Zadání projektu do předmětů **Algoritmy I** a **Sazba technických dokumentů** akademický rok 2024/2025, letní semestr kombinovaná forma studia

Historie modifikací

14. března 2025	První verze
25. března 2025	Hypertextová verze zadání, reformulace zadání B
1. dubna 2025	Doplněno zadání pro studenta STA0727

Obsah

Deadline

Vypracované řešení projektu je nutno odevzdat do **18. května 2025 23:59**.

1 Obecné pokyny

- Každý student či studentka má přiděleno právě jedno zadání projektu z předmětu Algoritmy I. Rozdělení zadání projektů mezi studenty je součástí tohoto dokumentu.
- Studenti, kteří mají zapsán i předmět Sazba technických dokumentů (STD) vypracují na základě řešení projektu do Algoritmů I i projekt do předmětu STD. Oba projekty pak odevzdají společně. *Studenti, kteří předmět STD zapsán nemají, pochopitelně žádný projekt do předmětu STD nevypracovávají!*
- S případnými dotazy kontaktujte tutora.
- Termíny obhajob budou vypsány v systému Edison.
- Deadline je konečný a nebude dále posunován. Na projekty odevzdané po tomto datu nebude brán zřetel. Projekt lze pochopitelně odevzdat a domluvit si termín obhajoby i dříve.
- Projekty odevzdávejte přes Dropbox na URL

  `https://www.dropbox.com/request/MBc27aZR1UpBV9bcbNsy`.

  Odevzdávají se soubory se zdrojovým kódem, hlavičkové soubory, soubory s projektem atd., jinak řečeno vše, co je potřebné pro bezproblémovou kompilaci odevzdaného projektu. V projektu do STD se odevzdávají zdrojové tex soubory, obrázky atd., tedy opět vše potřebné pro bezproblémovou kompilaci dokumentu.
- Součástí zdrojových kódů Vašeho programu bude programátorská dokumentace ve formě dokumentačních komentářů, zpracovatelných programem Doxygen, viz `www.doxygen.org`. Vygenerovanou dokumentaci není nutné odevzdávat. Postačuje, pokud Vámi odevzdaný archiv bude obsahovat konfigurační soubor `doxyfile`, případné adresáře pro vygenerovanou dokumentaci, vkládané obrázky atd.
- Ačkoliv se zadání mohou jevit na první pohled složitá, nepropadejte panice. Vyřešení kteréhokoliv zadání by nemělo zkušenému programátorovi zabrat více než „jeden večer“. Studentům prvního a druhé ročníku to zabere asi více času, ale v žádném případě by řešení nemělo trvat „desítky a desítky“ člověkohodin práce. Stejně tak co se týče délky vytvořeného kódu. Pokud jste už napsali „tisíce“ řádků kódu a stále není konec v dohledu, je to špatně. Takový zdrojový kód rovnou zahoďte. Klíčem k řešení je tužka, papír a hlava. Zkuste si řešení nejdříve promyslet, kreslit si u toho různá schémata, náčrtky datových struktur, volání funkcí a tak dále. Projděte si literaturu. A až se dostaví „aha moment“ tak začněte psát kód.

2 Kritéria hodnocení projektů

2.1 Algoritmy I

Kritéria hodnocení řešení projektů jsou tato:

1. **Správnost řešení** Správnost řešení je podmínka nezbytná. Aplikace, která nebude poskytovat správné výsledky, bude hodnocena automaticky 0 body bez ohledu na další kritéria. Ke každému zadání jsou k dispozici testovací data a výsledky, lze si tedy ověřit správnost řešení.
2. **Volba vhodných datových struktur** Toto kritérium hodnotí, v závislosti na konkrétním zadání, volbu vhodné datové struktury a algoritmů pro manipulaci se zvolenou datovou strukturou.

3. **Dekompozice problému na menší celky**

   - V předmětu Algoritmy I je požadována procedurální dekompozice, čili dekompozice řešení do funkcí. Využití objektově orientovaného programování je v tomto předmětu dobrovolné.
   - V předmětu Algoritmy II je požadován objektový návrh řešení a tomu odpovídající implementace.

