Párové programování s AI v porovnání s lidským párovým programováním

Softwarové inženýrství

Marie Kleckerová

13. 5. 2025

Úvod do tématu

Párové programování je technika vývoje softwaru, při níž dva programátoři spolupracují na jednom počítači. Tradičně pracují na jednom počítači, čím dal častěji ale i vzdáleně. Nejčastěji jeden z programátorů, v roli *řidiče* (driver), píše kód, a *navigátor* (navigator) sleduje každý řádek kódu, komentuje ho a navrhuje směr dalšího postupu práce. Existuje ale více variant rolí, od rovnocenné spolupráce po střídání rolí v pravidelných intervalech. Díky této spolupráci si programátoři předávají vlastnosti a dochází ke zlepšení kvality kódu a rychlejšímu odhalování chyb. Párové programování je klíčovou součástí extrémního programování (XP) a dalších agilních metodik vývoje softwaru [1].

Mezi hlavní výhody párového programování patří vyšší kvalita kódu, rychlejší odhalení chyb, efektivní sdílení know–how a lepší týmová komunikace. Na druhou stranu může být tato metoda náročnější na časovou organizaci a vyžaduje dobrou spolupráci mezi partnery. V praxi se často řeší rozdíly v dovednostech, které však nemusí být překážkou, pokud jsou vhodně rozloženy vzhledem k cíli projektu. Například zkušenější partner může mentorovat méně zkušeného, který zase přinese nové perspektivy. [1]

S nástupem umělé inteligence se objevily nástroje, které umožňují podobnou spolupráci s AI asistentem. GitHub Copilot, Amazon CodeWhisperer nebo Chat GPT využívají velké jazykové modely (LLM), které byly trénovány na velkém množství veřejně dostupného kódu. V prostředí vývojářských IDE navrhují kód, opravují chyby a pomáhají se strukturou i logikou. AI asistenti mají schopnost zrychlit psaní kódu, snížit kognitivní zátěž programátora a pomoct hlavně s rutinními úkoly. To může významně zvýšit produktivitu vývojářů [3]. AI nástroje však nejsou náhradou za lidskou spolupráci ani know–how, ale spíše doplňkem, který může podpořit efektivitu a kreativitu vývojářů.

Párové programování s AI je aktuální téma pro softwarové inženýrství, protože reflektuje aktuální snahu o automatizaci vývoje a využívání jazykových modelů v programování. Porovnání tradičního párového programování s lidským partnerem a párového programování s AI asistentem může poskytnout cenné poznatky o výhodách a nevýhodách obou přístupů a pomoci vývojářům lépe porozumět, kdy a jak využívat AI nástroje ve své práci.

Motivací pro zkoumání tohoto tématu je osobní zkušenost s párovým programováním a zájem zjistit, jak efektivně využít AI asistenty při práci na školních i osobních projektech. Cílem je porovnat efektivitu a subjektivní vnímání spolupráce při párovém programování s lidským partnerem a s AI nástrojem.

Výzkumná rešerše

Čím dál více se řeší, jestli AI asistenti zvládnou plnohodnotně nahradit lidského parťáka při párovém programování, a jaký má volba vliv na efektivitu a uživatelskou zkušenost.

Tradiční párové programování je v odborné literatuře popisováno jako účinná metoda pro zlepšení kvality kódu, rozvoj týmové spolupráce a snížení chybovosti díky průběžné

kontrole práce v reálném čase [1]. Na druhou stranu studie často zmiňují i nevýhody, jako jsou příliš velké rozdíly v dovednostech partnerů nebo organizační složitost domlouvání společných sezení [2].

V posledních letech se do tohoto procesu stále častěji zapojují AI nástroje, jako je GitHub Copilot, Amazon CodeWhisperer nebo ChatGPT. Tito asistenti využívají velké jazykové modely (LLM) k automatickému generování kódu. Vývojářům pomáhají zejména při řešení rutinních úkolů, takže šetří čas a kognitivní zátěž. Postupně se však ukazuje, že zvládají i komplexnější logiku – vědí, co „dává smysl“ v rámci dané funkce, ne jen co syntakticky pasuje.

Studie Fan et al. (2025) přinesla komplexní srovnání mezi třemi formami programování: individuálním, tradičně párovým (člověk–člověk) a párovým s AI asistentem. Ukázalo se, že AI asistenti významně snižují programátorskou úzkost, a to především díky své „neodsuzující“ povaze. Studenti se nebojí dělat chyby, neboť nemají strach ze společenského hodnocení. Tento efekt měl přímý vliv na zlepšení výkonu, zejména u méně zkušených účastníků [3].

