Bakalářské zkoušky (příklady otázek)

podzim 2018

1 Automaty (3 body)

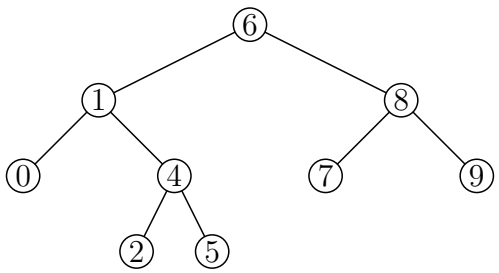
1. Definujte (generativní) gramatiku, jazyk generovaný gramatikou a Chomského hierarchii gramatik.
2. Zkonstruujte gramatiku generující jazyk

$$L = \{uvu^R \mid u \in \{a, b\}^*, v \in \{a\}^*\},$$

   kde u^R označuje zrcadlový obraz slova u.
3. Zařaďte jazyk L do Chomského hierarchie, tj. určete nejmenší třídu Chomského hierarchie, do které jazyk náleží, a dokažte, že žádná gramatika ze slabší třídy Chomského hierarchie nemůže generovat L.

2 Algoritmy a datové struktury (3 body)

1. Definujte AVL stromy a jejich základní vlastnosti (invarianty).
2. Popište průběh operace vložení prvku 3 do následujícího AVL stromu (s dodržením všech invariantů). Vysvětlete jednotlivé kroky.
3. Proveďte diskusi časové složitosti operací (1) vyhledání hodnoty a (2) vložení nového prvku, zejména ve srovnání se základní variantou binárního vyhledávacího stromu.



3 Databáze (3 body)

1. Uvažujte následující relační schéma: AUTOR(ID, Jméno, DatumNarození), KNIHA(Název, PočetStran, IDAutora).

   Vysvětlete pojem "integritní omezení". Doplňte chybějící integritní omezení do uvedeného schématu.

   Zapište následující dotaz v SQL: jména autorů, kteří napsali alespoň deset stostránkových knih.
2. Vysvětlete pojem "funkční závislost na atributech relace". Rozšiřte některou tabulku z prvního příkladu tak, aby v ní byla porušena třetí normální forma.

4 Programovací jazyky (3 body)

Pro tuto otázku si vyberte libovolný mainstreamový objektově orientovaný staticky typovaný jazyk (C++, C# nebo Java) a svůj výběr napište na začátek řešení.

1. Naimplementujte/nadeklarujte vhodné objektové rozhraní (pomocí interfaces nebo abstraktních tříd) pro neorientované grafy $G = (V, E)$ ve své nejběžnější podobě (hrany spojují právě dva vrcholy, mezi dvěma vrcholy vede nejvýš jedna hrana). Motivací je, aby toto rozhraní mohly implementovat různé reprezentace grafů (seznam sousedů, matice sousednosti, ...), zatímco programátor grafových algoritmů (BFS, hledání min. kostry, ...) může napsat svůj kód pouze jednou a bude fungovat se všemi reprezentacemi grafů.

   Vámi definované rozhraní by mělo pokrývat 3 entity (*vrchol*, *hranu* a *graf*), přičemž *vrchol* nabídne přístup ke všem incidentním hranám, *hrana* nabídne přístup ke svým dvěma vrcholům a *graf* umožní přístup k seznamu všech vrcholů a seznamu všech hran. Modifikace grafu v tomto rozhraní neuvažujeme (struktura grafu je pouze pro čtení).

   Každý vrchol a každá hrana má příznak (pro jednoduchost mu říkejme *tag*). Tag u vrcholu může například indikovat, zda byl vrchol již navštíven, tag u hrany může například značit její váhu. Tagy tedy mohou mít obecně různé datové typy, avšak pro danou instanci grafu mají všechny vrcholy stejný typ tagu a všechny hrany taktéž. Váš interface by měl být obecný, aby si programátor mohl při jeho implementaci zvolit typy tagů pro vrcholy a pro hrany. Interface také nabídne pro každý vrchol a hranu vhodné prostředky, jak tag přečíst a nastavit.

2. Představme si implementaci vašeho rozhraní, kde tagy vrcholů značí index komponenty souvislosti (typu `int`, indexované od 0) a tagy hran jsou typu `float` (případně podobného typu ve vašem zvoleném jazyce) a značí jejich délky. Tagy hran jsou na počátku již inicializované, tagy vrcholů musíte nastavit vy.

