Bakalářské zkoušky (příklady otázek)

léto 2017

1 Algoritmy: grafové algoritmy (3 body)

1. Definujte topologické uspořádání vrcholů acyklického orientovaného grafu.
2. Popište algoritmus, který toto uspořádání nalezne. Rozeberte jeho časovou složitost.
3. Ukažte, jak pomocí topologického uspořádání spočítat, kolik existuje cest ze zadaného vrcholu u do zadaného vrcholu v.

2 Architektura počítačů a operačních systémů (3 body)

Předpokládejte běžný operační systém jako Windows 10 nebo Linux, běžící na počítači s jedním jednojádrovým procesorem. Dále předpokládejte, že máme spustitelné soubory dvou aplikací `P1.EXE` a `P2.EXE`. Zdrojový kód obou aplikací v jazyce Pascal vypadá zhruba takto (programy `P1.EXE` a `P2.EXE` se liší pouze implementací funkce `Transform`):

```
function Transform (data : string) : string;
begin
  ...
end;

var
   line : string;

begin
   while not (eof (input)) do begin
     ReadLn (line);
     line := Transform (line);
     WriteLn (line);
   end;
end.
```

V aktuálním adresáři máme k dispozici textový soubor `IN.TXT` o velikosti 1 GiB. Oba programy spustíme na příkazové řádce jedním příkazem:

```
P1.EXE < IN.TXT | P2.EXE > OUT.TXT
```

1. Dá se předpokládat, že za této situace běží programy P1 a P2 alespoň nějakou chvíli opravdu současně? Vysvětlete proč, případně jak je toho docíleno. Dále schematicky načrtněte do časové osy jaký kód (kód programů P1 a P2, kód operačního systému či jiné komponenty) bude asi procesor provádět v nějaké zajímavé části zpracování výše uvedeného příkazu – zajímá nás ta část výpočtu, kde bude vidět alespoň část běhu programu P1 a současně také alespoň část běhu programu P2.
2. Bylo by možné programy P1 a P2 (a případně výše uvedený příkaz) upravit tak, aby program P1 svůj výstup zapisoval rovnou do proměnné `line` programu P2 (a tedy komunikace mezi P1 a P2 by z pohledu P1 probíhala přímými zápisy do paměti, nikoliv přes standardní výstup a vstup)? Vysvětlete proč, případně načrtněte potřebnou úpravu P1 a P2.
3. Pouze pro tuto část předpokládejte, že funkce `Transform` v P1 vypadá tak, že na začátek každé řádky přidává znak plus, funkce `Transform` v P2 na konec každé řádky přidává znak vykřičník. Co by se stalo, pokud bychom na výše uvedené příkazové řádce místo souboru `IN.TXT` předložili programu P1 nějaký binární soubor obsahující obrázek ve formátu BMP nebo JPG nebo PNG? Popište stručně chování programů P1 a P2 v takové situaci. Došlo by k nějaké chybě při spuštění nebo běhu programu P1 nebo P2? Vysvětlete proč.

3 Programovací jazyky: principy objektového návrhu (3 body)

Předpokládejte, že navrhujete objektový model editoru pro 2D vektorovou grafiku. Editor má dovolit uživateli vybírat z nabídky různých typů grafických tvarů (přímka, čtverec, kružnice ...) – tvar se nejprve vybere podle názvu a pak kliknutím umístí na plochu.

1. Navrhněte objektový model (hierarchii tříd, rozhraní objektů, signatury metod) umožňující snadno implementovat nové druhy tvarů. Rozhraní by mělo dovolit registraci každého typu grafického tvaru v editoru a dále vykreslování tvarů.
2. Doplňte model tak, aby uživatel editoru mohl zadávat parametry tvarů. Počet a názvy parametrů jsou pro každý typ tvaru jiné, typ parametru je vždy double.

Pro řešení úlohy si vyberte jeden z jazyků Java, C++, C#. Při hodnocení odpovědi nebudou uvažovány drobné syntaktické chyby, ale obecně by použité konstrukce měly odpovídat zvolenému programovacímu jazyku.

4 Základy sítí: SMTP (3 body)

Napište sekvenci odeslaných a přijatých zpráv protokolu SMTP při odesílání emailu na server `muj.smtpserver.edu` z klienta `lab.skola.edu`. Odesílatel má adresu `alice@skola.edu`, příjemce má adresu `bob@domecek.com`, subject je „Navsteva“, text je „Prijdu ve 2, A“, odesílá se právě teď.

Vedle základní sekvence SMTP zpráv s potřebnými parametry se očekává obecný popis relace, tedy zejména vysvětlení, jakým transportním protokolem a jak se zasílají jednotlivé zprávy. Drobné syntaktické chyby v odpovědi nehrají roli.