Z hlediska motivace a vnímání autonomie AI partner podporoval u studentů větší pocit kompetence – poskytoval okamžitou zpětnou vazbu a umožnil jim objevovat řešení vlastním tempem. V tom se ukázal jako vhodný nástroj zejména pro začátečníky. Nicméně v oblastech jako je sociální přítomnost tradiční lidské párování jednoznačně vedlo. Účastníci vnímali lidské partnery jako více podporující a oceňovali možnost verbální interakce, sdílení nápadů či společného ladění dalšího postupu práce [2].

Empirické studie ukazují, že vývojáři při spolupráci s AI pociťují nižší míru stresu díky okamžité zpětné vazbě a absenci hodnotícího postoje ve formě programovacího partnera [2]. Na druhou stranu však postrádají hlubší diskuzi a kreativní brainstorming, které při spolupráci s člověkem vznikají přirozeně. Fan et al. (2025) tvrdí, že ačkoliv AI asistenti mají pozitivní vliv na pocit autonomie a kompetence programátorů, nedokážou nahradit sociální dimenzi lidské spolupráce [3].

Navíc se stále řeší otázky týkající se ochrany dat a autorského práva. AI asistenti jsou obvykle trénováni na veřejném kódu, což vyvolává pochybnosti o tom, komu patří vygenerovaný výstup. Zároveň není jisté, co vše se odesílá na servery, a zda mezi tím může být i citlivý kód (např. hesla, firemní algoritmy, osobní údaje v datech). Tato rizika jsou zatím jen částečně ošetřena a v mnoha firmách představují důvod, proč AI nástroje oficiálně zakazují [2].

Z toho, co zatím víme, se zdá, že AI partner v párovém programování nepředstavuje plnohodnotnou náhradu lidské spolupráce, ale spíš specifický druh interakce s vlastními výhodami, limity i etickými otázkami.

Výzkumné otázky

Cílem tohoto výzkumu je porovnat zkušenost a efektivitu práce při párovém programování s lidským partnerem a s AI asistentem. Důraz bude kladen nejen na objektivní měřitelné výsledky, ale také na subjektivní introspektivní zkušenost. Výzkum je zaměřen na tři hlavní otázky:

- **VO1:** Jaký je rozdíl v efektivitě (čas, počet iterací, kvalita kódu) mezi párovým programováním s AI a s lidským asistentem?
- **VO2:** Jak se liší subjektivní vnímání programátorské úzkosti, motivace a míry kolaborace při párovém programování s AI oproti lidskému partnerovi?
- **VO3:** Do jaké míry dokáže AI kompenzovat kolaborativní benefity lidského partnera při párovém programování z pohledu uživatelské zkušenosti?

Pro účely experimentu bude jako AI asistent využit GitHub Copilot. Práce bude probíhat ve vývojovém prostředí Visual Studio Code. Všechny úlohy budou implementovány v programovacím jazyce Python.

Lidským partnerem v experimentu bude David Král, student oboru aplikovaná informatika, se srovnatelnou úrovní programátorských dovedností jako autorka.

Detailní popis úloh, měřených metrik a způsobu provedení je uveden v následující kapitole.

Návrh experimentu

Cílem experimentu bylo porovnat párové programování s AI asistentem a s lidským partnerem z hlediska efektivity i subjektivní zkušenosti.

V experimentu jsem sledovala dvě hlavní skupiny metrik: **výkonnostní** a **introspektivní**. První skupina zahrnovala **čas potřebný k vyřešení úlohy** (v minutách), **počet iterací** (kolikrát bylo nutné kód opravit, než prošel jednotkovými testy) a **kvalitu kódu**, kterou hodnotil nezávislý hodnotitel Roland Magera podle dvou kritérií – **srozumitelnost** (např. názvy proměnných, čitelnost, komentáře) a **technická úroveň** (správnost, efektivita, struktura kódu). Obě složky byly hodnoceny na škále 1 až 5; jejich součet určoval celkové hodnocení (2–10 bodů).