   Napište kód těla funkce využívající vaše rozhraní, která na vstupu dostane graf a reálné číslo r. Váš kód provede rozklad grafu na komponenty souvislosti tak, že každý vrchol bude mít svůj index komponenty uložený jako tag (konkrétní hodnoty indexů nejsou podstatné, důležité je, že vrcholy ve stejné komponentě mají stejný tag). Hrany, které mají větší délku (tag) než stanovený parametr r, nebereme při určování komponent v potaz (tedy jako by vůbec neexistovaly).

5 Zámky (3 body)

Pro řešení otázky je dostačující následující zjednodušená představa o konceptech jazyka C# (pokud ale skutečné chování využitých konceptů jazyka C# znáte přesněji, tak to není na závadu):

Každá instance třídy `object` v sobě obsahuje právě jeden unikátní zámek. Tento zámek je možno zamknout pomocí operace `lock` (viz řádek 13 kódu níže) – jedná se o běžnou implementaci zámků s podporou pro rekurzivní zamykání a s pasivním čekáním na uvolnění zámku. Otevírací složená závorka za příkazem `lock` odpovídá žádosti o zamčení zámku (řádek 13), párová zavírací složená závorka tohoto bloku pak odpovídá operaci odemčení zámku (řádek 18).

Metoda `Parallel.For(int fromInclusive, int toExclusive, Action<int> body)` vytvoří nová vlákna, jejichž počet odpovídá počtu logických procesorů na cílovém systému. Tato vlákna poté volají metodu předanou jako parametr `body` metody `Parallel.For` (v uvedeném příkladu níže tedy metodu `ProcessFilename`) tak, že pro každé `i` z rozsahu <`fromInclusive`, `toExclusive`) je metoda `body` zavolána právě jednou. Nicméně není definované, ve kterém z vyrobených vláken bude metoda `body` pro konkrétní `i` zavolána, stejně tak není definované pořadí, v jakém budou jednotlivé hodnoty `i` zvoleny. Návrat do volajícího vlákna z metody `Parallel.For` je bezpečně zaručen až v situaci, kdy všechna vytvořená vlákna dokončila všechna volání metody `body` pro všechny hodnoty `i`.

Předpokládejte následující třídu zapsanou v jazyce C#:

```
1  class JpegCounter {
2          public JpegCounter(string[] filenames) { _filenames = filenames; }
3
4          string[] _filenames;
5          int _count;
6          object _globalLock = new object();
7
8          public int CountJpegs() {
9                  _count = 0; Parallel.For(0, _filenames.Length, ProcessFilename); return _count;
10         }
11
12         void ProcessFilename(int i) {
```

```
13                  lock (_globalLock) {
14                          string ext = Path.GetExtension(_filenames[i]).ToUpper();
15                          if (ext == ".JPG" || ext == ".JPEG") {
16                                  _count++;
17                          }
18                  }
19          }
20  }
```

1. V kontextu vícevláknového programu stručně vysvětlete, co to je race condition, a jak může vznikat.
2. Vysvětlete, zda je v uvedeném kódu důležité použití zámku `_globalLock`, a zda by bez jeho použití mohlo docházet k nějakým race conditions. Mohla by metoda `ProcessFilename` bez využití zámku `_globalLock` (tj. po smazání řádků 6, 13 a 18) vracet pro stejný vstup předaný třídě `JpegCounter` jinou hodnotu, než verze uvedená zde v zadání?
3. Předpokládejte, že vytvoříme program, který vyrobí novou instanci třídy `JpegCounter`, která dostane odkaz na seznam 10.000.000 jmen souborů, kde rámcově desetina z nich má příponu `.jpg` nebo `.jpeg`. Dále předpokládejte nějaký typický dvoujádrový procesor, na kterém takový program spustíme. Proveďte hrubý odhad, kolikrát bude asi uvedená implementace metody `CountJpegs` rychlejší (nebo pomalejší) než její sekvenční implementace (kdybychom napsali běžný `for` cyklus a všechna volání metody `ProcessFilename` provedli postupně v jednotlivých iteracích `for` cyklu rovnou z volajícího vlákna). Vysvětlete, zda (a případně jak) by se dala metoda `ProcessFilename` co nejjednodušeji upravit tak, aby byla paralelní verze rychlejší než verze uvedená v zadání.

