2.9 Programování: Pomoc u vybraných úkolů

Naučte se programovat, aniž by kód psala AI

Předmět: Programování · **Ročník:** 9. ročník (14–15 let) · **Délka:** 1 dvouhodinový blok (90 min) · **AI:** Podpora **S** AI

Kompetence v oblasti AI (rámec DUAL.AI.TEACHer) – úroveň učitele	**Předmětové vzdělávací cíle – úroveň žáka**
AF-TL-2b *(Základy a aplikace AI × Výuka a učení, úroveň 2 – Reflektivní implementace)*: „Pedagogové dokážou uspořádat prvky výuky, které zapojují používání AI s jasnými pedagogickými rolemi a strategiemi pro ověřování výstupů.“ **AF-FC-2b** *(Základy a aplikace AI × Podpora digitální kompetence žáků (v oblasti AI), úroveň 2 – Reflektivní implementace)*: „Učitelé mohou zavést takové pokyny, které žákům pomohou rozvíjet praktické dovednosti v oblasti umělé inteligence, a to přizpůsobením úkolů jejich dosavadním znalostem a dostupným nástrojům umělé inteligence.“	Na konci hodiny žáci dokážou: • využít AI k opakujícím se nebo podpůrným úkolům při programování, jako je odsazování kódu nebo psaní komentářů • požádat model AI o návrhy na vylepšení kódu, aniž by mu samotný kód předali, aby si žáci mohli sami ověřit své znalosti • kriticky hodnotit model AI, když se ho ptají, jak by daný kód optimalizoval

> **Klíčové poselství:** Studenti musí znát základy programování, pokud chtějí vyhodnotit výsledky poskytnuté modelem. Studenti, kteří se teprve začínají učit programovat, by proto měli modely umělé inteligence využívat k provádění podpůrných a opakujících se úkolů souvisejících s programováním, nikoli však k vytváření kódu. Model mohou také využít k tomu, aby si vyžádali návrhy na možné vylepšení kódu, a tak si otestovat své programátorské schopnosti.



Obr. 1: „Naučte se programovat, aniž by kód psala AI“ (obrázek vygenerovaný pomocí AI, Google Gemini 3.6 Thinking)

Obsah

Výuka programování se mění v době, kdy systémy AI dokážou kód generovat, vysvětlovat i revidovat. Tyto nástroje sice nabízejí nové příležitosti k učení, přinášejí však i důležitou pedagogickou výzvu: žáci potřebují dostatečné znalosti programování, aby dokázali posoudit, zda je návrh vygenerovaný AI správný, efektivní, nebo vůbec relevantní pro zadaný úkol. Bez základního porozumění programovacím konceptům mohou žáci přijmout chybná řešení, přehlédnout chyby nebo nerozpoznat lepší alternativy. Proto zůstává osvojení základů programování zásadní i tehdy, jsou-li nástroje AI k dispozici.

V úvodních kurzech programování by AI měla mít roli nástroje na podporu učení, nikoli náhradního programátora. Žáci mohou AI účelně využívat k opakujícím se nebo rutinním úkolům, které nesnižují kognitivní úsilí spojené s učením se programovat. Příkladem je formátování a odsazování kódu, generování dokumentačních komentářů, vysvětlování chybových hlášení, shrnování pravidel syntaxe nebo pomoc s pochopením účelu konkrétních programových konstrukcí. Tato využití umožňují žákům soustředit pozornost na logiku, algoritmy a postupy řešení problémů, které tvoří jádro programátorské kompetence.

Dalším přínosným využitím AI je revize kódu. Místo toho, aby žáci model AI požádali o napsání řešení, napíšou nejprve vlastní kód a poté model požádají, aby určil možná vylepšení. AI může navrhnout srozumitelnější názvy proměnných, lepší uspořádání kódu, omezení redundance, lepší čitelnost nebo alternativní přístupy, které žák samostatně vyhodnotí. Tím se AI mění v partnera pro

zpětnou vazbu: žáci mohou porovnat vlastní úvahy s vnějšími návrhy a posoudit silné a slabé stránky svých řešení.

Zpětnou vazbu vygenerovanou AI by však žáci neměli přijímat nekriticky. Modely mohou doporučit změny, které jsou zbytečné, neefektivní nebo vycházejí z chybných předpokladů o účelu kódu. Ústředním cílem této hodiny je proto rozvíjet schopnost žáků kriticky hodnotit návrhy AI. Žáci se naučí ptát se modelu AI nejen na to, jaká vylepšení navrhuje, ale také proč tyto změny považuje za přínosné. Porovnáním argumentace modelu s osvědčenými programátorskými principy mohou žáci posoudit, zda navrhovaná optimalizace kód skutečně zlepšuje, nebo zda jde jen o jiné návrhové rozhodnutí.

Tato hodina pojímá AI jako nástroj, který podporuje reflexi, sebehodnocení a zvyšování kvality kódu, nikoli jeho generování. Žáci si procvičují psaní vlastních programů, využívání AI k podpůrným úkolům, vyžádání zpětné vazby ke své práci a kritickou analýzu kvality odpovědí, které dostanou. Konečným cílem je pomoci žákům stát se jak lepšími programátory, tak poučenějšími uživateli systémů AI.

