Didaktisches Konzept "COOL Informatics"

COoperative, COmputer-Science-supported & Cross-curricular Open Learning (COOL)

Die didaktische Gestaltung der Stationen, Aktivitäten und Materialien des COOL Labs basiert auf dem Unterrichtskonzept „COOL Informatics" (Cooperative, Computer-Science-supported and Cross-curricular Open Learning) auf, das auf der Grundlage von neurodidaktischen Prinzipien sowie der Analyse bereits bekannter effektiver Lehr- und Lernformen entwickelt wurde. Dieses Konzept wurde zunächst für das Fach Informatik konzipiert, ist aber inhaltlich flexibel und kann problemlos für alle Fächer und Altersgruppen adaptiert werden. Es eignet sich nicht nur für das COOL Lab sowie die LIT Open Labs[1] der JKU, sondern kann auch in Bachelor- oder Masterkursen verschiedener Fächer eingesetzt werden. Die folgende Tabelle zeigt einen Überblick über die vier Grundprinzipien von „COOL Informatics" (*discovery, cooperation, individuality*, *activity)*, eine Auswahl passender Lehr- und Lernmethoden und einige der zugrundeliegenden neurodidaktischen Prinzipien. [1]

"COOL Informatics" – Overview		
Principle	**Teaching and learning methods**	**Neurodidactical basis**
1. Discovery	Solution-based learning Step-by-step instructions & tasks Video tutorials, Observational learning Learning with all senses	Pattern recognition Mirror neurons Individual learning rhythm modality / multimedia effect
2. Cooperation	Team and group work Peer tutoring and –teaching Pair programming Cross-curricular learning Project-based learning	"A joy (=knowledge) shared is a joy (=knowledge) doubled." Recall = re-storage in memory Integrating individual needs, talents, competences, practical relevance
3. Individuality	Competence-based learning Questioning Self-organized learning with compulsory and optional tasks	Connecting new information to previous knowledge. Considering individual interests, needs, tasks, methods, learning rhythm
4. Activity	Hands-on, Minds-on Learning by doing Learning by animation, simulation by playing and designing games (creative learning)	Knowledge must be newly created (constructed) by each learner (= constructivism) Learning is an active process (=progressive education, e.g. Montessori)

Das Konzept wurde ab 2012 in der Lehrveranstaltung „Einführung in die strukturierte und objektbasierte Programmierung" an der Universität Klagenfurt eingesetzt und 2013 für den Ars Docendi – Staatspreis für exzellente Lehre an Universitäten vorgeschlagen. Die Evaluationsergebnisse sind vielversprechend: Sowohl in der Pilotphase 2012 als auch in drei Versuchsgruppen im Jahr darauf konnten

- die Klausurergebnisse insgesamt signifikant verbessert,
- der zuvor beobachteten Gender-Gap ausgeglichen,
- die Drop-out Rate gesenkt und
- die Zufriedenheit der Studierenden im Kurs erhöht werden. [2, 3]

Literatur

[1] Sabitzer, B. (2014). A Neurodidactical Approach to Cooperative and Cross-curricular Open Learning: "COOL Informatics". (Habilitation thesis). Alpen-Adria-Universität, Klagenfurt.

[2] Sabitzer, B.; Pasterk, S. (2014). Brain-based Programming Continued. In: Proceedings of 2014 IEEE Frontiers in Education Conference, Oct. 2014, Madrid, Spain. IEEE Computer Society Press, 2014, pp. 1495 - 1500.

[3] Sabitzer B., Strutzmann S. (2013): Brain-based Programming. In: R. Shehab, J. Sluss, D. Trytten (Hrsg.): Energizing our Future. Proceedings of 2013 Frontiers in Education Conference, October 2013, Oklahoma City, USA. Los Alamitos (CA): IEEE Computer Society Press, 2013, 1163-1170.