5 Logika (3 body)

1. Uveďte definici pojmu *model teorie* T. Uveďte znění věty o kompaktnosti predikátové logiky.
2. Nechť $\underline{\mathbb{N}} = \langle \mathbb{N}, S, +, \cdot, 0, \leq \rangle$ je standardní model přirozených čísel jazyka aritmetiky $L = \langle S, +, \cdot, 0, \leq \rangle$ s rovností. Pro $n \in \mathbb{N}$ označme $\underline{n}$ term $S(S(\cdots(S(0)\cdots)$, tzv. *n-tý numerál*, kde S je aplikováno n-krát. Uvažme teorii

$$T = \mathrm{Th}(\underline{\mathbb{N}}) \cup \{\underline{n} < c \mid n \in \mathbb{N}\},$$

kde $\mathrm{Th}(\underline{\mathbb{N}})$ označuje množinu všech pravdivých sentencí v $\underline{\mathbb{N}}$ a c je nový konstantní symbol. Má teorie T model? Uveďte zdůvodnění.

6 Norma (2 body)

Definujte pojem *norma* vektoru.

7 Lineární zobrazení (3 body)

Nad tělesem reálných čísel uvažme matici

$$A = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}.$$

Rozhodněte, zda lineární zobrazení $x \mapsto Ax$:

1. je prosté,
2. je na,
3. zobrazuje lineárně nezávislou množinu na lineárně nezávislou množinu.

8 Diagonalizovatelnost (3 body)

Určete, jestli následující matice jsou diagonalizovatelné (v teorii vlastních čísel, tedy podobné nějaké diagonální matici):

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 4 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 3 \\ 3 & 4 & 3 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}.$$

Odpovědi zdůvodněte, rozklad počítat nemusíte.

9 Limita posloupnosti (2 body)

Definujte pojem limita posloupnosti.

10 Taylorův polynom (3 body)

Napište Taylorův polynom druhého stupně pro funkci danou předpisem $f(x) = \operatorname{tg} x$ (funkce tangens) v okolí bodu 0. Výpočet zdůvodněte.

11 Integrál (4 body)

Spočtěte hodnotu

$$\int_0^1 \frac{1}{e^x + 1} dx \ .$$

Zmiňte, jaké věty při výpočtu používáte.

12 Souvislost grafů (3 body)

1. Napište definici tahu v grafu.
2. Za jakých podmínek existuje v grafu G tah, který obsahuje všechny jeho hrany?
3. Napište definici hranové k-souvislosti.
4. Ukažte, že je-li graf G souvislý a má všechny stupně sudé, pak je hranově 2-souvislý.

13 Pravděpodobnost a střední hodnota (4 body)

Uvažme pravděpodobnostní prostor slov délky 8 nad abecedou $\{a, b\}$, každé se stejnou pravděpodobností. Nechť A označuje jev: "Posloupnost $(x_1, \ldots, x_8)$ obsahuje podposloupnost aaa." a náhodná veličina F_A je rovna počtu výskytů podposloupnosti aaa. Analogicky, jako B označme jev: "Posloupnost $(x_1, \ldots, x_8)$ obsahuje podposloupnost aba." a F_B je rovna počtu výskytů podposloupnosti aba.

(Posloupnost $(x_1, \ldots, x_8)$ *obsahuje podposloupnost* rst, pokud existují indexy $1 \leq i < j < k \leq 8$ takové, že $x_i = r, x_j = s, x_k = t$. Tedy někde v posloupnosti je r, za ním, ne nutně bezprostředně, je s a o něco dále t. Například $abba$ obsahuje podposloupnost aba, ale ne podposloupnost bab.)

1. Určete pravděpodobnosti jevů A a B.
2. Spočtěte střední hodnoty náhodných veličin F_A a F_B.

14 Izomorfismus grup (3 body)

Uvažte následující grupy $\mathbb{Z}_{12}, \mathbb{Z}_2 \times \mathbb{Z}_6, \mathbb{Z}_3 \times \mathbb{Z}_4, \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_3$. Rozhodněte, které dvojice z nich jsou navzájem izomorfní a odpovědi zdůvodněte.

15 Algoritmy: třídění Merge Sort (3 body)

1. Napište pseudokód třídícího algoritmu Merge Sort.
2. Napište časovou složitost tohoto algoritmu pro n prvků v průměrném a nejhorším případě.
3. Je váš algoritmus stabilní? Proč?

16 Algoritmy a optimalizace: splnitelnost (otázka pro studijní zaměření – 5 bodů)

Na problému splnitelnosti ilustrujte využití lineárního programování při návrhu aproximačních algoritmů. Porovnejte s jinými algoritmy pro splnitelnost.

1. Formulujte lineární program pro instanci problému splnitelnosti (obecně, případně ilustrujte na příkladě), popište algoritmus založený na LP a proveďte jeho analýzu.
2. Formulujte jednoduchý pravděpodobnostní algoritmus pro problém splnitelnosti a určete jeho vlastnosti.
3. Znáte nějaký lepší přístup? (Uvažte kombinaci přístupů z bodu 1. a 2..)

17 Algoritmy a optimalizace: Voroného diagramy (otázka pro studijní zaměření – 4 body)

Definujte Voroného diagram diagram n bodů v rovině.