6 Sítě (3 body)

1. Stručně vysvětlete význam protokolu IP z pohledu vrstevnaté architektury. Jaké funkce nabízí vrstvám nad sebou a co požaduje od vrstev pod sebou?
2. Protokol IPv4 v sobě nativně podporuje fragmentaci datagramů. Vysvětlete, kdy k fragmentaci může dojít a jak je možné jí předcházet.
3. Uvažme použití protokolu IP nad Ethernetovou linkou (1000BASE-T). Vysvětlete, jak v tomto případě síťový uzel přeloží IPv4 a IPv6 adresy na adresy linkové vrstvy (MAC) pro lokální doručení v rámci dané sítě.

7 Dualita lineárního programování (3 body)

1. Formulujte přesné znění silné věty o dualitě lineárního programování.
2. Je dán neorientovaný graf $G = (V, E)$ a vrcholy $s_1, s_2, t_1, t_2 \in V$. Pro $i = 1, 2$ označme P_i množinu všech cest mezi vrcholy s_i a t_i v daném grafu a nechť $P = P_1 \cup P_2$. Uvažme následující lineární program, ve kterém máme proměnnou x_p pro každou cestu z množiny P:

$$\begin{aligned} &\max \sum_{p \in P} x_p \\ \text{t.ž.} \quad &\sum_{p : e \in p} x_p \leq 1 \quad \text{pro} \quad \forall e \in E \\ &x_p \geq 0 \quad \text{pro} \quad \forall p \in P \,. \end{aligned}$$

   Napište duální program (duální proměnné označujte y, s patřičnými indexy).
3. Uvažme graf G daný množinou vrcholů $V = \{s_1, s_2, t_1, t_2, a, b, c, d\}$ a množinou hran

$$E = \{(s_1, a), (s_2, a), (s_1, b), (s_2, b), (a, c), (b, d), (t_1, c), (t_2, c), (t_1, d), (t_2, d)\} \,.$$

   Pokud existuje, najděte optimální řešení výše uvedené primární úlohy a pomocí silné věty o dualitě dokažte, že je opravdu optimální.

8 Překladače (3 body)

Gramatika má počáteční neterminál S a terminály **id**, **","**, **";"**, **long**, **int**, **double** a #. Terminál **id** představuje libovolný identifikátor (různý od terminálů - klíčových slov) tvořený sekvencí písmen a číslic začínající písmenem:

$$\begin{aligned} S &\rightarrow D\ \# \\ D &\rightarrow T\ L\ \mathbf{;} \\ T &\rightarrow M\ B \\ M &\rightarrow \lambda \\ M &\rightarrow \mathbf{long} \\ B &\rightarrow \mathbf{int} \\ B &\rightarrow \mathbf{double} \\ L &\rightarrow \mathbf{id} \\ L &\rightarrow L\ \mathbf{,\ id} \end{aligned}$$

1. Nakreslete derivační strom pro řetězec **int i, j;** #.
2. Určete množiny $FIRST(A)$ a $FOLLOW(A)$ pro $A \in \{D, T, M, B, L\}$.
3. Určete, zda je gramatika LL(1), a stručně vysvětlete proč.

9 Aproximační algoritmy (otázka studijního zaměření – 3 body)

1. Zaveďte pojem "aproximační faktor algoritmu".
2. Popište co nejlepší aproximační algoritmus pro problém vrcholového pokrytí v grafu, jaký znáte.
3. Analyzujte výše popsaný algoritmus (tj. správnost, časová složitost, aproximační poměr).

10 Kombinatorická geometrie (otázka studijního zaměření – 3 body)

Zformulujte a dokažte Clarksonovu větu o horním odhadu na složitost k-té úrovně jednoduchého arrangementu n přímek v rovině.

(Nápověda k důkazu: Uvažujte náhodný podarrangement R, do nějž umístíme každou přímku původního arrangementu s pravděpodobností $p = \frac{1}{k+1}$ nezávisle na ostatních. Dvěma způsoby odhadněte střední hodnotu počtu vrcholů na hladině 0 arrangementu R, seshora s využitím velikosti R a zespoda odhadnutím pravděpodobnosti že vrchol na hladině nejvýše k v původním arrangementu se stane vrcholem na hladině 0 v arrangementu R.)

11 Pravděpodobnostní metoda (otázka studijního zaměření – 3 body)

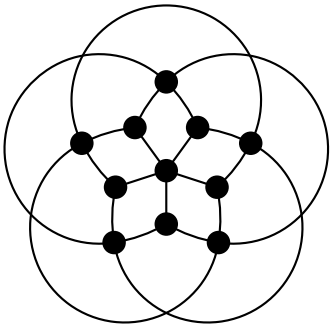
Nechť $G = (V, E)$ je graf s m hranami. Dokažte, že G obsahuje bipartitní podgraf (ne nutně indukovaný) s alespoň $m/2$ hranami.