[1] Siehe auch Konzept LIT Open Lab vom 20.7.2017

COOL Informatics in der Praxis

Unterrichtsdesign

Neue Lerninhalte werden anhand von Musterbeispielen und -lösungen, Schritt-für-Schritt-Aufgaben, etc. (*discovery*) eingeführt und/oder in kooperativen Lernsettings (*cooperation*) erarbeitet. Bei der Erarbeitung, Wiederholung, Reproduktion und Anwendung von (Fach)Wissen und Kompetenzen werden individuelle Bedürfnisse, Interessen, Fragen und/oder Talente berücksichtigt (z.B. bei der Auswahl von Themen, Materialien, Lernmethoden, Technologien, Lernsettings, Aufgabentypen etc.) (*individuality*). Wesentlich ist, dass sich Lernende aktiv einbringen (*activity*), z.B. indem sie anderen etwas erklären (*cooperation, peer-tutoring, peer-teaching*), Musterlösungen oder kreative Lernprodukte erstellen (*individuality*) oder anhand von Musterbeispielen Regeln und Strukturen entdecken (*discovery*). Das Besondere an der Umsetzung von COOL Informatics ist die Einbindung verschiedenster Technologien (*instructional technology*) wie z.B. BeeBots oder auch Smartphones (im Sinne von BYOD) und informatischen Konzepten und Techniken, wie z.B. Modellierung oder Aussagenlogik, als Tools zur Unterstützung von Lehren und Lernen in allen Fächern. Das Konzept ist sehr flexibel und kann an die Bedürfnisse aller Altersgruppen, Schultypen und Fächer angepasst werden.

Einführung in die strukturierte und objektbasierte Programmierung (Bachelor-LV)

Kompetenzniveau berücksichtigen

Einteilung der Studierenden in kleine Teams und zusätzlich in drei Kompetenzniveaus, wobei die Zuteilung nicht fix sein muss, sondern je nach Thema und Vorkenntnissen variieren kann (z.B. Einsteiger, die ein neues Thema sofort verstanden haben, agieren auch als Peer-TutorInnen

- *Profis* mit guten Vorkenntnissen = Peer-Tutoren, Peer-Teacher, die Fragen beantworten, Themen erklären, Lösungen kontrollieren, Aufgaben und Musterlösungen selbst entwickeln etc.
- *Amateure* mit verschiedenen geringeren Vorkenntnissen warden, je nach Thema, teilweise als Peer-TutorInnen eingesetzt
- *Einsteiger* ohne Vorkenntnisse können je nach Möglichkeit ebenfalls als Peer-TutorInnen fungieren.

LV-Phasen

- (*Vortrag* bei Bedarf max. 20 min)
- *Fragerunde* (ca. 10 min) individuelle Fragemöglichkeiten, Vorwissen berücksichtigen (individuality, cooperation durch Talente-Tausch und Peer-Tutoring)
- *Entdecken* (10-20 min) (individuality, cooperation)
- Aufgaben für entdeckendes Lernen (discovery), ev. in Kleingruppen, Musterbeispiele, Aufgaben mit Musterlösungen, Schritt-für-Schritt-Aufgaben, Puzzles, Erklärvideos
- *Labor* (restliche Zeit) Pair-Programming (cooperation, activity)

Aufgaben

- Entdeckendes Lernen – Mustererkennung:
  - Leseecken (Programmcode, Puzzle, Lückentext, Fehlersuche)
  - Schritt-für-Schritt-Anleitungen mit Lösungen
  - Aufgabenstellungen mit Musterlösung
- Kompetenzorientiertes Üben – Vorwissen
  - Miniübungen
- Programmieren – Selbständiges Üben
  - Kleine, vollständige Programme
  - Teile bzw. Unterprogramme für ein Semesterthema

COOLer Fremdsprachenunterricht (Sekundarstufe I und II)

z.B.: Designing Games in and for Primary and Secondary Education, Modeling in Language Lessons,

Informatik-Werkstatt (Primarstufe, Sekundarstufe I)

http://informatikwerkstatt.aau.at