Programujeme, a co dál?

Dan Lessner

ksvi.mff.cuni.cz/ucebnice
ucime-informatiku.blogspot.cz

Učebnice informatiky

Přihlaste se

Informatika pro každého

Do této učebnice průběžně přibývá obsah. Současný stav je už ale dobře použitelný a byla by škoda čekat, až bude publikace zcela hotová. Pravděpodobně bude pořád co zlepšovat. Na stránce Kontakt nám můžete dát vědět, pokud si všimnete nějakých chyb či jiných nedostatků.

učebnice
Informatika pro každého

Hlavní strana
Obsah
Kapitoly
Informace
Algoritmus
Nástroje
Další odkazy

Zobrazit či skrýt:
Rozšíření
Učitel
Nápovědy
Řešení
Odkaz na tento pohled

Učebnici podporují:

Google

Informatika

Možná tě to překvapí, ale počítač je pro informatiku pouhý (byť skvělý) nástroj. Pronikni do základních principů **efektivního zpracování informací**. Kromě **porozumění moderním technologiím** se ti otevře nová paleta možností, jak **uvažovat o světě**, **racionálně se rozhodovat** a **efektivně řešit komplexní problémy**. Čti dál: Co je informatika?

Pro každého

Učebnice pokrývá základy informatiky tak, aby je dokázal využít každý student gymnázia. Tedy **kdokoliv, kdo zvládl látku základní školy, chce přijít světu na kloub a nevzdává se snadno**. Nejde o to, aby se znenadání každý stal informatikem. Informatika a její principy se ale uplatňují prakticky všude. Nevyužívat je znamená **mrhat lidským potenciálem**. I proto se informatika stává vedle ostatních přírodních věd **součástí všeobecného rozhledu**.

Učebnice

Představuje ucelený program s **výkladem**, **řešenými příklady**, **úlohami** a množstvím **rozšiřujících materiálů**. Bohatě pokryje dvě hodiny výuky týdně po dobu jednoho roku. Má všechny funkce běžné učebnice: lze ji použít přímo v hodinách informatiky, k samostatnému opakování a procvičení před prověrkou, k „dohonění" látky po nemoci i k inspiraci a nasměrování k dalším zdrojům.

Samozřejmě nestačí jen číst. Studium informatiky podle této učebnice znamená především **zapojení vlastní hlavy** a vynaložení odpovídajícího úsilí.

Chci **studovat** informatiku

Chci **učit** informatiku

Úvod

1. Předmluva pro studenty
2. Předmluva pro učitele

Přípravy

ksvi.mff.cuni.cz/ucebnice

Patří programování do všeobecného vzdělávání?

- Potřebujeme snad z každého vychovat programátora?
- Takže proč tedy?
  - Všeobecný přehled — jak funguje svět kolem nás
  - Rozvoj informatického myšlení

- Jak těchto cílů dosáhnout? Jak se odlišuje výuka programátorů?

Obecné empiricky ověřené principy

Prolínání

- Slovní a vizuální popis
- Abstraktní koncepty a jejich konkrétní příklady
- Řešené a neřešené úlohy
- Opakování „probrané látky“

Zpětná vazba

- Otázky ověřující pochopení, podněcující přemýšlení
- Testíky na podporu zapamatování

„Naprogramuj ověření trojúhelníkové nerovnosti, ...“

- Proč?!
- Co to je?!

Kolik jakých kroků žák při řešení úlohy vykoná?

„Naprogramuj ověření trojúhelníkové nerovnosti, ...“

Jsou i jiné typy úloh:

- Zaměření na různé fáze postupu řešení
- Zaměření na různou práci s programem
- Doplňování, upravování
- Hledání chyb, opravování
- Porovnávání a hodnocení programů z různých hledisek
- Čtení (Co dělá tento program? Co se stane, když...?)

Parsonsovy úlohy

Drag from here

```
REPEAT ?? TIMES
ENDREPEAT
left(120)
forward(100)
```

Construct your solution here

Model Drawing

Your Code Drawing

Reset

Feedback

(http://js-parsons.github.io)

Multimodalita

1) Vysvětlím while-cyklus
2) Ukážu příklad
3) Zadám úlohu

Naučil jsem while cyklus?