12 Axiom výběru (otázka studijního zaměření – 3 body)

Zformulujte co nejpřesněji axiom výběru a podrobně dokažte, že je ekvivalentní následujícímu tvrzení: Kartézský součin neprázdného souboru neprázdných množin je vždy neprázdný.

13 Barevnost grafů (otázka studijního zaměření – 3 body)

1. Napište definici vrcholové a hranové barevnosti grafu. Určete vrcholovou a hranovou barevnost následujícího grafu:



2. Nechť G je graf s lichým počtem vrcholů takový, že všechny vrcholy G mají stejný stupeň d, $d \geq 1$. Určete hranovou barevnost G.

14 Pravděpodobnostní metoda (otázka studijního zaměření – 3 body)

Nechť $G = (V, E)$ je graf s m hranami. Dokažte, že G obsahuje bipartitní podgraf (ne nutně indukovaný) s alespoň $m/2$ hranami.

15 Matematická lingvistika: morfologická, syntaktická a sémantická analýza přirozeného jazyka (otázka studijního zaměření – 3 body)

1. Vysvětlete pojem "ontologie" ve zpracování sémantiky přirozeného jazyka.
2. Popište sémantickou síť Wordnet.
3. Vysvětlete pojem "anafora" a uveďte základní kategorie anafory v textu.

16 Matematická lingvistika: základy teorie informace (otázka studijního zaměření – 3 body)

Mějme dvě diskrétní náhodné veličiny X a Y. Obě nabývají čtyř různých hodnot z množiny $\{a, b, c, d\}$. Sdružené pravděpodobnostní rozdělení je následující:

	$X = a$	$X = b$	$X = c$	$X = d$
$Y = a$	1/8	1/16	1/16	1/4
$Y = b$	1/16	1/8	1/16	0
$Y = c$	1/32	1/32	1/16	0
$Y = d$	1/32	1/32	1/16	0

1. Rozhodněte, která z veličin X a Y má větší entropii. Odpověď zdůvodněte.
2. Vypočítejte podmíněnou entropii $H(Y|X = c)$.
3. Jaký vzorec použijete pro výpočet vzájemné informace $I(X; Y)$ na základě pravděpodobností uvedených v tabulce, tj. bez znalosti entropie?

17 Matematická lingvistika: formální jazyky a automaty (otázka studijního zaměření – 3 body)

Nejpopulárnějším nástrojem pro zpracování morfologie přirozených jazyků byla v 80. letech minulého století tzv. Two-level morphology profesorů Kartunnena a Koskenniemiho.

1. Pojmenujte dvě úrovně reprezentace zmíněné v názvu teorie.
2. Na jakém základním formálním nástroji bylo zpracování morfologie založeno?
3. Uveďte alespoň dvě ze tří základních myšlenek tohoto mechanismu.

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

1. Vyjmenujte a popište tři základní komponenty transformační gramatiky.
2. Vysvětlete pojem transformace v Chomského transformační gramatice. Kterými dvěma složkami je transformace definována?
3. Proč je nutné zavádět typy sestav rysů v unifikačních gramatikách?

19 Počítačová grafika: rekurzivní sledování paprsku (otázka studijního zaměření – 3 body)

1. Popište princip rekurzivního sledování paprsku (ray tracing). Co je na vstupu a co je výsledkem?
2. Z jakých složek se počítá světlo při dopadu paprsku v rekurzivní proceduře `shade()`?
3. Jaké hlavní nedostatky vykazuje základní algoritmus ray tracing popsaný v prvním bodě, pokud bychom ho chtěli použít jako fotorealistickou zobrazovací metodu?

20 Počítačová grafika: barvy (otázka studijního zaměření – 3 body)

1. Jak barvy vnímá lidské oko? Uveďte alespoň přibližný koncept.
2. Jak barvy zobrazuje počítačová technika? Popište systémy pro ukládání barev na počítači (barevné prostory).
3. Napište příklad barevného systému vhodného pro intuitivní zadávání barvy laickým uživatelem.