Introspektivní metriky zjišťovaly subjektivní prožitek z programování – míru stresu, motivaci a vnímanou kvalitu spolupráce. Vycházela jsem z metodologie rozsáhlejší studie [3] a použila tři validované škály: **Programming Anxiety Scale (PAS)**: 14 položek zaměřených na pocity nervozity, pochybností a stresu během psaní kódu [4], **Intrisic Motivation Inventory (IMI)**: 23 položek zaměřených na zájem, úsilí a kompetenci [5], **Collaborative Learning & Sense of Community Scale (CLS/SoC):** 11 položek hodnotících spolupráci a sociální propojení [6].

Všechny tři introspektivní škály byly upraveny tak, aby odpovídaly podmínkám mého experimentu. To zahrnovalo jak jazykové přizpůsobení (osobní zájmena, kontext), tak v případě CLS také výraznější úpravy formulací, aby bylo možné škálu použít v kontextu párového programování. Původní význam jednotlivých položek zůstal zachován. Škály byly poté sjednoceny na pětibodovu Likertovu škálu (1 = zcela nesouhlasím/nikdy, 5 = zcela souhlasím/vždy). U původně sedmibodové škály IMI byla použita lineární transformace. Výsledkem každé škály bylo průměrné skóre ze všech položek.

Celý experiment probíhal během tří dnů (úterý až čtvrtek) v květnu 2025. Úlohy, které jsem řešila s lidským partnerem (Davidem Králem), jsme programovali společně v univerzitní knihovně. Druhou polovinu úloh jsem řešila samostatně doma. V obou případech se pracovalo ve vývojovém prostředí **Visual Studio Code** v programovacím jazyce **Python**. Jako AI nástroj byl použit **GitHub Copilot**, aktivovaný pomocí oficiálního rozšíření (v rámci bezplatné zkušební verze).

Každá úloha měla vlastní specifikaci a byla implementována jako samostatný projekt podle předepsaného zadání. Pro úspěšné dokončení bylo nutné, aby řešení prošlo všemi připravenými jednotkovými testy. Aby bylo možné metriky měřit jednotně a bezchybně, vytvořila jsem jednoduchou infrastrukturu, která je veřejně dostupná v repozitáři: *http://github.com/majdakleckerova/KI-SWI*.

Rozhraní jednotlivých úloh byla definována v samostatných souborech ve složce *solutions/*. Ke každé úloze náležela odpovídající sada testů ve složce *tests/*. Celkem bylo v rámci celého experimentu použito 26 jednotkových testů. Zadání všech úloh je přehledně shrnuto v souboru *zadani.md*. Pro měření času a počtu iterací byl použit wrapper skript *track_solution.py*, který se spustil před začátkem práce na konkrétní úloze. Po každém uložení kódu se automaticky spustily testy a výsledek se zobrazil v terminálu. Pokud některý test

neprošel, kód bylo možné opravit a uložit znovu. Skript pokračoval v měření až do chvíle, kdy kód prošel všemi jednotkovými testy úspěšně. Následně v terminálu zobrazil celkový čas řešení a počet provedených iterací.

Dohromady experiment obsahoval šest úloh ve třech kategoriích podle typu zaměření:

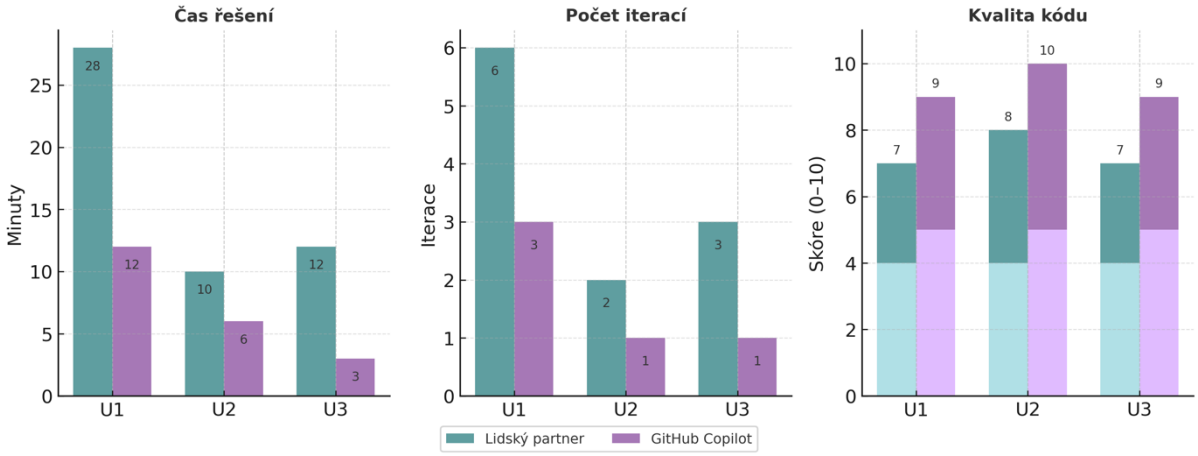
- **Logické úlohy:**
  - *Piškvorky (U1A – člověk):* klasická hra pro dva hráče na poli 3x3,
  - *Kámen-nůžky-papír (U1B – AI):* hra na tři vítězná kola proti náhodně generovaným tahům.
- **Agregační úlohy:**
  - *Průměrné známky (U2A – člověk):* načtení CSV souboru a výpočet průměrné známky pro každý předmět pomocí knihovny Pandas,
  - *Tržby (U2B – AI):* filtrování objednávek z roku 2024 a výpočet celkových tržeb za každého zákazníka, opět na základě dat z CSV.
- **Algoritmické úlohy:**
  - *FizzBuzz (U3A – člověk):* klasická úloha, kde se čísla od 1 do 100 nahrazují řetezci podle dělitelnosti třemi a pěti,
  - *Palindrom (U3B – AI):* kontrola, zda je zadaný text palindrom, přičemž se ignorují mezery i velikost písmen.