Multimodalita

Práce se stejným konceptem v různých situacích a kontextech

- Matematické (číselné) výpočty
- Textové řetězce
- Obrazová, zvuková data
- Objekty v nějaké simulaci, hře

Motivace: potřeba pro vyřešení problému,
nikoliv pořadí v referenční příručce

Metakognice

Překračování úrovní je pro informatiku charakteristické

řešení problému
⇒ řešení všech podobných problémů

zapamatování a provádění algoritmu
⇒ vytvoření nejvhodnějšího algoritmu

přemýšlení o problému
⇒ **přemýšlení o procesu vlastního přemýšlení**

Metakognice

- Vyučující programuje „u tabule“
  - Přemýšlí nahlas!
- Vyučující programuje bez tabule, žáci píší

- Komentování zdrojového kódu
- Subgoal labeling

Metakognice – řešené příklady

- Proces, nejen výsledek!
- Hodně
- Proložit s úlohami pro žáky

Metakognice – práce ve dvojici

- Snazší překonání překážek
- Snazší odhalení chyb (ještě před spuštěním)
- Jeden "kóduje", druhý má čas přemýšlet
  - Pozor na střídání rolí

Nutnost spolu o programu mluvit, nahlas přemýšlet a argumentovat.

Samostatné objevování ve výuce programování

- Náročný a motivující problém
- Žádný postup řešení
- Možnost různých výsledků
- Poznání konstruuje žák na základě vlastní přímé zkušenosti

Samostatné objevování ve výuce programování

Výhody

- Podpora tvořivosti žáků
- Trvalejší a hlubší poznání
- Práce na vyšší kognitivní úrovni

Rizika

- Nedostatek času
- Objev nebude objeven
- Objev zapadne v detailech
- Pocity zoufalství ze selhání a vlastní neschopnosti

K přemýšlení na léto:
Kirschner, Sweller, Clark (2006): *Why Minimal Guidance During Instruction Does Not Work...* *(a* reakce na tento článek)

Pojem: Cognitive (over)load

Teorie:

- Učení je nějaká změna v dlouhodobé paměti.
- Do té přechází informace opakovaně zpracovávané v krátkodobé (pracovní) paměti.
- Přetížení pracovní paměti brání procesu učení.

Při programování a práci na otevřených problémech velmi snadné!

- Přísná syntaxe
- Netriviální sémantika
- Bohaté IDE
- Složitý postup před spuštěním programu
- Komplexní nestrukturované zadání
- Množství nových informací a prog. konceptů ke zpracování
- Překládání ze "svými slovy" do zdrojového kódu

Samostatné objevování ve výuce programování

Předpoklady (ne)fungování

- Vhodně nastavené vzdělávací cíle
- Přiměřená náročnost problému
- Přiměřená časové dotace
- Přiměřeně strukturovaný postup

Komu a kdy to tedy funguje?

Pojem: Scaffolding

Scaffolding („lešení“) žákům zpřístupňuje řešení obtížnějších (a snad zajímavějších) projektů

- Hrubý návrh postupu (kroků, fází) řešení
- Poskytnutí řešení podobného případu
- Poskytnutí dílčích řešení
- Vysvětlení neznámých či složitých pojmů
- Autocorrect, autocomplete
- Ladicí výstupy, krokování

Předcházení bezmoci a frustraci

- Atmosféra: Dělání a odstraňování chyb je běžnou součástí práce informatika.
- Nejen úlohy „naprogramuj“
- Úlohou učitele není opravovat žákovské algoritmy a programy.
  - Můžeme ale nabídnout postupy a strategie,jak si poradit samostatně.
- Při programování myslíme nahlas (jako učitelé i žáci ve dvojicích).
- Počítač je jen stroj, nedělá podlé naschvály.

csteachingtips.org

Search | Tags | Source | Sort by | Order

Pair Programming [T&LS] ×

Choose some options | New | Desc | Reset | Apply

- Address Misconceptions About the Field of CS
- Algorithms and Design
- AP A
- AP Computer Science Principles (CSP)
- Beauty and Joy of Computing (BJC)
- C
- C#
- C++

Pass out the grading rubric w ssignments so students understand how they will be assessed.

Like (5 Likes)

Specify how students can re u to let them know you are interested in their opinion and to create a better learning environment o express themselves.

Like (0 Likes)

Have your students practice Multiple Choice questions as preparation for the AP exam to facilitate information recall.

Like (3 Likes)

Don't worry about trying to match the modality of your instruction to students' "learning styles" since the importance of learning styles is a myth.

Like (3 Likes)

Závěr

Co **přesně** výukou programování sledujeme?

(Na to pozor i při aplikaci empirických studií.)

Díky za pozornost

Dan Lessner

lessner@ksvi.mff.cuni.cz
ksvi.mff.cuni.cz/ucebnice
ucime-informatiku.blogspot.cz

Google