Dokažte horní odhad jeho složitosti s využitím výsledku o maximální složitosti konvexního mnohostěnu na n vrcholech v 3-dimenzionálním prostoru. (Stačí princip důkazu, technické výpočty je možno vynechat.)

18 Architektura počítačů: stránkování (3 body)

Uvažujte architekturu s podporou stránkování a délkou virtuální i fyzické adresy 32 bitů. K překladu adres je použita dvouúrovňová stránkovací tabulka, každá úroveň tabulky má 1024 položek.

Ve výsledcích můžete v případě potřeby používat i šestnáctkovou nebo dvojkovou soustavu.

1. Rozdělte virtuální adresu $12AB34CD_{16}$ na indexy do obou úrovní stránkovací tabulky a offset v rámci stránky.
2. Nakreslete obsah položek stránkovací tabulky, který bude uvedenou virtuální adresu mapovat na fyzickou adresu $EF7654CD_{16}$. Všechna pole, která procesor použije k překladu této adresy, musí být vyplněna konkrétními hodnotami. Pokud zadání nějakou hodnotu neuvádí, doplňte jí podle vlastního uvážení.
3. Pokud je obsah fyzické paměti na uvažované architektuře ukládán v cache s délkou řádku 64 bajtů, je v pořádku na uvedenou virtuální adresu uložit celočíselnou proměnnou o délce 4 bajty? Vysvětlete, o jakou úvahu se v odpovědi opíráte.

19 Automaty a gramatiky (3 body)

V následujícím příkladě uvažujme standardní binární reprezentaci přirozených čísel, tj. binární slovo $w = w_n w_{n-1} \dots w_0$ reprezentuje přirozené číslo $\sum_{i=0}^{n} w_i 2^i$. Speciálně, prázdné slovo λ reprezentuje přirozené číslo 0.

1. Sestrojte deterministický konečný automat A takový, že

$$L(A) = \{w \in \{0,1\}^* \mid w \text{ reprezentuje přirozené číslo dělitelné 3}\}.$$

2. Napište regulární výraz pro jazyk $L(A)$.
3. Napište gramatiku co nejvyššího (tj. nejméně obecného) typu v Chomského hierarchii generující jazyk $L(A)$. O jaký typ gramatiky se jedná?

20 Databáze (3 body)

Uvažme následující tabulku v relační SQL databázi, která obsahuje (zjednodušená) data rozvrhu na naší fakultě. Jeden řádek odpovídá jednomu záznamu (například přednášce) v rozvrhu. Jeden takový záznam chápeme jako pravidelnou výuku, která se opakuje týdně po celý jeden semestr.

Tabulka `Rozvrh`:

- `Předmět` (kód vyučovaného předmětu, např. „NDBI025“)
- `Název` (název vyučovaného předmětu, např. „Databázové systémy“)
- `Rok` (kalendářní rok, ve kterém začíná daný akademický rok – např. 2016 je akademický rok 2016/17)

- `Semestr` („LS“ nebo „ZS“)
- `Den` (den v týdnu 0 = neděle, 1 = pondělí, ...).
- `Začátek` (čas začátku jako počet minut od půlnoci v daný den v týdnu – např. 640 odpovídá času 10:40)
- `Délka` (délka trvání v minutách)
- `Typ` (typ výuky – „P“ = přednáška, „C“ = cvičení)
- `Místnost` (kód místnosti, kde probíhá výuka – např. „S3“)
- `Učitel` (osobní číslo učitele, který za výuku odpovídá)

Žádný atribut nesmí být `NULL`. Databázové schéma by mělo garantovat, že v jednu dobu začíná v jedné místnosti pouze jedna rozvrhová událost. Schéma již negarantuje, že se události nepřekrývají (tedy čas začátku + délka jedné události může být větší než čas začátku následující události). Schéma také neřeší, zda má jeden učitel v danou dobu pouze jednu výuku.

Závislosti mezi jednotlivými atributy chápejte přirozeně (podle toho, jak skutečně probíhá výuka na naší fakultě). Například platí, že k jednomu předmětu může být libovolný počet přednášek i cvičení a každá taková přednáška či cvičení může mít jiného vyučujícího.

1. Najděte nejmenší vhodný (primární) klíč. Pokud je takových klíčů více, napište jeden libovolný z nich.
2. Určete, jaké normální formy (1NF až 5NF) dané schéma splňuje a stručně zdůvodněte.
3. Upravte dané schéma, aby vyhovovalo BCNF (Boyce-Coddově normální formě).

21 Databáze: B-stromy (3 body)

Postavte nejmenší možný neredundantní B-strom se stupněm vrcholů $N = 4$ pro následující množinu klíčů a nakreslete výsledek:

$$3, 8, 19, 21, 27, 33, 39, 42, 49, 54, 61, 64, 71, 77, 88$$

Na stromě následně vykonejte níže uvedené operace (v daném pořadí) a zakreslete stav stromu po jejich provedení. Pokud je nějaký dílčí krok některé operace možné provést různými způsoby (např. vyjmout největší prvek z levého nebo nejmenší prvek z pravého podstromu), vyberte si jeden libovolný z nich, ale buďte konzistentní v rámci všech úprav (tj. vyberte si jeden způsob na začátku a používejte ho pro všechny operace).

