Zadání projektu z předmětu IPP 2012/2013

Zbyněk Křivka a Dušan Kolář
E-mail: {krivka, kolar}@fit.vutbr.cz, {54 114 1313, 54 114 1238}

1 Základní charakteristika projektu

Název projektu:

Návrh, implementace a dokumentace **dvou skriptů** v jazyce **Perl** a **Python 3**

Revize zadání: Naleznete-li v zadání nějakou chybu nebo nepřesnost, dejte mi prosím vědět na Fóru předmětu nebo mi pošlete email na krivka@fit.vutbr.cz.

2013-01-21: Vznik zadání a vytvoření konkrétních zadání úloh

2013-02-04: Zveřejnění zadání, registrace bude spuštěna 11. února večer.

Rámcové zadání: Projekt je individuální a sestává se z naprogramování skriptů pro **dvě úlohy**, které si student vybírá z možných variant (viz sekce 2). První skript bude povinně implementován v jazyce **Perl** a druhý v jazyce **Python 3** (možnost použití technik OOP v kombinaci s dalšími programovacími paradigmaty).

Skripty jsou aplikace pro příkazovou řádku (tzv. filtry) se společným základem zadání (viz sekce 3). Každá úloha (viz sekce 7) má identifikátor ze tří písmen (např. CHA, CST, CSV, DKA, JSN, MKA, SYN, XQR, XTD).

2 Organizační informace

Kromě implementace úlohy a vytvoření dokumentace k ní je třeba dodržet také řadu formálních požadavků definovaných v této části.

Termíny odevzdání:

První úloha v pátek **15. března** 2013 do 23:59:59,

Druhá úloha v pátek **19. dubna** 2013 do 23:59:59.

Dodatečné informace a konzultace k projektu z IPP:

- Často kladené otázky k projektu z předmětu IPP, viz Wiki předmětu v IS FIT
- Fórum předmětu IPP pro ak. rok 2012/2013, témata `IPP: Projekt.*`
- U cvičících zodpovědných za jednotlivé úlohy (viz zadání konkrétních úloh)
- Zbyněk Křivka: v konzultačních hodinách nebo po dohodě e-mailem (uvádějte předmět začínající `"IPP:"`), viz `http://www.fit.vutbr.cz/~krivka`
- Dušan Kolář: po dohodě e-mailem (uvádějte předmět začínající `"IPP:"`), viz `http://www.fit.vutbr.cz/~kolar` (jen v závažných případech)

Pokud máte jakékoliv dotazy, problémy, či nejasnosti ohledně tohoto projektu, tak po přečtění často kladených otázek a fóra předmětu (využívejte i možnosti hledání na fóru), neváhejte napsat na fórum (u obecného problému, jenž se potencionálně týká i Vašich kolegů) či kontaktovat zodpovědného cvičícího (viz zadání úlohy), garanta projektu (v případě individuálního problému, e-mail: krivka@fit.vutbr.cz), nebo nouzově garanta předmětu (e-mail: kolar@fit.vutbr.cz), kdy do předmětu uveďte na začátek řetězec `"IPP:"`. Na problémy zjištěné v řádu hodin až jednotek dní před termínem odevzdání některé části projektu nebude brán zřetel. Začněte proto projekt řešit s dostatečným předstihem.

Forma a způsob odevzdání: Projekt se odevzdává individuálně prostřednictvím IS FIT v předmětu IPP (odevzdání emailem není možné).

V rámci termínu „Projekt (Registrace + 1. úloha v Perlu)“ se v IS FIT přihlásíte na variantu projektu (dvě konkrétní úlohy s určením konkrétních implementačních jazyků) a do tohoto termínu také odevzdáte archív pro první úlohu v jazyce Perl. Po termínu odevzdání 1. úlohy bude otevřen termín „Projekt (2. úloha v Pythonu 3)“ pro odevzdání archívu pro druhou úlohu v jazyce Python 3.