4. **Způsob implementace** Toto kritérium hodnotí oddělení deklarace a definice funkcí nebo tříd do h a cpp souborů, využívání konstant místo přímo zapsaných hodnot, využívání parametrů funkcí a výsledků funkcí místo využívání side efektu založeném na globálních proměnných. Dále sem patří i úroveň zápisu zdrojového kódu, například odsazování vnořených konstrukcí, vhodné pojmenování proměnných, funkcí, tříd, dodržování zvolené konvence pojmenování[1] proměnných, funkcí a tříd, dodržování zvolené konvence psaní bloků kódu[2] a tak dále.

5. **Efektivita implementovaného algoritmu** Smyslem tohoto kritéria není nutit vás k implementaci nejlepšího známého algoritmu tím nejlepším možným způsobem. Tímto kritériem si vyučující ponechávají prostor pro případné snížení bodového hodnocení za použití algoritmu zcela nevhodného, nesmyslného, zmateného. *Příklad*: Součástí řešení projektu je třídění pole či vektoru s n prvky. Pokud použijete algoritmus se složitostí $O(n \log_2 n)$ je vše v pořádku. Pokud použijete některý z jednodušších algoritmů se složitostí $O(n^2)$, asi vás vyučující při obhajobě upozorní, že to nebyla dobrá volba, ale stále je to v pořádku. Za zcela nesmyslný algoritmus je v tomto případě považován algoritmus se složitostí větší než $O(n^2)$, protože takový algoritmus z podstaty věci dělá zbytečnou práci.

6. **Dokumentace k projektu** Ke každé funkci, třídě, atributu třídy musí existovat alespoň krátký dokumentační komentář ve formátu zpracovatelném programem Doxygen. Pro zápis dokumentačních komentářů lze využít libovolný z formátů, které Doxygen podporuje. Vygenerovanou dokumentaci není nutné odevzdávat, ale je nutné odevzdat konfigurační soubor `doxyfile`, aby bylo možné dokumentaci bez problémů vygenerovat.

   K programu Doxygen existuje rozsáhlá dokumentace volně dostupná na URL `https://www.doxygen.nl/manual/index.html`. Pro dokumentaci řešení semestrálního projektu je vhodné si prostudovat následující části dokumentace:

   - „Getting started“, `https://www.doxygen.nl/manual/starting.html`,
   - „Documenting the code“, `https://www.doxygen.nl/manual/docblocks.html`
   - „Doxygen usage“, `https://www.doxygen.nl/manual/doxygen_usage.html`
   - „Doxywizard usage“, `https://www.doxygen.nl/manual/doxywizard_usage.html`

     Program Doxywizard slouží k pohodlnému vygenerování konfiguračního souboru `doxyfile`, který tak není nutné vytvářet ručně.

7. **Citace zdrojů** Žádný program nevzniká úplně z ničeho, ani řešení semestrálních projektů. K řešení projektů můžete používat odbornou literaturu, učebnice, příklady zdrojových kódů z ostatních předmětů, internetové zdroje. V tom případě je nutné uvést, že „Tento algoritmus jsem převzal z...“, „Tuto část kódu jsem převzal z...“. Tyto případné citace musí umístit do dokumentačních komentářů. Asi není nutné citovat Levitinovu knihu, že jsem si tam „přečetl něco o průchodu grafem do šířky“ nebo, že „toto jsme řešili na cvičení“. Ostatní zdroje by se však citovat měly.

[1] Typicky camelCase, PascalCase, méně vhodná je už například maďarská notace.
[2] Typicky – složená levá závorka za příkazem nebo na novém řádku.

2.2 Sazba technických dokumentů

Kritéria hodnocení tohoto projektu jsou benevolentnější než u Algoritmů I. Zde je ponechán prostor Vaší kreativitě. Nicméně se očekává, že vznikne text o několika stranách, členěný nadpisy do dvou, maximálně tří úrovní, a obsahující:

- titulní stranu,
- obsah,
- seznam obrázků,
- seznam tabulek.