- `INSERT` 43
- `DELETE` 64
- `DELETE` 49

22 Diskrétní modely a struktury: axiom výběru (otázka pro studijní zaměření – 3 body)

1. Zformulujte axiom výběru.
2. Jak souvisí axiom výběru s existencí neměřitelných množin?

23 Diskrétní modely a struktury: párování v grafech (otázka pro studijní zaměření – 3 body)

1. Nechť $K_{a,b,c}$ označuje úplný tripartitní graf s partitami velikosti a, b a c, tj. graf s množinou vrcholů $\{x_1, \ldots, x_a, y_1 \ldots, y_b, z_1, \ldots$ a hranami $x_i y_j$, $y_j z_k$ a $x_i z_k$ pro $1 \leq i \leq a$, $1 \leq j \leq b$ a $1 \leq k \leq c$. Pro která přirozená čísla a, b a c má $K_{a,b,c}$ perfektní párování?
2. Zformulujte Tutteho větu o existenci perfektního párování.
3. Nechť T je strom. Ukažte, že T má perfektní párování právě tehdy, když pro každý vrchol $v \in V(T)$ má $T - v$ právě jednu komponentu liché velikosti.

24 Diskrétní modely a struktury: Ramseyova teorie (otázka pro studijní zaměření – 3 body)

1. Zformulujte Ramseyovu větu pro úplné grafy s hranami obarvenými dvěma barvami.
2. Ukažte, že z každé nekonečné posloupnosti navzájem různých celých čísel lze vybrat nekonečnou monotónní (rostoucí nebo klesající) podposloupnost.

25 Matematická lingvistika: formální jazyky a automaty (otázka pro studijní zaměření – 3 body)

1. Vysvětlete pojem neprojektivity v závislostních stromech.
2. Uveďte příklad nějaké neprojektivní věty.
3. Vysvětlete, do jaké třídy jazyků podle Chomského hierarchie patří přirozené jazyky, které obsahují neprojektivní konstrukce.

26 Matematická lingvistika: základní formalismy pro popis přirozených jazyků (otázka pro studijní zaměření – 3 body)

Popište základní vlastnosti teorie Funkčního generativního popisu. Vysvětlete pojem valence.

27 Matematická lingvistika: základy teorie informace (otázka pro studijní zaměření – 3 body)

V textu pozorujeme slovní druhy slov vyskytujících se bezprostředně před slovesem. Zjistili jsme, že pravděpodobnost výskytu příslovce před slovesem je $1/4$ a pravděpodobnost výskytu substantiva před slovesem je $1/2$.

1. Nechť dále platí, že pravděpodobnost výskytu slovesa v textu je $P(V) = 1/8$. Vypočítejte pravděpodobnost výskytu příslovce následovaného slovesem.
2. Vypočítejte entropii náhodné veličiny, která reprezentuje slovní druh slov před slovesem a nabývá tří různých hodnot N, D, X, kde N je substantivum, D je příslovce a X je libovolný jiný slovní druh.

28 Počítačová grafika: anti-aliasing (otázka pro studijní zaměření – 3 body)

1. Který z parametrů rastrového zobrazovacího zařízení je anti-aliasingem vylepšen? Jak se to pozná na výsledném obrázku?
2. Jak se může anti-aliasing implementovat v klasickém rastrovém vykreslování?
3. Jak se anti-aliasing implementuje v prostředí paprskového zobrazovače (např. ray-tracing)?

29 Počítačová grafika: zobrazování 3D scény na GPU (otázka pro studijní zaměření – 3 body)

1. Jaká 3D primitiva umí GPU přímo zobrazit (v klasickém 3D zobrazovacím řetězci)?
2. Jak se posílají 3D data z aplikace do GPU? Uveďte jen přehledně několik příkladů.
3. Navrhněte rámcově efektivní systém organizace 3D scény a její odesílání do GPU. Předpokládejte, že scéna se skládá z pevné části (např. terén = povrch Země) a z pohybujících se objektů, které se však nedeformují.

30 Počítačová grafika: rastrová a vektorová grafika (otázka pro studijní zaměření – 3 body)

1. Uveďte základní princip rastrové a vektorové grafiky, uveďte příklady grafických formátů, vektorových i rastrových.
2. Jaké jsou výhody vektorové grafiky a pro které operace bychom ji preferovali? Uveďte konkrétní příklady.
3. Naznačte principy komprese rastrových grafických dat.

31 Databáze (3 body)

Uvažme následující tabulku v relační SQL databázi, která obsahuje (zjednodušená) data rozvrhu na naší fakultě. Jeden řádek odpovídá jednomu záznamu (například přednášce) v rozvrhu. Jeden takový záznam chápeme jako pravidelnou výuku, která se opakuje týdně po celý jeden semestr.