Tato sada úloh a použitá měřicí struktura vytvořily dostatečně různorodý rámec pro porovnání. Výsledky jednotlivých metrik jsou podrobně rozpracovány v následující kapitole.

Výsledky a diskuze

Cílem experimentu bylo zjistit, jak se liší zážitek i efektivita programování s AI Asistentem a lidským partnerem. Nešlo jen o to, který způsob je rychlejší, ale o hlubší pochopení toho, co vlastně znamená spolupracovat – s člověkem, nebo s nástrojem, který člověka pouze imituje. Zajímalo mě, jak se cítím a jak přemýšlím, a co mi ta interakce dává nebo bere. Následující část je rozdělena podle výzkumných otázek a zaměřuje se na to, co výsledky znamenají pro programátorskou praxi.

VO1: Srovnání efektivity

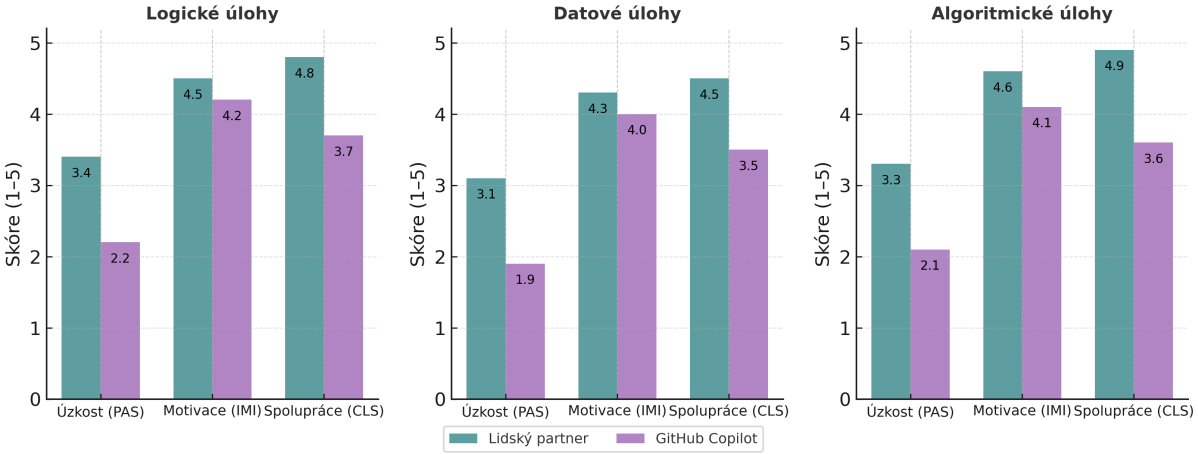


Při porovnání výkonu obou přístupů se ukázalo, že GitHub Copilot výrazně zkrátil dobu potřebnou k vyřešení úloh. Ve všech případech byla práce s AI rychlejší, u algoritmické úlohy dokonce trojnásobně. Práce navíc vyžadovala méně oprav, tedy iterací. Kód generovaný s pomocí AI zároveň dosahoval srovnatelné nebo vyšší kvality, jak po technické stránce (správnost, efektivita), tak po stránce srozumitelnosti (čitelnost, pojmenování proměnných, struktura).