Skript a dokumentace budou součástí řešení každé úlohy. Každá úloha bude odevzdána ve zvláštním archívu, kde budou soubory k dané úloze zkomprimovány programem TAR+GZIP či ZIP do jediného archívu pojmenovaném `xlogin00-XYZ.zip` nebo `xlogin00-XYZ.tgz`, kde `xlogin00` je Váš login a `XYZ` je identifikátor úlohy implementované a dokumentované v daném archívu. Velikost každého archívu bude omezena informačním systémem (pravděpodobně na 1 MB). Archív nesmí obsahovat speciální či spustitelné soubory. Názvy všech souborů mohou obsahovat pouze písmena anglické abecedy, číslice, tečku, pomlčku a podtržítko. Archív nebude obsahovat žádné adresáře výjma adresářů pro přiložené vlastní knihovny nebo povolené knihovny, které nejsou nainstalovány na serveru Merlin. Skript bude umístěn v kořenovém adresáři odevzdaného archívu. Po rozbalení archívu na serveru Merlin bude možné skript spustit.

Pokud některý pokyn nebude dodržen, může být projekt považován za neodevzdaný a bude hodnocen nula body!

Hodnocení:

Výše základního bodového hodnocení projektu v předmětu IPP je **maximálně 20 bodů**. Navíc lze získat maximálně 5 bonusových bodů.

Každý skript včetně dokumentace bude hodnocen **10 body**. Z toho bude dokumentace až za 2 body, avšak maximálně 30 % hodnocení skriptu (tedy v případě neodevzdání funkčního skriptu bude samotná dokumentace hodnocena 0 body). Body za úlohu se zaokrouhlují na celé body.

Za řešení je možné získat až 25 % bonusových bodů, kdy se Váš maximální bonusový zisk počítá z aktuálního základního bodového zisku (např. při získání 10 bodů za úlohu může být bonus celkem až 2,5 bodu, při získání 8 bodů může být bonus až 2 body, hranice se zaokrouhluje nahoru). Takový bonus je možné získat pouze za obzvláště kvalitní a podařené řešení některého z registrovaných rozšíření (viz zadání jednotlivých úloh v sekci 7), kvalitativně nadprůměrnou účast na fóru projektu apod.

Skripty budou spouštěny na serveru Merlin příkazem: *interpret skript parametry*, kde *interpret* bude `perl` nebo `python3`, *skript* a *parametry* závisí na dané úloze. Hodnocení funkčnosti bude zajišťovat automatizovaný nástroj. Kvalitu dokumentace, komentářů a strukturu zdrojového kódu budou hodnotit cvičící.

Podmínky pro opakující studenty ohledně případného uznání hodnocení loňského projektu najdete v často kladených otázkách na Wiki stránkách předmětu.

Projekt je typicky hodnocen jako neodevzdaný, pokud:

- nebude doručen včas (viz termíny odevzdání jednotlivých úloh; opoždění o několik hodin je řešeno jako pozdní odevzdání se znatelnou bodovou srážkou až 1 bod za hodinu zpoždění, viz FAQ)
- nebude doručen v předepsaném formátu (viz požadavky na formu a způsob odevzdání)
- nebude možné interpretovat skript tak, jak je uvedeno v zadání (např. je nutný další zásah do skriptu, ruční nastavení přístupových cest, apod.)
- nebude skript vykazovat ani základní funkčnost požadovanou zadáním
- půjde o plagiát v jakékoliv podobě[1]; u pozitivního nálezu dojde kromě anulování hodnocení také k neudělení zápočtu (nebo dodatečnému odebrání včetně bodů ze zkoušky; poloautomatické hledání plagiátů může být provedeno kdykoli před definitivním uzavření hodnocení) a zvážení podání návrhu na disciplinární řízení
- zdrojový text skriptu nebo dokumentace budou obsahovat vulgarismy

Pro účely kontroly některých formálních požadavků lze využít skript `is_it_ok.sh` dostupný v *Souborech* předmětu IPP.

Pokusné odevzdání: Pro zvýšení motivace studentů pro včasné vypracování úloh nabízíme koncept nepovinného pokusného odevzdání. Výměnou za pokusné odevzdání do uvedeného termínu (vždy týden před finálním termínem) dostanete zpětnou vazbu v podobě zařazení do některého ze čtyř rozmezí hodnocení (0-5%, 5-20%, 20-80%, 80-100%). Bude-li vaše pokusné odevzdání v prvním rozmezí hodnocení, máte možnost osobně konzultovat důvod, pokud jej neodhalíte sami. U ostatních rozmezí nebudou detailnější informace poskytovány.

Pokusné odevzdání bude relativně rychle vyhodnoceno automatickými testy a studentům zaslána orientační informace o správnosti pokusně odevzdané úlohy z hlediska části automatických testů (tj. nebude se jednat o finální hodnocení; proto nebudou sdělovány ani body či přesnější procentuální hodnocení). Využití pokusného termínu není povinné, ale jeho nevyužití může být vzato v úvahu v případě reklamace hodnocení projektu.