Tabulka `Rozvrh`:

- `Předmět` (kód vyučovaného předmětu, např. „NDBI025“)
- `Název` (název vyučovaného předmětu, např. „Databázové systémy“)
- `Rok` (kalendářní rok, ve kterém začíná daný akademický rok – např. 2016 je akademický rok 2016/17)
- `Semestr` („LS“ nebo „ZS“)
- `Den` (den v týdnu 0 = neděle, 1 = pondělí, ...).
- `Začátek` (čas začátku jako počet minut od půlnoci v daný den v týdnu – např. 640 odpovídá času 10:40)
- `Délka` (délka trvání v minutách)
- `Typ` (typ výuky – „P“ = přednáška, „C“ = cvičení)
- `Místnost` (kód místnosti, kde probíhá výuka – např. „S3“)
- `Učitel` (osobní číslo učitele, který za výuku odpovídá)

Žádný atribut nesmí být `NULL`. Databázové schéma by mělo garantovat, že v jednu dobu začíná v jedné místnosti pouze jedna rozvrhová událost. Schéma již negarantuje, že se události nepřekrývají (tedy čas začátku + délka jedné události může být větší než čas začátku následující události). Schéma také neřeší, zda má jeden učitel v danou dobu pouze jednu výuku.

Závislosti mezi jednotlivými atributy chápejte přirozeně (podle toho, jak skutečně probíhá výuka na naší fakultě). Například platí, že k jednomu předmětu může být libovolný počet přednášek i cvičení a každá taková přednáška či cvičení může mít jiného vyučujícího.

1. Najděte nejmenší vhodný (primární) klíč. Pokud je takových klíčů více, napište jeden libovolný z nich.
2. Upravte dané schéma, aby vyhovovalo BCNF (Boyce-Coddově normální formě).
3. Napište SQL dotaz, který z výše popsané tabulky vrátí všechny učitele, kteří v letním semestru 2016/17 mají špatně rozvrženo a mají dvě překrývající se výuky (přednášky nebo cvičení) v jeden čas. Plynule navazující události výuky (tedy pokud jedna výuka učitele končí v čase T a v témže čase T jiná výuka začíná) nepovažujeme za překrývající se.

32 Automaty a gramatiky, překladače (3 body)

Aplikace ukládá konfiguraci ve formátu `json`, přičemž využívá pouze číselné a strukturované položky (do hloubky 3). Konkrétní položky se v jednotlivých verzích liší, vždy je přítomna (právě jednou) položka `"config"/"version"` určující verzi aplikace (celé číslo), např.:

```
{
        "config": {
                "somestruct": {
                        "somenumber": 3
                },
                "version": 7,
                "otherdata": 729
        }
}
```

1. Napište bezkontextovou gramatiku (nejlépe v nějakém standardním formátu jako yacc/bison, ale není nutností), která akceptuje konfigurace verze 10 (a žádné jiné), tj. splňující podmínku `"config"/"version" = 10`. Detaily jako formát řetězců a čísel, whitespace nebo oddělující čárky můžete vynechat.
2. Patří tato gramatika do třídy LL(1)?
3. Lze uvedený jazyk, tedy konfigurace verze právě 10, rozpoznat konečným automatem?

33 Sítě a internetové technologie: firewall a FTP (otázka pro studijní zaměření – 3 body)

Předpokládejme, že jste správce serveru zabezpečeného firewallem, který nemá otevřené žádně nepoužívané porty pro příchozí komunikaci. Vedení firmy se rozhodlo, že na tomto serveru bude provozována nově služba FTP, očekává se spíše malý provoz (max. 10 souběžně připojených klientů). Bylo rozhodnuto, že FTP bude podporovat pouze pasivní datové přenosy. Jaké porty (příp. rozsahy portů) bude nutné otevřít pro příchozí komunikaci ve firewallu a proč?

34 Sítě a internetové technologie: BSD sockety (otázka pro studijní zaměření – 3 body)

Naprogramujte (ideálně v C nebo C++, možné jsou i podobné jazyky jako C# nebo Java) pomocí BSD socketů server přijímající na portu 6677 TCP připojení pro IPv4. Pro každého připojeného klienta přečte binárně 4 bajty bezznaménkového celého čísla zaslaného v pořadí bajtů obvyklém na síti. Toto číslo zvětší o 1, odešle ho klientovi zpět ve stejném binárním formátu a uzavře spojení s klientem.

Při zápisu kódu nejsou důležité drobné syntaktické chyby, rovněž nejsou hodnoceny faktory jako chybějící nepodstatné parametry volaných funkcí, jiné pořadí parametrů a podobně, hlavní je správné zachycení celkové struktury kódu serveru.