Tyto rozdíly lze ilustrovat v tabulce a grafu výkonnostních metrik, které ukazují, že Copilot nejen urychluje vývoj, ale zároveň pomáhá udržet vysokou kvalitu výstupu. Zvláště v jednodušších a jasně specifikovaných úlohách, kde nehrozí nepochopení zadání, AI exceluje díky své schopnosti nabídnout rychlé, syntakticky korektní a elegantní řešení.

Z hlediska praktického využití se ukazuje, že AI asistence je ideální v situacích, kdy je potřeba pracovat rychle, efektivně a bez zbytečné kognitivní zátěže. Typicky se to týká rutinních nebo repetitivních úloh, kde není třeba rozsáhlá diskuze nad strategií. V takových případech může AI fungovat jako urychlovač – nástroj, který umožní vývojáři rychleji se dostat k podstatě problému. Naopak při komplexnějších úlohách může lidský partner nabídnout hlubší vhled, navrhnout alternativní postupy nebo rozpoznat skryté souvislosti. Výsledky tak potvrzují hypotézu, že AI v roli „parťáka“ spíše doplňuje lidské schopnosti, než aby je plně nahrazovala.

VO2: Srovnání subjektivního vnímání spolupráce



Subjektivní rozdíly mezi prací s člověkem a AI byly výrazné. Při spolupráci s GitHub Copilotem jsem vnímala nižší míru stresu – necítila jsem tlak na výkon, nebála jsem se chyb a mohla jsem pracovat vlastním tempem. Výsledky Programming Anxiety Scale tuto zkušenost potvrzují: úzkost při práci s AI byla znatelně nižší než při práci s člověkem. Tento efekt byl obzvlášť patrný u logické úlohy, kdy AI poskytovala klidné, rychlé a „neodsuzující“ návrhy.

Na druhou stranu motivace i vnímaná míra kolaborace byla vyšší při práci s lidským partnerem. S Davidem jsme si vyměňovali nápady, doplňovali se, smáli se absurditám a společně ladili strategii. Spolupráce s člověkem tak přinášela nejen vyšší emoční zapojení, ale i pocit sdílení a týmovosti. Výsledky škál IMI a CLS ukazují, že lidské párové programování bylo vnímáno jako angažovanější, motivující a společensky bohatší.

Z těchto poznatků vyplývá, že AI může být velmi vhodným nástrojem pro situace, kdy je potřeba pracovat bez stresu, rychle a samostatně. Vývojáři, kteří se potýkají s nejistotou, únavou nebo strachem ze selhání, mohou v Copilotu najít klidného a spolehlivého „spoluhráče“. Naopak lidský partner přináší přidanou hodnotu tam, kde je cílem sdílení znalostí, kreativita nebo výuka. Pro začátečníky může být AI bezpečným prostorem pro pokusy bez hodnocení, zatímco člověk nabídne hlubší zpětnou vazbu a osobní podporu.

VO3: Srovnání kolaborativních benefitů

Z hlediska uživatelské zkušenosti působí AI jako zcela odlišný typ partnera než člověk. GitHub Copilot je vždy připravený, nikdy nezaváhá, nehodnotí a nevnucuje pocit, že něco dělám špatně. Právě to, že jsem se mohla svobodně zkoušet, chybovat a experimentovat bez obav z posuzování, pro mě bylo až překvapivě osvobozující. V tom smyslu Copilot

nepůsobí jen jako chytrý nástroj, ale spíš jako médium, které mění způsob, jak uvažujeme o vlastní kompetenci a komfortní zóně při psaní kódu.

Současně jsem si ale uvědomila, co mi v této formě spolupráce chybí. Scházela mi přirozená interakce – humor, společné přemýšlení, okamžitá zpětná vazba. Spolupráce s lidským partnerem je v ideálním případě nejen technicky efektivní, ale i lidsky naplňující. Přináší pocit sdílení, sounáležitosti a radosti z tvoření. V tandemu s člověkem nejsem nikdy „sama na problém“. S AI sice nepociťuji tlak, ale často i určité osamění.

Tato zkušenost koresponduje se závěry odborných studií – například Fan et al. (2025) uvádějí, že AI asistenti mohou účinně podporovat samostatnou práci, snižovat úzkost a posilovat autonomii. Zároveň ale nevyvolávají pocit skutečné přítomnosti, sdílení či emoční odezvy. Z toho důvodu by měli vývojáři chápat AI spíše jako doplňkový nástroj, který má své místo – zejména v individuální, klidné práci nebo při opakovaných úlohách. Tam, kde je potřeba kreativní spolupráce, sdílení perspektiv nebo lidské porozumění, zůstává lidský partner nenahraditelný.