Formální požadavky na pokusné odevzdání jsou totožné s požadavky na finální termín a odevzdání se bude provádět do speciálních termínů „Projekt (Pokusné odevzdání 1. úlohy)“ do **8. března 2013** a „Projekt (Pokusné odevzdání 2. úlohy)“ do **12. dubna 2013**. Není nutné zahrnout dokumentaci, která spolu s rozšířeními pokusně vyhodnocena nebude.

[1]Mimo jiné je za plagiát považováno i dílo obsahující shody pouze v částech s jiným dílem, kde není prokazatelně stejný autor (včetně převzetí zdrojových kódů z veřejných zdrojů). Například převzetí cizího kódu, změna jmen identifikátorů a komentářů je jednoznačně plagiátorstvím. Stejně tak krádež zdrojového kódu je považována za důvod k zahájení disciplinárního řízení. POZOR! Umístění vaší implementace na veřejně přístupnou lokaci může být zneužito i bez Vašeho vědomí, což se těžko prokazuje.

Varianty projektu: Projekt se skládá z několika úloh (viz sekce 7), z nichž si student vybírá dvojici úloh (první implementuje v Perlu, druhou v Pythonu 3) z následujících povolených variant:

1. CHA+XQR (C Header Analysis v Perlu, XML Query v Pythonu 3)
2. CST+CHA (C Stats v Perlu, C Header Analysis v Pythonu 3)
3. CST+XTD (C Stats v Perlu, XML2DDL v Pythonu 3)
4. DKA+CSV (Determinizace konečného automatu v Perlu, CSV2XML v Pythonu 3)
5. JSN+XQR (JSON2XML v Perlu, XML Query v Pythonu 3)
6. JSN+XTD (JSON2XML v Perlu, XML2DDL v Pythonu 3)
7. SYN+DKA (Zvýraznění syntaxe v Perlu, Determinizace konečného automatu v Pythonu 3)
8. SYN+MKA (Zvýraznění syntaxe v Perlu, Minimalizace konečného automatu v Pythonu 3)
9. XQR+CST (XML Query v Perlu, C Stats v Pythonu 3)
10. XTD+SYN (XML2DDL v Perlu, Zvýraznění syntaxe v Pythonu 3)

3 Společný základ zadání všech úloh

1. Implementujte v daném programovacím jazyce programové vybavení (skript) pro řešení přidělené úlohy.
2. Toto programové vybavení doplňte o vhodnou implementační dokumentaci.

3.1 Dokumentace

Implementační dokumentace (dále jen dokumentace) musí být stručným a uceleným průvodcem **vašeho způsobu řešení** zadané úlohy. Bude vytvořena ve formátu **PDF**. Jakékoliv jiné formáty dokumentace než PDF budou ignorovány, což povede ke ztrátě bodů za dokumentaci. Dokumentaci je možné psát buď česky (s diakritikou, formálně čistě), nebo anglicky (formálně čistě). Dokumentace může být doplněna o graf pro použitý konečný automat, pravidla Vámi vytvořené gramatiky nebo popis jiných speciálních technik a algoritmů. Nicméně **nesmí obsahovat ani částečnou kopii zadání**.

Dokumentace bude dále popisovat způsob a Váš specifický postup řešení (např. řešení sporných případů nedostatečně upřesněných zadáním, konkrétní řešení rozšíření) a celkovou filozofii návrhu. **Rozsah** celého dokumentu je minimálně jedna celá strana a maximálně dvě strany A4 (10-bodové písmo Times New Roman a případně Courier pro identifikátory a skutečně krátké úryvky kódu; nevkládejte žádnou zvláštní úvodní stranu, obsah ani závěr) pro řešení každé úlohy. V rozumné míře je vhodné používat nadpisy první a druhé úrovně (12-bodové a 11-bodové písmo Times New Roman) pro vytvoření logické struktury dokumentace.

Nadpis a hlavička dokumentace bude na prvních třech řádcích obsahovat:

```
Dokumentace úlohy %XYZ%: %task% v %language% do IPP 2012/2013
Jméno a příjmení: %name_surname%
Login: %xlogin00%
```

kde `%XYZ%` je identifikátor Vaší úlohy, `%task%` jméno Vaší úlohy, `%language%` implementační jazyk dané úlohy (Perl nebo Python 3), `%name_surname%` Vaše jméno a příjmení a Váš login `%xlogin00%`.