35 Sítě a internetové technologie: WiFi (otázka pro studijní zaměření – 3 body)

Vaše domácí síť si dosud vystačila se starším WiFi routerem (podporuje pouze 802.11b/g). Stahování multimediálního obsahu do tabletu se Vám zdá příliš zdlouhavé. Zároveň s tím významně vzrostl počet sousedů v okolí Vašeho bytu se zapnutou WiFi rovněž využívající 802.11b/g.

Rozhodnete se tuto situaci řešit zakoupením nového routeru.

1. Jaké další protokoly 802.11 musí podporovat nový, běžně dostupný router, aby se zmírnil problém se sousedy (včetně zdůvodnění proč)?
2. Jaké další protokoly 802.11 musí podporovat nový, běžně dostupný router, aby se významně zrychlilo stahování multimediálního obsahu (včetně zdůvodnění proč)?

36 Systémové programování: middleware (otázka pro studijní zaměření – 3 body)

Předpokládejte, že následující rozhraní je zpřístupněno pomocí mechanismu volání vzdálených procedur (remote procedure call), tedy implementace tohoto rozhraní je umístěna na vzdáleném serveru a klient toto rozhraní volá skrz automaticky generovanou proxy.

```
public interface RemoteServer {
  public int doComputation (int computationInput);
}
```

1. Napište, jak bude vypadat rozhraní, které nabízí automaticky generovaná proxy na straně klienta.
2. Napište, co bude obsahem zpráv, které si vyměňují klient a server při volání uvedeného rozhraní.
3. Načrtněte kód, který implementuje proxy na straně klienta a její protějšek na straně serveru. Předpokládejte, že máte k dispozici funkce pro spolehlivý přenos zpráv mezi klientem a serverem, formát zprávy můžete zvolit libovolně tak, aby umožnil jednoduchou implementaci.

Poznámka: Pro zápis kódu používá otázka základní konstrukce jazyka Java, nicméně tohoto jazyka se přímo netýká a použité konstrukce je možné vnímat obecně jako běžné konstrukce imperativního objektového jazyka. Případný kód v zápisu odpovědi může být zapsán podobně volně.

37 Systémové programování: správa verzí (otázka pro studijní zaměření – 3 body)

Vysvětlete následující výpisy nástroje git (obsahy commit messages není třeba vysvětlovat):

1. V následujícím, co je řetězec za polem commit (nejde o to jak se dojde ke konkrétní hodnotě, ale co reprezentuje)? Odkud se berou pole author a date a o jaké operaci log informuje?

```
> git log
commit 6d53cefb18e4646fb4bf62ccb6098fb3808486df
Author: Linus Torvalds <torvalds@linux-foundation.org>
Date:   Sun Jun 11 15:51:56 2017 -0700

    compiler, clang: properly override 'inline' for clang
    ...
```

2. V následujícím, co je branch? Co znamená označení souboru „modified“?

```
> git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   README
```

3. V následujícím, co znamená řádek s textem „merge“ a o jaké operaci log informuje (vysvětlení by mělo jít za prostý překlad termínu)?

```
> git log
commit 5ad9345d2321eb1442794098506d136d01cf8345
Merge: 5e38b72 b169c13
Author: Linus Torvalds <torvalds@linux-foundation.org>
Date:   Sun Jun 11 12:02:01 2017 -0700

    Pull randomness fixes from Ted Ts'o
    ...
```

38 Systémové programování: rozhraní pro synchronizaci (otázka pro studijní zaměření – 3 body)

Následující kód používá třídy `Lock` a `Semaphore`, s těmito deklaracemi:

```
public class Lock {
  public Lock () { ... }
  public void lock () { ... }
  public void unlock () { ... }
  ...
}

public class Semaphore {
  public Semaphore (int permits) { ... }
  public void acquire () { ... }
  public void release () { ... }
  ...
}
```

1. Napište stručně semantiku jednotlivých metod tříd `Lock` a `Semaphore`.
2. Pomocí třídy `Lock` je implementována tato třída reprezentující jednoduchý zásobník:

```
public class StackWithLock {

  private final int MAX_SIZE = 1000;
  private int [] storage = new int [MAX_SIZE];
  private int top = 0;

  private Lock lock = new Lock ();

  public void push (int value) {
    lock.lock ();
    storage [top] = value;
    top ++;
    lock.unlock ();
  }

  public int pop () {
    lock.lock ();
    top --;
    int value = storage [top];
    lock.unlock ();
    return (value);
  }
}
```

Rozhodněte, zda je bezpečné volat funkce třídy `StackWithLock` z více vláken a vysvětlete proč. Třída netestuje přetečení ani podtečení implementovaného zásobníku, ve své odpovědi uvažujte odděleně případ, kdy se lze spolehnout na to, že přetečení ani podtečení nikdy nenastane, i případ, kdy nastat mohou.