Pokud se podaří najít kolegu, se kterým je spolupráce přirozená a respektující, může být párové programování nejen efektivní technikou, ale i hluboce obohacujícím zážitkem – profesně i osobně. A právě tento rozměr lidské spolupráce zatím žádný jazykový model napodobit nedokáže.

Závěr

Výsledky tohoto experimentu ukazují, že párové programování s AI asistentem GitHub Copilotem vede k vyšší efektivitě. Úlohy byly řešeny rychleji, s menším počtem iterací a bez negativního dopadu na kvalitu kódu. Introspektivní data navíc potvrdila, že práce s AI snižuje míru prožívané úzkosti a zvyšuje pocit autonomie. Naopak lidské párování přinášelo vyšší motivaci, vnímanou spolupráci a celkově silnější pocit sdílení a podpory. Odpovědi na výzkumné otázky tedy ukazují, že AI může být velmi efektivním pomocníkem zejména v individuální nebo stresové práci, ale lidský partner zůstává nenahraditelný v oblastech, kde záleží na týmovosti, empatii a vzájemném učení.

Výzkum měl však řadu omezení, která je třeba brát v úvahu. Pracovala jsem pouze se šesti triviálními úlohami, navrženými tak, aby se daly zvládnout během několika hodin. Experiment probíhal jen se dvěma účastníky – mnou a jedním lidským partnerem – a výsledky introspektivních metrik jsou založené výhradně na mém subjektivním prožitku. Hodnocení kvality kódu provedla pouze jedna osoba, a to na základě předem definovaných kritérií, ale bez objektivních metrik (např. počet bugů v produkci). AI asistentem byl pouze GitHub Copilot, a není jasné, jak by se výsledky lišily s jinými nástroji nebo v jiném vývojovém prostředí.

Do budoucna by proto bylo vhodné provést rozsáhlejší výzkum s větším počtem účastníků a zapojením různých AI nástrojů. Zajímavou možností by bylo sledovat delší spolupráci (např. v rámci semestrálního projektu), případně sbírat fyziologická data (např. pomocí chytrých hodinek) pro objektivní měření stresu. Výzkum by také mohl rozšířit spektrum úloh na komplexnější zadání blízká praxi a zkoumat nejen binární srovnání člověk–AI, ale i další formy spolupráce, jako je individuální řešení bez partnera nebo střídání rolí mezi AI a člověkem. Téma párového programování v éře jazykových modelů zůstává otevřené a zasluhuje si další pozornost.

Použitá literatura

1. HOREJŠEK, Jan. O čem je párové programování [online]. 2016 [cit. 2025-05-13]. Dostupné z: https://blog.horejsek.com/o-cem-je-parove-programovani/
2. MA, Quianou, Tongshuang WU a Kenneth KOEDINGER. *Is AI the better programming partner? Human-Human Pair Programming vs. Human-AI pAIr Programming* [online]. [cit. 2025-05-13]. Dostupné z: https://arxiv.org/pdf/2306.05153
3. FAN, Guangrui, Dandan LIU, Rui ZHANG a Lihu PAN. *The impact of AI-assisted pair programming on student motivation, programming anxiety, collaborative learning, and programming performance: a comparative study with traditional pair programming and individual approaches* [online]. 2025 [cit. 2025-05-13]. Dostupné z: https://stemeducationjournal.springeropen.com/articles/10.1186/s40594-025-00537-3
4. YILDIRIM, Osman Gazi a Nesrin OZDENER. *The Development and Validation of the Programming Anxiety Scale* [online]. 2022 [cit. 2025-05-17]. Dostupné z: https://files.eric.ed.gov/fulltext/EJ1345555.pdf
5. DECI, Edward L. a Richard M. RYAN. *Intrinsic Motivation Inventory (IMI)*[online]. [cit. 2025-05-17]. Dostupné z: https://selfdeterminationtheory.org/intrinsic-motivation-inventory/
6. RYAN, Richard M. a Ana-Paula CORREIA. *Online Students' Attitudes Toward Collaborative Learning and Sense of Community* [online]. 2019 [cit. 2025-05-17]. Dostupné z: https://www.researchgate.net/publication/338162263_Online_Students'_Attitudes_Toward_Collaborative_Learning_and_Sense_of_Community