Dokumentace bude v kořenovém adresáři odevzdaného archívu a pojmenována `XYZ-doc.pdf`, kde `XYZ` je identifikátor úlohy velkými písmeny.

V rámci dokumentace bude hodnoceno i komentování zdrojového kódu skriptu (minimálně každá funkce a modul by měly mít svůj komentář o jejich funkci a parametrech; u složitějších funkcí dokumentujte i omezení na vstupy či výstupy). Každý skript (i případný pomocný) musí v řádkovém komentáři na třetím řádku[2] zdrojového textu obsahovat zakomentovaný dvojtečkou oddělený identifikátor úlohy a login autora skriptu, čímž stvrzujete výhradní autorství daného skriptu. Například (v jazyce Perl i Python 3):

```
#XQR:xnovak99
```

3.2 Programová část

Zadání úloh vyžadují implementaci textového filtru nebo jiné aplikace příkazové řádky[3], která má vstupní parametry a je definováno, jakým způsobem manipuluje se vstupy a výstupy. Skript nesmí spouštět žádné další procesy či dokonce jiné příkazy operačního systému. Veškerá chybová hlášení, varování a ladicí výpisy směřujte pouze na standardní chybový výstup, jinak pravděpodobně nedodržíte zadání kvůli modifikaci definovaných výstupů (ať již do externích souborů nebo do standardního výstupu). Jestliže proběhne činnost skriptu bez chyb, vrací se návratová hodnota `0` (nula). Jestliže došlo k nějaké chybě, vrací se výstupní návratová hodnota větší jak nula. Některé typy chyb mají závazné chybové návratové hodnoty:

1. špatný formát parametrů skriptu nebo byla použita zakázaná kombinace parametrů;
2. neexistující zadaný vstupní soubor nebo chyba otevření zadaného vstupního souboru pro čtení;
3. chyba při pokusu o otevření zadaného výstupního souboru pro zápis (např. kvůli nedostatečnému oprávnění);
4. chybný formát vstupního souboru (pokud nebyla bezchybovost vstupu zaručena v konkrétní úloze nebo nebyla specifikována jiná návratová hodnota pro konkrétní chybu v konkrétní úloze);

10. - 99. návratové kódy chyb specifických pro jednotlivé úlohy.

100. - 255. ostatní typy chyb, které nejsou postihnuty žádnou z předchozích variant.

Pokud zadání konkrétní úlohy nestanoví jinak, tak veškeré vstupy a výstupy jsou v kódování UTF-8. Implicitní nastavení `locale` **na serveru Merlin bude pro účely projektu z IPP změněno**[4] na `LC_ALL=cs_CZ.UTF-8`. V případě XML souborů není třeba uvažovat deklarace jmenných prostorů. Všechna vstupní XML budou v tomto smyslu nekolizní a nebudou jmenné prostory využívat. Nicméně deklarace se v XML souboru vyskytnout může a tu potom ignorujte. Je-li výstupem soubor ve formátu XML, tak bude většinou porovnáván s referenčními výsledky pomocí nástroje A7Soft JExamXML[5].

[2]Na první řádek se typicky píše hlavička pro interpret příkazového řádku a na třetím řádku se v Pythonu typicky uvádí informace o kódování zdrojového textu.

[3]konzolové aplikace

[4]Změna prostředí je nezbytná, aby bylo možné používat a správně zpracovávat parametry příkazové řádky v UTF-8. Pro správnou funkčnost je třeba mít na UTF-8 nastavenu i znakovou sadu konzole (např. u programu PuTTY v kategorii *Window.Translation* nastavíte *Remote character set* na UTF-8). Pro změnu ovlivňující aktuální sezení lze využít příkaz `export LC_ALL=cs_CZ.UTF-8`

[5]Parametry nastavení nástroje A7Soft JExamXML při vyhodnocování budou zveřejněny na fóru k projektu.

Každý skript se bude jmenovat podle identifikátoru úlohy převedeného na malá písmena a příponu[6] bude mít podle daného skriptovacího jazyka dle platných zvyklostí (Perl (`.pl`), Python 3 (`.py`)). Vyhodnocení skriptů bude prováděno na serveru `Merlin` s aktuálními verzemi interpretů (dne 21. 1. 2013 byl na tomto serveru nainstalován `perl` verze 5.8.8 a `python3`[7] verze 3.2.3).