3. Pomocí třídy `Semaphore` je implementována další třída reprezentující jednoduchý zásobník:

```
public class StackWithSemaphore {

  private final int MAX_SIZE = 1000;
  private int [] storage = new int [MAX_SIZE];
  private int top = 0;

  private Semaphore semaphore = new Semaphore (0);
  private Lock lock = new Lock ();

  public void push (int value) {
    lock.lock ();
    storage [top] = value;
    top ++;
    lock.unlock ();
    semaphore.release ();
  }

  public int pop () {
    semaphore.acquire ();
    lock.lock ();
    top --;
    int value = storage [top];
    lock.unlock ();
    return (value);
  }
}
```

Vysvětlete, v jaké situaci se bude třída `StackWithSemaphore` chovat jinak než `StackWithLock` a proč.

Poznámka: Pro zápis kódu používá otázka základní konstrukce jazyka Java, nicméně tohoto jazyka se přímo netýká a použité konstrukce je možné vnímat obecně jako běžné konstrukce imperativního objektového jazyka. Případný kód v zápisu odpovědi může být zapsán podobně volně.

39 Databáze a web: programování na straně serveru (otázka pro studijní zaměření – 3 body)

Uvažme webovou aplikaci, která zobrazuje katalog produktů administrátorovi e-shopu. Produkty jsou zobrazeny v klasické HTML tabulce. Produktů je mnoho, a proto se v tabulce zobrazuje vždy pouze 50 produktů najednou. Pod tabulkou se nachází odkazy pojmenované $1, 2, \ldots, P$ ($P = \lceil N/50 \rceil$, kde N je celkový počet produktů), které slouží ke „stránkování" tabulky (odkaz 1 nechá zobrazit prvních 50 produktů, odkaz 2 produkty $51 \ldots 100$, atd.).

Produkty mají dále zaškrtávací políčka (checkbox), která umožňují produkty vybrat. Pod tabulkou se nachází tlačítka *"Skladem"*, *"Na objednávku"*, *"Nedostupné"*, která přepnou vybrané (zaškrtnuté) produkty do příslušného stavu dostupnosti. Skript zobrazující stránku s tabulkou je v souboru `index.php`, skript zpracovávající výše popsané operace je v souboru `change_availability.php`.

Pro zjednodušení máte již předimplementované funkce:

- `get_product_count()` vrací celkový počet produktů (int)
- `get_products($offset, $count)` vrací pole indexované DB klíči produktů (int) a hodnoty jsou samotné produkty (object)
- `render_product_row($product)` vrací řetězec obsahující produkt naformátovaný jako řádek tabulky (`<tr>...</tr>`). V první buňce vytvořeného řádku se nachází `<input>` element typu 'checkbox', který má jako hodnotu atributu `name` řetězec `'product[`*id*`]'`, kde *id* je unikátní identifikátor produktu získaný z objektu `$product`, a atribut `value` s hodnotou 1.

1. Napište klíčovou část skriptu `index.php`, aby aplikace mohla být funkční dle popisu výše (za předpokladu doplnění odpovídající implementace skriptu `change_availability.php`) a aby neporušovala specifikaci HTTP protokolu.
2. Stručně popište, co musí provést skript `change_availability.php` poté, co zpracuje požadavek a provede změny v databázi, aby uživateli opět zobrazila tabulka se stejnými produkty (tj. na stejném místě stránkování). Navržené řešení musí fungovat korektně i pokud uživatel během svojí práce nechá znovu načíst stránku prohlížečem nebo použije tlačítka pro navigaci v historii (zpět/vpřed).

40 Databáze a web: REST (otázka pro studijní zaměření – 3 body)

Mějme aplikaci pro ukládání webových záložek (hyperlinků). Hyperlinky jsou ukládány do kategorií, každý hyperlink je právě v jedné kategorii. Struktura kategorií je plochá (kategorie se do sebe nevnořují). Kategorie i hyperlinky jsou interně identifikovány unikátními číselnými ID (které generují sekvence v DB). Každá kategorie a každý hyperlink má popisek, který se zobrazuje uživateli. Příklad vizualizace všech uložených hyperlinků může vypadat následovně:

Vyhledávače

- Google `[https://www.google.cz/]`
- Seznam `[https://www.seznam.cz/]`

Komixy

- Garfield `[https://garfield.com/]`
- Dilbert `[http://dilbert.com/]`
- XKCD `[https://xkcd.com/]`

Navrhněte RESTfull API pro tuto aplikaci, přičemž API musí nabídnout kompletní manipulaci s databází záložek (čtení, modifikace, ...). Popište jednotlivé funkce tohoto API, přičemž konkrétní formát těla HTTP požadavku a HTTP odpovědí nemusíte vysvětlovat detailně – stačí stručně zmínit, co je v těle a v jakém formátování. Nezapomeňte také popsat možné chybové stavy – resp. co je uvedeno v HTTP odpovědi, když dojde k chybě.

API se pokuste vytvořit nejmenší možné, které pokrývá zadání. Není např. nutné mít funkci smazání celé databáze, pokud je možné postupně smazat jednotlivé kategorie. Detaily autentizace a autorizace uživatele při manipulaci se záložkami neřešte.