Dále by se v textu měly objevit:

- citace literatury a ostatních zdrojů[3],
- obrázky – schémata, grafy zobrazující experimentální výsledky atd.,
- tabulky s experimentálními výsledky,
- odkazy na obrázky a tabulky,
- ukázky zdrojového kódu v C++, který řeší podle Vás zajímavou část projektu a
- algoritmy zapsaný formou pseudokódu.

Složité matematické vzorce se pravděpodobně nebudou v projektu objevovat, ale je nutno dbát na to, aby matematické symboly byly sázeny matematickou sazbou, například „V projektu pracujeme s grafem G…“, kde označení grafu je vysázeno matematickou sazbou čili `$G$`.

Minimální počet stran není stanoven, protože je nežádoucí aby byl dokument vyplněn bezobsažným textem, *vatou*, jen proto, aby byl naplněn požadovaný počet stran. Předpokládá se však, že dokument bude obsahovat vyšší jednotky stránek, spíše však se bude jednat o více než deset stran textu[4].

[3] Tady lze použit samozřejmě stejné zdroje jako v projektů do Algoritmů.

[4] Jedna strana bude titulní, další strana bude obsahovat obsah dokumentu, seznam obrázků a tabulek. Na konci dokumentu bude na další straně seznam citované literatury. A pochopitelně tabulkami a obrázky strany docela rychle přibývají…

3 Zadání projektu do Algoritmů I

Zadání projektu se skládá ze dvou částí. První část je společná pro obě zadání. Tuto část vypracují všichni studenti. A dále se zadání projektu dělí na dvě části označené **A** a **B**. Rozdělení studentů mezi tyto části zadání najdete v sekci 5.

3.1 Společná část zadání

V zadání projektu budeme zpracovávat graf české mutace Wikipedie, kde jednotlivé stránky Wikipedie odpovídají vrcholům grafu a odkazy mezi stránkami budeme chápat jako orientované hrany. Wikipedie tak bude pro nás představovat orientovaný graf $G = (V, E)$, kde G je množina vrcholů a E je množina hran. Vrcholy grafu jsou označeny celočíselnými identifikátory většími než nula.

Vaším úkolem ve společné části zadání projektu je:

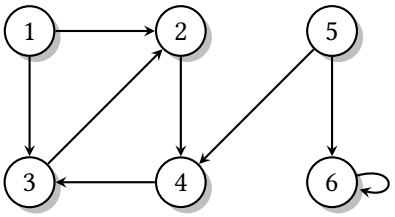
1. Navrhnout a implementovat vhodnou reprezentaci grafu G.
2. Implementovat funkci či funkce pro načtení grafu G do této datové struktury. Graf je uložen v textovém souboru ve formě seznamu hran. Každá hrana je uložena na samostatném řádku jako dvojice kladných celých čísel představující počáteční vrchol a koncový vrchol hrany. Číselné identifikátory obou vrcholů jsou odděleny bílými znaky. Ukázkový graf na obrázku 1 bude uložen například takto:

   ```
   5 6
   3 2
   6 6
   4 3
   1 2
   1 3
   2 4
   5 4
   ```

   Jak je vidět hrany nejsou nijak uspořádány, pořadí hran je tedy zcela náhodné.
3. Implementovat funkce, které do textového souboru či souborů vypíšou základní informace o grafu:
   - počet vrcholů,
   - počet hran[5],
   - tabulku četností vstupních stupňů všech vrcholů grafu a
   - tabulku četností výstupních stupňů všech vrcholů grafu.

   Vstupní stupeň $indegree_v$ vrcholu v v grafu G definujeme neformálně jako počet hran, které vstupují do vrcholu v. Výstupní stupeň $outdegree_v$ vrcholu v definujeme jako počet hran vystupujících z vrcholu v. Vstupní a výstupní stupně všech vrcholů ukázkového grafu z obrázku 1 jsou uvedeny v tabulce 1. Četnosti vstupních a výstupních stupňů ukázkového grafu jsou pak uvedeny v tabulce 2.
4. Implementovat funkci, která bude vracet **true**, pokud graf G je acyklický. Jinak bude vracet **false**. Acykličnost grafu G ověřte pomocí algoritmu topologického třídění.

