edge (pstar,pstar2) (out)
* @param d           distance between q and phat (out)
*
* @warning
* Assumes that a SIG has been computed!!
* @bug
* Bis jetzt ist pstar2 nur NN zu pstar im Delaunay-Graph, nicht im SIG!!
**/

void nearest_on_graph( const FPoint & q, const std::vector<FPoint> & points,
                      const Proximity & proximity,
                      unsigned int * const pstar, unsigned int * const pstar2,
                      FPoint * const phat, float * const d )
{
```

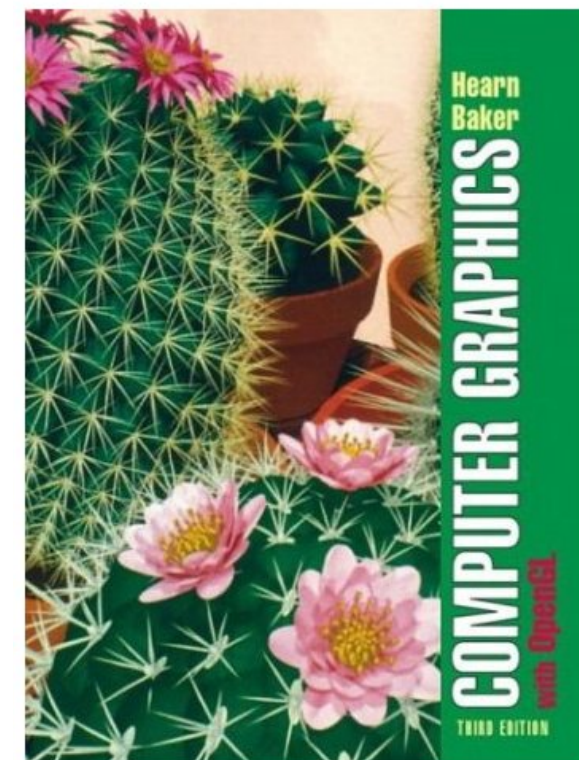
# Plagiarismus

- Ernsthafter Fall von akademischem Fehlverhalten!
- 1. Versuch: "gelbe Karte"
  - 0 Punkte für **beide** Gruppen auf das **gesamte Übungsblatt**
  - Plagiats-Eintrag in der Akte im Prüfungsamt
- 2. Versuch: "rote Karte"
  - Übungsvornote = 5.0
  - Zweite Meldung ans Prüfungsamt
- Dito für Kopieren aus dem Internet!
- Seit ChatGPT:
  - Lernt man dabei etwas???
  - Programmier-Aufgaben ... naja ... man muss als User sehr genau hinschauen ...

- Peter Shirley: *Fundamentals of Computer Graphics*.  
Francis & Taylor



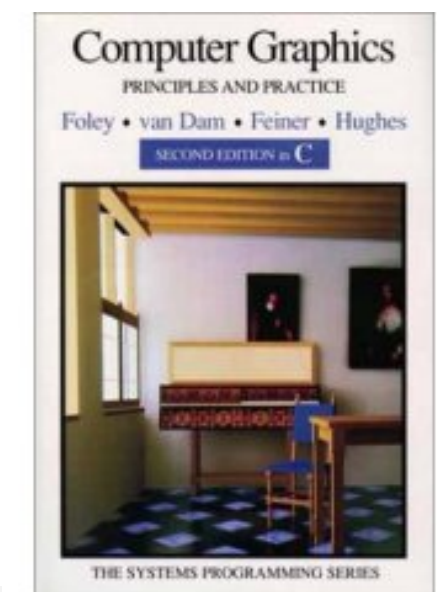
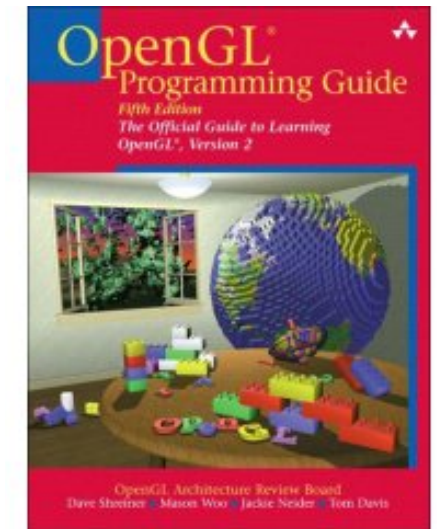
- Donald Hearn, M. Pauline Baker: *Computer Graphics with OpenGL*. Prentice Hall





# Literatur, nicht mehr ganz aktuell, aber teilweise brauchbar

- Dave Shreiner: *The OpenGL Programming Guide: The Official Guide to Learning OpenG*. Addison-Wesley
  - Auf der VL-Homepage als PDF !