Plán hodiny

Fáze	Čas	Aktivita
Úvod	10 min	Učitel představí dvě krátká řešení téhož problému – jedno napsané žákem a jedno vygenerované modelem AI. Žáci diskutují o tom, které řešení působí srozumitelněji, efektivněji nebo je snazší na pochopení.
Vstup	10 min	Učitel představí vhodná využití AI ve výuce programování a zdůrazní rozdíl mezi podpůrnými úkoly, revizí kódu a generováním kódu.
Zkoumání	15 min	Žáci pracují ve dvojicích na malém programátorském cvičení a určují, které úkoly by bylo vhodné svěřit asistentovi AI a které mají zůstat jejich odpovědností.
Cyklus vylepšování	35 min	Žáci napíšou vlastní řešení a projdou 2 až 3 kola cyklu napiš → zeptej se → vyhodnoť → uprav, přičemž AI využívají pouze k podpůrným úkolům a návrhům na vylepšení.
Reflexe	20 min	Diskuse s celou třídou o užitečnosti a omezeních zpětné vazby od AI se zaměřením na to, jak žáci určovali, zda jsou návrhy přínosné, nebo ne.

Vodicí otázky: postupné zvyšování složitosti

Žáci projdou tolik kol, kolik během 35minutového cyklu vylepšování stihnou. Každé kolo má stejné čtyři kroky: Napiš → Zeptej se → Vyhodnoť → Uprav

1. **Kolo 1 — Využití AI k podpůrným úkolům**
   - Napiš: Samostatně vypracuj krátké programátorské cvičení.
   - Zeptej se: Požádej AI o podporu u úkolu, který se netýká samotného programování, například:
     - zlepšení odsazení,
     - vygenerování komentářů,
     - vysvětlení chybového hlášení,
     - popis toho, co dělá určitá funkce.
   - Vyhodnoť: Popisuje vysvětlení od AI kód přesně? Jsou vygenerované komentáře užitečné a srozumitelné?
   - Uprav: Zapracuj užitečné změny, ale dbej na to, aby logika programu zůstala výhradně tvým dílem.
2. **Kolo 2 — Žádost o návrhy na vylepšení**
   - Napiš: Projdi si svůj hotový program a najdi části, které by bylo možné vylepšit.
   - Zeptej se: Požádej o návrhy na vylepšení, aniž bys AI nechal kód přepsat. Například:

     > „Navrhni tři způsoby, jak zlepšit čitelnost a udržovatelnost tohoto programu, ale neposkytuj náhradní kód.“

   - Vyhodnoť: Které návrhy se zdají přínosné? Které návrhy by na program měly jen malý dopad?
   - Uprav: Proveď vylepšení, se kterými souhlasíš, a otestuj, zda se program stále chová správně.
2. **Kolo 3 — Kritické hodnocení rad k optimalizaci**
   - Napiš: Vyber část svého kódu, která provádí opakující se úkol nebo používá cykly.
   - Zeptej se: Požádej o návrhy na optimalizaci a vyžádej si, aby AI svá doporučení zdůvodnila.

     Ukázkový prompt:

     > „Vysvětli, jak by bylo možné tento kód optimalizovat a proč by každá navrhovaná optimalizace zlepšila výkon nebo čitelnost. Neposkytuj optimalizovaný kód.“

   - Vyhodnoť: Jsou navrhované optimalizace skutečně přínosné? Které z nich jsou podloženy jasnou argumentací? Které se jeví jako zbytečné nebo těžko zdůvodnitelné?
   - Uprav: Rozhodni, která doporučení uplatníš, a své rozhodnutí zdůvodni.

Reflektivní otázky:

- Které návrhy AI byly pro vylepšení tvého kódu nejužitečnější?
- Poskytla AI nějaké návrhy, se kterými jsi nesouhlasil? Proč?
- Jak jsi určil, zda se optimalizace vyplatí?
- Kdy je AI během programování nejužitečnější?

- Které programátorské úkoly by měl stále vykonávat žák sám?
- Proč jsou znalosti programování nezbytné pro hodnocení zpětné vazby vygenerované AI?
- Jak se mohou žáci při učení programování vyhnout přílišné závislosti na nástrojích AI?

Závěrečná otázka (pro fázi reflexe):

Kolik znalostí programování je nutných k tomu, aby člověk posoudil, zda návrh AI program skutečně zlepšuje, a jaká rizika vznikají, když se uživatelé spoléhají na rady vygenerované AI, aniž by kódu sami rozuměli?

Materiály

- Jeden notebook na dvojici žáků s programovacím prostředím
- Přístup k chatovacímu asistentovi AI nebo lokálně provozovanému jazykovému modelu
- Krátké programátorské cvičení odpovídající úrovni zkušeností žáků
- Pracovní list podle cyklu napiš → zeptej se → vyhodnoť → uprav
- Reflektivní otázky zaměřené na gramotnost v oblasti AI a kritické hodnocení
- Volitelně ukázky kódu ilustrující dobrou a špatnou programátorskou praxi

---

Revision #6
Created 2026-09-17 15:56:31 UTC by Alexander
Updated 2026-10-29 14:15:39 UTC by Jannik