21 Počítačová grafika: real-time 3D grafika (otázka studijního zaměření – 3 body)

1. Stručně popište, jak vypadá pipeline pro vykreslení scény v existujících knihovnách pro 3D real-time grafiku s podporou HW akcelerace (např. OpenGL nebo DirectX). Zaměřte se pouze na hlavní komponenty této pipeline, které se přímo týkají renderingu scény na GPU, a stručně (nejlépe jednou větou) popište jejich funkci.
2. Která místa z výše uvedené pipeline může programátor ovlivnit (nahradit je vlastním kódem) a jak?
3. Kde je v této pipeline řešena otázka korektního zakrývání objektů (že obraz objektů blíže ke kameře může zakrýt objekty vzdálenější) a jak? Také v tomto kontextu vysvětlete, jak se zpracují poloprůhledné objekty.

22 Garbage collection (otázka studijního zaměření – 3 body)

1. Popište "mark and sweep" algoritmus pro garbage collection. V popisu ideálně použijte standardní barevné značení objektů (černá pro navštívené objekty, šedá pro zpracovávané objekty, bílá pro ostatní objekty).
2. Uvažujte následující program:

```
1  class Data {
2      public Data neighbor;
3  }
4
5  public class Main {
6
7      public static void one (Data d) {
8          Data x = new Data ();
9          // Bod kolekce 1
10         d.neighbor = x;
11         two (d);
12     }
13
14     public static void two (Data d) {
15         Data y = new Data ();
16         // Bod kolekce 2
17         y.neighbor = d;
18     }
19
20     public static void main (String [] arguments) {
21         Data d = new Data ();
22         // Bod kolekce 3
23         one (d);
24     }
25 }
```

Jaké objekty by "mark and sweep" garbage collector považoval za kořeny (roots), pokud by byla "stop the world" kolekce vykonána na řádcích označených v komentářích body 1, 2 nebo 3 (odpovídejte pro každou pozici jednotlivě) ? Vysvětlete proč.

23 Testování funkčnosti (otázka studijního zaměření – 3 body)

Zadání používá pro ilustraci jazyk Java a framework JUnit 4, nicméně otázky se týkají principů testování a v odpovědích můžete použít i jiná podobná prostředí.

Uvažujte následující fragment kódu:

```
1  import java.util.List;
2  import java.util.ArrayList;
3
4  import org.junit.*;
5  import static org.junit.Assert.*;
6
7  public class ListTest {
8      private List<String> list;
9      @Before public void setUp () { list = new ArrayList<String> (); }
10     @Test public void testEmpty () { assertTrue (list.isEmpty ()); }
11     @Test public void testAddSize () {
12         list.add ("X");
13         assertEquals (1, list.size ());
14     }
15     @Test public void testAddContents () {
16         list.add ("X");
```

```
17          assertTrue (list.contains ("X"));
18      }
19  }
```

1. Vysvětlete účel třídy `ListTest` a jejích jednotlivých atributů a metod.
2. Do třídy `ListTest` by šlo evidentně analogicky doplňovat další metody například takto:

```
1  public class ListTest {
2      ...
3      @Test public void testTwoAddSize () {
4          list.add ("X"); list.add ("X");
5          assertEquals (2, list.size ());
6      }
7      @Test public void testThreeAddSize () {
8          list.add ("X"); list.add ("X"); list.add ("X");
9          assertEquals (3, list.size ());
10     }
11     ...
12 }
```

   Za jakých okolností by takové doplnění dávalo smysl, pokud vůbec ? Obecněji, jakými ohledy se řídí volba metod, které implementuje třída `ListTest`, případně jiné podobné třídy stejného účelu ?

3. Co se nejspíš stane, pokud v metodě `testAddContents` vypustíme volání `list.add` ? Diskutujte v širším kontextu procesu vývoje software, nikoliv pouze chování volaných metod.

24 Paralelismus a synchronizace na multiprocesorech (otázka studijního zaměření – 3 body)

1. Předpokládejte následující kód zámku (spinlock):

```
1  #include <atomic>
2
3  void lock (std::atomic<bool> &lock) {
4      while (lock.exchange (true)) {
5      }
6  }
7
8  void unlock (std::atomic<bool> &lock) {
9      lock = false;
10 }
```

   Krok po kroku (řádek po řádku) vysvětlete, jak bude tento kód synchronizovat dvě vlákna, která se současně pokusí získat zámek. Pozornost věnujte zdůvodnění správnosti zámku.

2. Uvažujte situaci, kdy se dvě vlákna ucházejí o zámek z předchozího bodu, který je ale trvale zamčen třetím vláknem. Může se v takové situaci lišit provoz mezi procesorem a pamětí v případě, kdy tento kód poběží na multiprocesoru, od případu, kdy by běžel na uniprocesoru ? Proč ?
3. Bylo by možné v zámku z předchozího bodu nahradit typ `std::atomic<bool>` prostým typem `bool` ? Proč ne nebo jak ano ?