  - Passwort ...
- J. L. Encarnaçã, W. Strasser, R. Klein: *Graphische Datenverarbeitung 1 und 2*. Oldenbourg, 1996
- J. Foley, A. van Dam, S. Feiner, J. Hughes: *Computer Graphics: Principles and Practice*. Addison-Wesley Professional; 2nd Edition, 1995

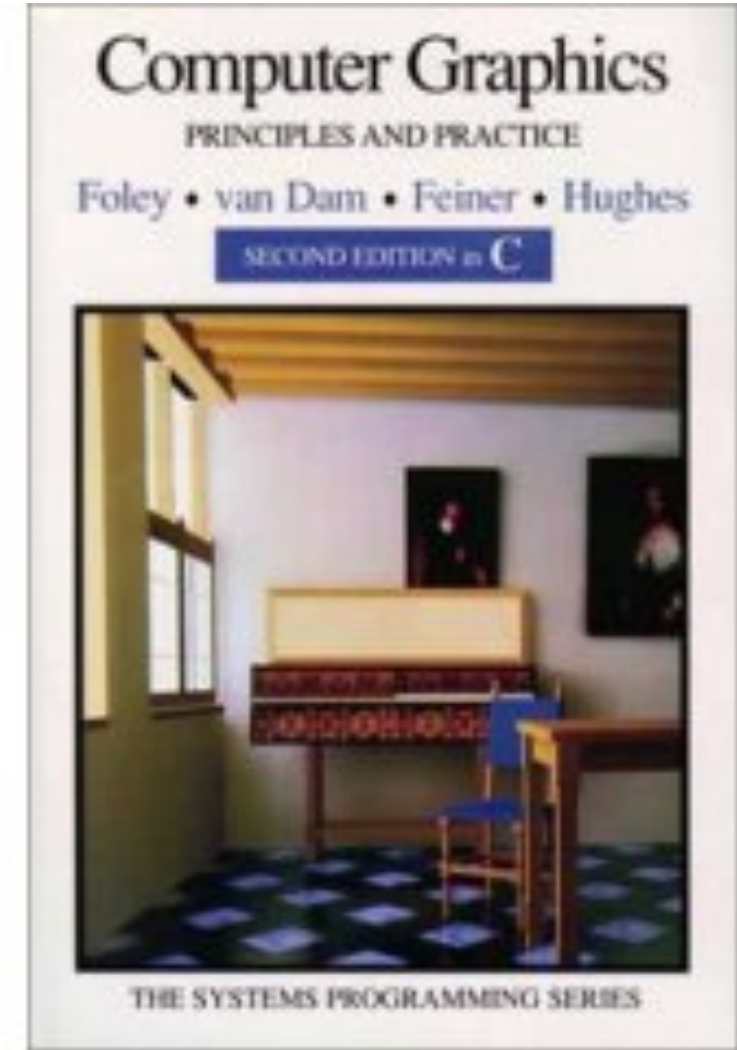




# Exkurs: Jan Vermeer



Jan (Johannes) Vermeer  
*The Music Lesson*  
 Ca. 1662-1665  
 Oil on Canvas  
 75x64 cm





# For the Creatively Inclined

## RenderMan **Woodville** Art Challenge

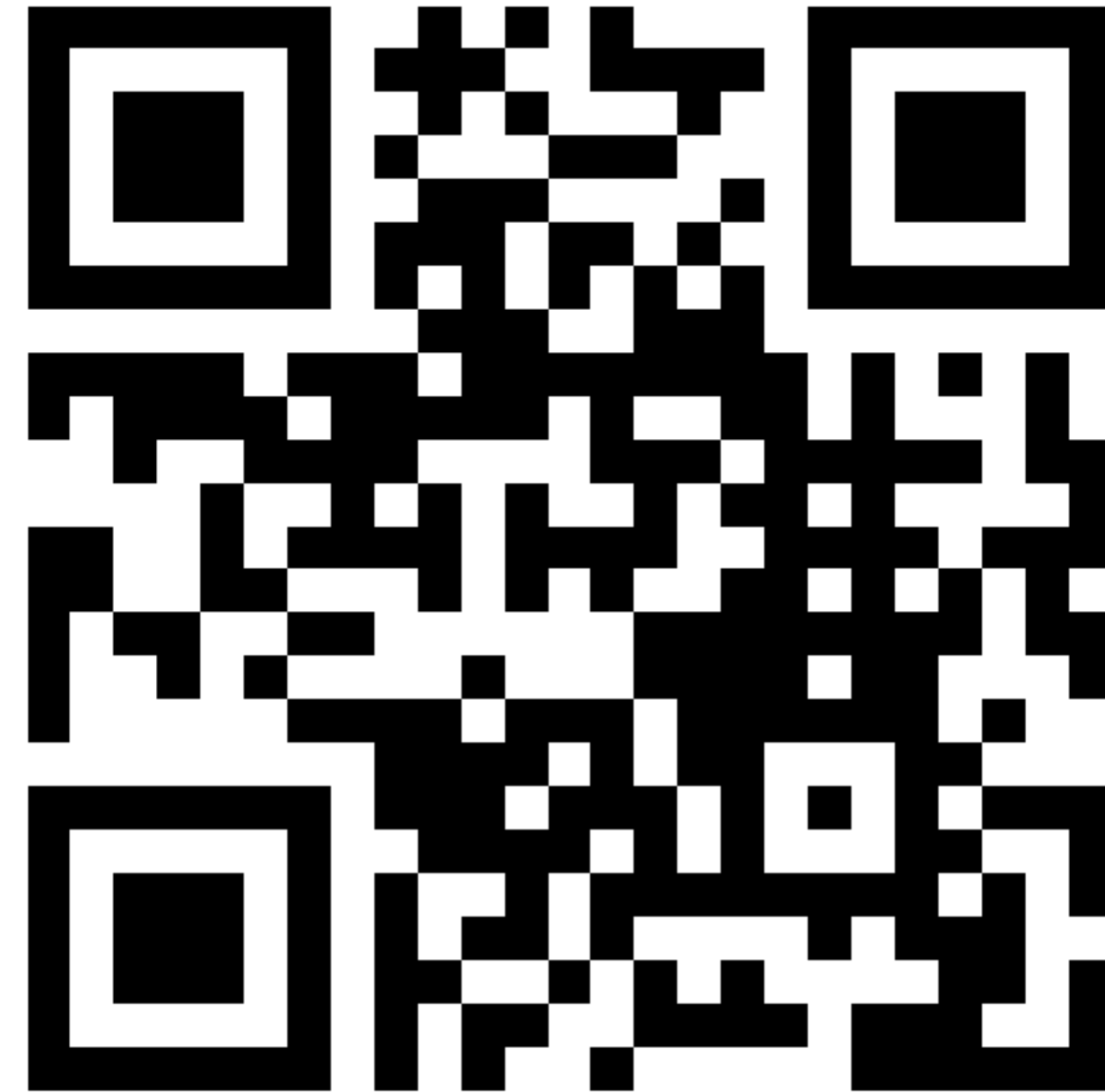


Deadlines: twice a year (spring and fall)

Get Pixar feedback, improve your skills, and win amazing prizes!

"In this "Woodville" Art Challenge, you'll get to work with a fantastic scene from Pixar's Undergraduate Program and SpeedTree, providing you with production-quality assets to showcase your shading, lighting, rendering, and compositing skills for the possibility of winning amazing prizes"

# Questions?



<https://www.menti.com/m6zmongd5r>