41 Databáze a web: B-stromy (otázka pro studijní zaměření – 3 body)

Postavte nejmenší možný neredundantní B-strom se stupněm vrcholů $N = 4$ pro následující množinu klíčů a nakreslete výsledek:

$$3, 8, 19, 21, 27, 33, 39, 42, 49, 54, 61, 64, 71, 77, 88$$

Na stromě následně vykonejte níže uvedené operace (v daném pořadí) a zakreslete stav stromu po jejich provedení. Pokud je nějaký dílčí krok některé operace možné provést různými způsoby (např. vyjmout největší prvek z levého nebo nejmenší

prvek z pravého podstromu), vyberte si jeden libovolný z nich, ale buďte konzistentní v rámci všech úprav (tj. vyberte si jeden způsob na začátku a používejte ho pro všechny operace).

- `INSERT` 43
- `DELETE` 64
- `DELETE` 49

42 Softwarové inženýrství: XSLT (otázka pro studijní zaměření – 3 body)

Mějme následující XML dokument:

```
<?xml version="1.0"?>
<database>
  <employee id="z001" supervisor="z002">
    <name>Steaven</name>  <surname>Blamer</surname>
  </employee>
  <employee id="z002">
    <name>Gill</name>  <surname>Bates</surname>
  </employee>
  <employee id="z003" supervisor="z002">
    <name>Satyra</name>  <surname>Mandela</surname>
  </employee>
  <employee id="z004">
    <name>Tim</name>  <surname>Boil</surname>
  </employee>
  <employee id="z005" supervisor="z004">
    <name>Lucas</name>  <surname>Masterpiece</surname>
  </employee>
</database>
```

Tento dokument transformujeme pomocí následující XSLT šablony.

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" indent="yes"/>
  <xsl:template match="/">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <body>
        <ul><xsl:apply-templates/></ul>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="database/employee[not(@supervisor)]">
    <xsl:call-template name="employee"/>
  </xsl:template>
  <xsl:template name="employee">
    <xsl:variable name="id" select="@id"/>
    <li xmlns="http://www.w3.org/1999/xhtml">
      <xsl:value-of select="name" />
      <xsl:text disable-output-escaping="yes"> </xsl:text>
      <xsl:value-of select="surname" />
      <xsl:if test="count(/database/employee[@supervisor=$id]) > 0">
        <ul>
          <xsl:for-each select="//employee[@supervisor=$id]">
            <xsl:call-template name="employee"/>
          </xsl:for-each>
         </ul>
       </xsl:if>
    </li>
  </xsl:template>
  <xsl:template match="/database/employee/*" />
</xsl:stylesheet>
```

Naznačte, jak bude přibližně vypadat výsledné HTML zobrazené v prohlížeči.

43 Softwarové inženýrství: traits a policies (otázka pro studijní zaměření – 3 body)

Uvažme následující (nekompletní) třídu v jazyce C++:

```
template<typename T = float> class Matrix
{
private:
    std::vector<T> data;
    std::size_t M, N;
public:
    // ...
    T operator[](std::size_t i, std::size_t j) { return data[i*M + j]; }
};
```

Jak je vidět, tato implementace matice používá řádkovou reprezentaci dat (řádky matice jsou lineárně seřazeny za sebe v paměti). S použitím class policies a traits upravte kód tak, aby si uživatel této třídy mohl (v době kompilace) vybrat, zda chce mít matici uloženou v paměti po řádcích nebo po sloupcích (případně si mohl naimplementovat vlastní organizaci).

44 Softwarové inženýrství: návrhové vzory (otázka pro studijní zaměření – 3 body)

Připravujeme nový informační systém pro univerzitu, který má být implementován pomocí běžného objektového imperativního jazyka (vyberte si z C#, C++, Java). Existují tři typy uživatelů (*Student*, *Učitel*, *Zaměstnanec*) a je velmi nepravděpodobné, že se někdy v budoucnu objeví další typy uživatelů. Systém musí naopak podporovat snadné přidávání nových typů operací nad těmito uživateli, protože ministerstvo školství a rektorát vydávají velké množství nových vyhlášek, úprav předpisů, atd.

1. Vyberte vhodný návrhový vzor (případně vzory) a načrtněte nejdůležitější třídy a rozhraní (interfaces nebo ryze abstraktní třídy). U klíčových metod napište jejich hlavičky a slovy vysvětlete, co která metoda dělá.
2. Rektorát si vyžádal novou funkci, která spočítá průměrnou dobu studia na univerzitě (předpokládejme, že každý student má u sebe datum zahájení a případně datum ukončení studia). Stručně popište, co je potřeba doplnit do vašeho návrhu (jaké třídy/metody/rozhraní je potřeba naimplementovat, co je potřeba změnit, atd.).
3. Co bude potřeba změnit (doimplementovat), pokud porušíme původní předpoklad ze zadání a vytvoříme nového uživatele typu *Knihovník*?

V této otázce se zaměřme pouze na úroveň implementace business logiky aplikace. Uložení dat do databáze nebo uživatelské rozhraní neřešte.