K řešení lze využít standardně předinstalované knihovny obou jazykových prostředí na serveru `Merlin`. V případě využití jiné knihovny kromě knihovny podporující načítání formátů XML, popř. CSV či JSON je třeba konzultovat s garantem projektu (především z důvodu, aby se řešení projektu použitím vhodné knihovny nestalo zcela triviálním).

Každý skript bude pracovat s těmito společnými parametry:

- `--help` vypíše na standardní výstup nápovědu skriptu, kterou lze převzít ze zadání (lze odstranit diakritiku, případně přeložit do angličtiny dle zvoleného jazyka dokumentace). Tento parametr nelze kombinovat s žádným dalším parametrem, jinak skript ukončete s chybou.

- `--input=filename` zadaný vstupní soubor může být zadán relativní cestou[8] nebo absolutní cestou; v případě, že by název či cesta obsahovala mezeru nebo uvozovky, tak musí být uvedena celá cesta i se jménem souboru v uvozovkách (uvozovky ve jméně souboru není třeba uvažovat); chybí-li tento parametr, tak je uvažován standardní vstup.

- `--output=filename` zadaný výstupní soubor může být zadán relativní cestou nebo absolutní cestou; v případě, že by název či cesta obsahovala mezeru, tak musí být uvedena cesta i se jménem souboru v uvozovkách (uvozovky ve jméně souboru není třeba uvažovat); chybí-li tento parametr, tak je výstup přesměrován na standardní výstup. **Existuje-li již výstupní soubor, bude bez varování přepsán, jinak bude vytvořen nový soubor.**

Kombinovatelné parametry skriptů jsou odděleny alespoň jedním bílým znakem a mohou být uváděny v libovolném pořadí, pokud nebude řečeno jinak. Zakázané kombinace parametrů ukončí skript s chybou. Pokud to nekoliduje se zadáním (např. se zakázanou kombinací parametrů) a domluvíte-li se se zodpovědným cvičícím dané úlohy, je možno implementovat i další parametry skriptů nad rámec zadání. Není-li u úlohy specifikováno jinak, vstupní parametry se nesmí opakovat! Například nepovolujeme zadat dvakrát stejný nebo i různý soubor parametrem `--input`.

Dle konvencí unixových systémů budou v některých úlohách zástupné zkrácené (s jednou pomlčkou) a dlouhé parametry (se dvěma pomlčkami), které lze se zachováním sémantiky zaměňovat (tzv. alias parametry).

Reference:

- Extensible Markup Language (XML) 1.0. W3C. World Wide Web Consortium [online]. 5. vydání. 26. 11. 2008 [cit. 2012-02-02]. Dostupné z: `http://www.w3.org/TR/xml/`

- A7Soft JExamXML is a java based command line XML diff tool for comparing and merging XML documents. c2012 [cit. 2012-02-07]. Dostupné z: `http://www.a7soft.com/jexamxml.html`

[6] Pomocné skripty nebo knihovny mohou mít jinou příponu. Např. `.pm` pro Perl.

[7] Upozornění: Na serveru Merlin je třeba dodržet testování příkazem `python3`, protože pouhým `python` se spouští stará nekompatibilní verze! Python 3.x není zpětně kompatibilní s verzí 2.x!

[8] Relativní cesta nebude obsahovat zástupný symbol `~`.

4 Registrovaná rozšíření

V případě implementace některých registrovaných rozšíření za bonusové body bude odevzdaný archív obsahovat soubor `rozsireni`, ve kterém uvedete na každém řádku identifikátor jednoho implementovaného rozšíření[9] (řádky jsou ukončeny znakem `LF`). V průběhu řešení mohou být zaregistrována nová rozšíření úlohy za bonusové body (viz fórum předmětu IPP). Cvičícím můžete přes fórum zasílat návrhy na nová netriviální rozšíření, která byste chtěli navíc implementovat. Cvičící rozhodnou o přijetí/nepřijetí rozšíření a hodnocení rozšíření dle jeho náročnosti včetně přiřazení unikátního identifikátoru. Implementovaná rozšíření neidentifikovaná v souboru `rozsireni` nebudou hodnocena.

5 Nejčastější chyby v minulých letech

Tato sekce popisuje nejčastější chyby (neseřazeno) studentů v projektech z minulých let, kterým se lze snadno vyhnout.