[5] Jen pro kontrolu, graf obsahuje 2 673 524 vrcholů a 40 769 236 hran.



Obrázek 1: Ukázkový graf

v	$indegree_v$	$outdegree_v$
1	0	2
2	2	1
3	2	1
4	2	1
5	0	2
6	2	1

Tabulka 1: Vstupní a výstupní stupně vrcholů ukázkového grafu

3.2 Část zadání A

V této části zadání máte vypočítat vzdálenosti všech dosažitelných vrcholů grafu G od daného počátečního vrcholu s. Vzdálenost $d(u, v)$ dvojice vrcholů u, v grafu G definujeme jako počet hran na nejkratší orientované cestě z vrcholu u do vrcholu v[6]. Pokud taková cesta neexistuje, definujeme $d(u, v) = \infty$ a říkáme, že vrchol v je z vrcholu u nedosažitelný.

Z takto vypočtených vzdáleností opět sestavte tabulku četností, tj. tabulku která bude udávat kolikrát se určitá délka orientované cesty ve vypočtených vzdálenostech vyskytuje. Výsledek vypište do textového souboru či souborů.

Vzdálenosti vypočtěte pro tyto počáteční vrcholy s:

- stránku VŠB-TU Ostrava, $s = 22197$,
- stránku Univerzity Karlovy v Praze, $s = 1083$.
- Na tomto místě můžete namítnout, že mám jít s celým zadáním projektu už do Prčic a že si jdete dát pivo. Takže další dva počáteční vrcholy budou: stránka obce Prčice, $s = 122606$ a stránka o pivu $s = 6916$.

3.3 Část zadání B

V bodě 4 společné části zadání jste měli za úkol otestovat acykličnost grafu G. V této části zadání máte za úkol, převést graf G na acyklický graf G' odstraněním zpětných hran při průchodu grafem

[6]Pozor, graf G je orientovaný, orientovaná cesta z vrcholu u do vrcholu v nemusí být shodná z orientovanou cestou z v do u nebo orientovaná cesta z v do u nemusí vůbec existovat.

$indegree_v$	četnost
0	2
2	4

(a) vstupní stupně

$outdegree_v$	četnost
1	4
2	2

(b) výstupní stupně

Tabulka 2: Tabulka četností vstupních a výstupních stupňů

G do hloubky. Výsledek otestuje pomocí funkce implementované v bodě 4 společné části zadání. Pro graf G' zjistěte opět hodnoty uvedené v bodu 3 společné části zadání.

3.4 Doporučená literatura a další poznámky k řešení

Než se pustíte do programování douporučuji si pozorně přečíst kapitoly 1.4, 3.5 a 4.2 z Levitinovy knihy [1]. Stejně tak kapitolu 20 z knihy [2]. Množství informací o grafech najdete v českém učebním textu docenta Kováře z naší univerzity [3]. K topologickému třídění by mohl být užitečný i článek na anglické Wikipedii, https://en.wikipedia.org/wiki/Topological_sorting.

Dále bych rád upozornil, že pokud se budete snažit implementovat jakýkoliv algoritmus, který bude rekurzivně procházet všechny vrcholy nebo hrany grafu, tak na to rovnou zapomeňte. Vrcholů v grafu české Wikipedie je už takové množství, že to systémový zásobník sloužící k jinak perfektnímu fungování rekurzivních funkcí nezvládne. Proto je nutné takové algoritmy implementovat *nerekurzivně*.

Vzhledem k množství dat, které máte zpracovat, bude vhodné využít i jiné datové struktury než jen `vector`, případně pole.

Jak společnou část, tak obě části zadání A a B jsem si cvičně naimplementoval, abych věděl, zda se dá úloha řešit i na normálním počítači a jak dlouho bude výpočet trvat. Doba běhu společné části, části A a pak části B, včetně načtení dat, byla v produkčním (release) režimu 152 sekund. Řešení jsem implementoval v jazyce C# s použitím .NET 8.0 a pro testování použil starý počítač s procesorem Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz.

Tvar výstupu do textového souboru si zvolte jaký chcete. Třeba takový, aby jej bylo možné lehce použít v projektu do STD.

