# Bewertungsrubrik für Systemsimulation-Projekte (Autoren):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Kriterium** | | **Experte (1)** | | | | **Begabt (2)** | **Kompetent (3)** | **Lernende (4): Projekt sollte …** | **(5)** |
| **Argument: Darlegung** | | Benützt ***prägnante***, ***präzise,*** ***leicht zugängliche Text- und Grafik***-Bausteine um Vorhaben plus Forschungs- und Null-Hypothesen darzulegen. | | | | Präsentiert Forschungsvorhaben und Hypothesen durch ***prägnante*** und ***präzise Text- und Grafik***-Bausteine. | Benützt ***prägnanten*** und ***präzisen Text*** um Forschungsvorhaben und Hypothesen zu präsentieren. | … ***Vorhaben plus Forschungs- und Null-Hypothesen leicht verständlich*** präsentieren. |  |
| **Argument: Prüfbarkeit** | | ***Forschungs- und Null-Hypothesen*** sind wohl­formuliert***:*** ihre Aussage ist ***widerlegbar***. | | | | ***Forschungs- und Null-Hypothesen*** sind ***größtenteils widerlegbar***. | ***Forschungs- und Null-Hypothese*** sind nur teilweise ***widerlegbar***. | … Null- und Forschungs-Hypo­thesen ***operationalisieren***. |  |
| **Argument: Logik** | | Das ***Forschungsvorhaben prüft konsistent und messbar*** die Korrektheit der zwei ***Hypothesen***. | | | | ***Forschungsvorhaben prüft größten­teils*** die Korrektheit der ***Hypothesen***. | ***Forschungsvorhaben prüft nur zum Teil*** die Korrektheit der ***Hypothesen***. | … ***Hinreichende Bedingungen*** für die Hypothesen klären. |  |
| **Argument: Methoden** | | ***Versuchsdesign erfüllt alle Anforderungen des Vorhabens,*** um Hypothesen zu bestätigen. | | | | ***Versuchsdesign folgt meist dem Vorhaben*** um Hypothesen zu prüfen. | ***Versuchsdesign folgt z.T. dem Vorhaben*** um Hypothesen zu prüfen. | … sicherstellen, dass ***Design die Hypothesen robust prüft***. |  |
| **Argument: Analyse** | | Präsentiert Ergebnisse so, dass Lesende daraus die ***Bestätigung der Forschungshypothese sofort nachvollziehen*** können. | | | | Präsentiert Ergebnisse so, dass die ***Bestätigung der Forschungs­hypothese nachvollziehbar*** ist. | Präsentiert einigermaßen überzeugend aus den Ergebnissen die ***Bestätigung der Forschungs­hypothese***. | … aus den Ergebnissen die ***Bestäti­gung der Hypothesen überzeugend herleiten***. |  |
| **Argument: Diskussion** | | Breitere ***Bedeutung und Relevanz*** der Rückschlüsse sind ***sofort ersichtlich***. | | | | Breitere ***Bedeutung und Relevanz*** der Rückschlüsse sind ***erkennbar***. | Breitere ***Relevanz*** der Rückschlüsse ist nur ***schwer erkennbar***. | … breite ***Relevanz*** der Arbeit ***für andere Gebiete*** darstellen. |  |
| **Argument: Software-Apparat** | | ***Software-Code erfüllt alle Anforderungen*** des Vorhabens ohne logische Lücken, prüft Be­nutzer­fehler und zeigt passende Ausgaben an. | | | | ***Code erfüllt Anforderungen*** ***ohne Fehler*** mit z.T. unpassender Ausgabe. Prüft manche Eingabefehler. | ***Code liefert korrekte Ergebnisse***, zeigt sie aber inkorrekt an. Prüft manche Eingabe- und Bereichsfehler. | … ***Versuchsanforderungen*** korrekt ***erfüllen*** und Benutzereingabe prüfen. |  |
| **Software: Darstellung** | | ***Der Code ist klar strukturiert und formatiert.*** Klare Codeblöcke, Methoden, Einrückung und Zeilenumbrüche lassen ihn leicht verstehen. | | | | ***Code ist einfach zu folgen*** mit kleinen Formatierungs-, Einrückungs- und Klammer-Fehlern. | ***Code ist meistens einfach zu folgen***, aber Auslegung erschwert das Verstehen. | … durch klare Auslegung den ***Softwareablauf*** für Uneinge­weihte ***deutlich darlegen***. |  |
| **Software: Kohärenz** | | ***Gestaltung, Benennung und*** ***Kommentare*** machen stets ***deutlich die verbindende Absicht hinter allen*** Modulen und ***Code-Komponenten***. | | | | ***Kommentare drücken die Absicht der Komponenten aus***, ***Benennung*** ist aber etwas ***unhandlich***. | ***Kommentare drücken Absichten aus***, aber ***Benennungen deuten*** ***nicht*** offensichtlich ***auf ihren Zweck***. | … Benennung und Gestaltung zur ***klaren Kommunikation der Code-Absicht*** einsetzen. |  |
| **Software: Typisierung** | | ***Verwendet*** primitive und benutzerdefinierte ***Typen*** ***effizient und korrekt*** um Code sauber und konzeptionell zu strukturieren. | | | | ***Verwendet Typen passend*** um Code effizient und nachvollziehbar zu strukturieren. | ***Verwendet Typen passend***, aber auf konzeptionell unsaubere Strukturen abgestimmt. | … ***Typen als Werkzeug zum konzeptionellen Strukturieren*** des Designs verwenden. |  |
| **Software: Kontrolle** | | ***Kontrollstrukturen (Rekursion, Fal­tung, Itera­tion) fördern wirksam*** das Algorithmendesign. | | | | ***Setzt Kontrollstrukturen ein meist passend*** zum algorithmischen Zweck. | ***Setzt Kontrollstrukturen passend ein***, liefert aber inkorrekte Algorithmen. | … ***Kontrollstrukturen als Design-Tool*** einsetzen. |  |
| **Software: Modularität** | | ***Module***, Methoden und Schnittstellen besitzen und ***kapseln*** eine klare ***leicht verständliche*** ***Absicht und Verantwortung***, um Redundanz und Fehlerausbreitung zu minimieren. | | | | ***Modularität ist verständlich und klar***, durchlässige Partitionierung von Verantwortlichkeiten lässt allerdings Ausbreitung oder Redundanz zu. | ***Modularität ist nachvollziehbar***, Code lässt allerdings Fehlerausbreitung und Redundanz zu, indem er den ***globalen Zugriff auf Daten*** verwendet. | … ***Module***, Methoden und Schnittstellen ***aufgrund ihrer Absicht und Verantwortung*** sauber ***partitionieren***. |  |
| **Software: Effizienz** | | ***Code ist sehr effizient***, minimiert Operation-Komplexität und zwischenspeichert mehrfach verwendete Daten, ***ohne Lesbarkeit zu opfern***. | | | | ***Code ist effizient***, ***ohne Einbußen von Lesbarkeit*** und Verständlichkeit. | ***Code ist effizient mit wenig Verlust an Lesbarkeit*** und Verständlichkeit. | … Kommunikation zwischen Code-Komponenten ***effizient und lesbar*** gestalten. |  |
| **Total =** |  | | **/ 13 =** |  |  |  |  |  |  |