1. Nedodržování návratové hodnoty. Studenti si často pletli standardní výstup a návratovou hodnotu vykonání skriptu. V případě nulové návratové hodnoty je třeba mít i správný výstup. Obsah standardního chybového výstupu se nehodnotí, ale může být relevantní při reklamaci hodnocení projektu.

2. Laxní až vágní výstupy. Hlavním hodnotícím prostředkem je automatický překlad, spouštění nad mnoha testy a na výsledcích založené hodnocení. Pokud jsou vaše výsledky automaticky neporovnatelné s referenčními, nebudou Vám body přiděleny. Pro ověření pochopení základních požadavků na skripty jsou k dispozici veřejné testovací sady.

3. Pokud Váš skript neumí otevřít a načíst soubor ze zadané relativní (např. začínající předponou „`./`“) či absolutní cesty a uložit výsledek do zadané relativní nebo absolutní cesty, nelze Vaši úlohu vůbec hodnotit. Některé úlohy jsou více testovány se zadáním relativních cest a některé zase s využitím absolutních cest, takže vždy podporujte všechny způsoby načítání dat (především ze souborů), abyste se vyvarovali zbytečně velkým bodovým srážkám.

4. Projekt byl nefunkční kvůli neotestování na serveru Merlin. Většina studentů vytváří poslední verzi projektu na poslední chvíli před odevzdáním a neprovede pak otestování na referenčním serveru Merlin, kde bude také probíhat opravování. Dodatečné zásahy do kódu projektu jsou však přísně penalizovány včetně těch celkem banálních jako změna kódování, protože testování a ladění výsledné aplikace je také součást tohoto projektu.

5. Studenti podcenili své znalosti daného implementačního jazyka a jeho povolených knihoven.

6. Nepochopení lexikálních pravidel jazyka Python. Ve zdrojovém textu nelze mixovat odsazení příkazů pomocí mezer a pomocí tabulátorů. Doporučujeme zvolit vhodný editor, který vám umožní tyto problémy ohlídat a všechny tabulátory nahrazovat mezerami.

7. Implementace skriptu ve špatné verzi interpretu (např. v Python 2.7 místo Python 3.2)

8. Nekvalitní dokumentace. Dokumentaci čteme podstatně pozorněji než potom zdrojový kód, který pouze projdeme pro ohodnocení přehlednosti a základní struktury.

[9] Identifikátory rozšíření jsou uvedeny u konkrétního rozšíření tučně (viz zadání jednotlivých úloh).

6 Obecné slovo závěrem

Na stránkách předmětu IPP a v příspěvcích na fóru předmětu IPP se budou v prvních týdnech řešení projektu objevovat doporučení a návody na vhodné řešení zadání. Tato doporučení však nebudou závazná a jejich nedodržení je vhodné obhájit v dokumentaci projektu.

Projekt z předmětu IPP si klade za cíl vás seznámit se dvěma skriptovacími jazyky a možnostmi kombinování různých programovacích paradigmat. Dále bychom rádi studenty přesvědčili o vhodnosti různých jazyků pro různé úlohy. Formou diskuzí na fóru potom budeme částečně simulovat komunikaci vás, návrhářů a programátorů, se zadavatelem či zákazníkem.

Na závěr opět zdůrazněme, že všechny úlohy budou opravovány automatizovaně a tudíž je nezbytné stoprocentní splnění formálních požadavků jak na formáty vstupů a výstupů, tak i na samotné transformace. Pro usnadnění precizního pochopení zadání bude zveřejněna základní sada příkladů vstupů a výstupů jednotlivých skriptů. Je ale především na Vás samotných, abyste si vytvořili i vlastní rozsáhlé balíky testů.

7 Zadání jednotlivých úloh

Zadání jednotlivých úloh jsou nově umístěna ve zvláštních souborech:

1. cha.pdf: C Header Analysis
2. cst.pdf: C Stats
3. csv.pdf: CSV2XML
4. dka.pdf: Determinizace konečného automatu
5. jsn.pdf: JSON2XML
6. mka.pdf: Minimalizace konečného automatu
7. syn.pdf: Zvýraznění syntaxe
8. xqr.pdf: XML Query
9. xtd.pdf: XML2DDL

Opakující studenti POZOR!!! Zadání většiny úloh bylo aktualizováno. Na vaše omyly ohledně implementace zadání z minulého roku nebude brán žádný zřetel.