4 Zadání projektu do Sazby technických dokumentů

Cílem tohoto projektu je demonstrace vaší schopnosti vysázet v LaTeXu technický dokument se všemi jeho typickými prvky. V dokumentu, který vznikne v rámci projektu do STD, tak nejdříve stručně zopakujte zadání svého projektu, pak popište postup svého řešení, jaké algoritmy jste použili, jaké datové struktury jste při řešení využili, uveďte zdroje z nichž jste čerpali. Dále v dokumentu zpracujte výsledky svého řešení ve formě tabulek a grafů.

V dokumentu popište i jak jste řešení svého projektu přepracovávali, co se nepovedlo, co se nakonec ukázalo jako špatný nápad, špatná cesta k řešení, jaká část řešení se ukázala jako nejnáročnější a tak dále. Jde o akademický dokument, takže je bez problémů možné, a téměř žádoucí, popsat slepé uličky, nezdary a tak dále.

Co se týče technické stránky – použijte třídu dokumentů `article`, písmo velikosti 11 bodů a formát strany samozřejmě A4. Okraje a další nastavení není nutné nijak měnit. Dokument vysázejte písmem Latin Modern.

Odkazy

1. LEVITIN, Anany. *Introduction to the Design and Analysis of Algorithms*. 3rd ed. Boston: Pearson, 2012. ISBN 978-0-13-231681-1.
2. CORMEN, Thomas H.; LEISERSON, Charles Eric; RIVEST, Ronald L.; STEIN, Clifford. *Introduction to algorithms*. Fourth edition. Cambridge, Massachusetts: The MIT Press, 2022. ISBN 978-026--2046-305.
3. KOVÁŘ, Petr. *Teorie grafů*. Ostrava, 2022. Dostupné také z: https://mi21.vsb.cz/modul/teorie-grafu.

5 Rozdělení zadání mezi studenty

	Login	Příjmení jméno	Zadaný projekt
1	BOR0250	Borovyk Oryna	A
2	BUJ0045	Bujnovský František	B
3	CEP0027	Čep Radim	A
4	DAD0025	Ďaďová Hana, Bc.	B
5	DAN0172	Daňo Tomáš	A
6	DRA0190	Drábik Róbert, Ing.	B
7	FEI0017	Feifič Jan	A
8	FIA0120	Fiala Jiří	B
9	FUK0014	Fukala Stanislav	A
10	HER0027	Herfert Radovan	B
11	HOR0615	Horová Simona	A
12	HYV002	Hyvnar Radim, Ing.	B
13	KAT0030	Katzer Jakub	A
14	KUL0166	Kulíšková Lucie	B
15	KUL0172	Kulíšek Dušan	A
16	KUN212	Kunčík David	B
17	KUP0148	Kupec Timoteus	A
18	LAK0028	Lakomý Petr, Ing.	B
19	LAN0245	Langer Vít	A
20	LAZ0090	Lazarević Lazar	B
21	MAC0690	Machů Zdeněk, Ing.	A
22	MAL0488	Malý Dominik	B
23	MAR1023	Marejka Miloš	A
24	MIK0545	Mikešová Alena	B
25	MOL0125	Molinova Karolina	A
26	OBS0020	Obšivač Martin	B
27	PEN0050	Pěnkava Daniel	A
28	PFE0013	Pfeffer Steven	B
29	REZ0130	Řezníčková Martina, Bc.	A

pokračování na další stránce…

	Login	Příjmení jméno	Zadaný projekt
30	RYB0127	Ryba Jiří	B
31	SAL0110	Salamon Ondřej, Bc.	A
32	SED0253	Sedláček Michal	B
33	SIM0446	Šimková Henrieta	A
34	SOK0055	Sokola Robert	B
35	SOL0139	Šoltys Daniel	A
36	SPI096	Špička Petr	B
37	SUC0149	Suč Timotej	A
38	SZO0046	Szotek David	B
39	URB0312	Urbanec Filip Martin	A
40	ZEG0010	Zegzulková Eliška	B
41	ZOU0031	Zoubek Martin	A
42	STA0727	Starý Radek	B