25 Databáze a Web: Validita XML dat (otázka studijního zaměření – 3 body)

1. Vysvětlete pojem "validní XML schéma".
2. Uveďte alespoň 3 rozdíly mezi jazykem DTD a XML Schema.
3. V jazyce XML Schema vyjádřete element adresa, který obsahuje podelementy ulice, číslo popisné, město a PSČ v libovolném pořadí. Lze takové schéma vyjádřit i v jazyce DTD? Pokud ano, jak. Pokud ne, proč.

26 Databáze a Web: XSLT (otázka studijního zaměření – 3 body)

1. Krátce popište jak pracuje XSLT procesor. Vysvětlete pojem "implicitní XSLT šablona".
2. Jaký bude výstup aplikace prázdného XSLT skriptu na dokument obsahující pouze element `<h1 c="blue">Hello world!</h1>`?
3. Napište XSLT skript, jehož výstupem je seznam názvů a hodnot všech atributů libovolného vstupního XML dokumentu.

27 Databáze a Web: Relační úplnost, spojení tabulek a SQL (otázka studijního zaměření – 3 body)

1. Na vhodném jednoduchém příkladu vysvětlete rozdíl mezi vnitřním a vnějším spojením tabulek v jazyce SQL.
2. Vysvětlete pojmy "relace" a "relačně úplný jazyk". Je jazyk SQL relačně úplný? Proč?

28 XML (otázka studijního zaměření – 3 body)

Uvažujte seznam institucí veřejné správy (např. Ministerstvo financí, Český statistický úřad nebo Město Turnov). Pro každý úřad evidujeme

- název (text, povinné)
- IČO (osmimístné číslo, zleva doplněno 0)
- adresu sídla sestávající z názvu ulice, čísla popisného a města (vše povinné)
- kontaktní email (povinné)
- seznam úřadoven (tj. míst, kam mohou občané chodit vyřizovat svojí agendu s úřadem osobně).

Seznam může obsahovat libovolný počet úřadoven a může být i prázdný. Pro každou úřadovnu jsou evidovány kontaktní hodiny (povinné).

Vytvořte XML schéma popisující XML dokumenty pro zápis seznamu institucí veřejné správy. K XML schématu napište také příklad validního XML dokumentu (s alespoň jednou institucí). Napiště XPath výraz, který z XML dokumentu vybere instituce s úřadovnou, která má otevřeno v úterý odpoledne (tj. od 12:00 či později).

29 Návrhové vzory (otázka studijního zaměření – 3 body)

Stejně jako v předchozí otázce uvažujte seznam institucí veřejné správy. Uvažujte dvě různé třídy (v libovolném objektovém či objektově orientovaném jazyku – C++, C#, Java, ...) reprezentující služby pro správu seznamu. První třída, `SluzbaA`, nabízí ve svém rozhraní pro vložení nové instituce metodu `vytvorInstituci`, druhá třída, `SluzbaB`, nabízí metodu `zalozInstituci`. Obě mají také různé sady parametrů pro specifikaci nové instituce.

Uvažujte třídu `Klient`, jejíž implementace umožňuje vytvořit novou instituci voláním metody `vytvorInstituci`. Přišel požadavek, aby byla využívána metoda `zalozInstituci` místo metody `vytvorInstituci`. Navrhněte řešení, které má co nejmenší vliv na vnitřní kód uvedených tříd. Dále rozšiřte třídu `SluzbaB` a případně další potřebné třídy tak, aby se k instanci třídy `SluzbaB` mohly registrovat různí klienti (instance třídy `Klient` i jiných tříd), kteří budou instancí třídy `SluzbaB` notifikováni v případě jakékoliv změny v seznamu institucí.

Pro řešení zvolte vhodné návrhové vzory. Zvolené návrhové vzory pojmenujte. Řešení buď zakreslete v podobě UML diagramu či diagramů nebo zapište v kódu ve zvoleném jazyce.

30 Testování funkčnosti (otázka studijního zaměření – 3 body)

Stručně popište běžně využívané postupy na testování komponent (tříd, modulů), které mají netriviální závislosti. Proveďte jejich vzájemné srovnání (diskuzi výhod a omezení) z pohledu kritérií jako například vyjadřovací síla nebo jednoduchost použití během vytváření testů.

31 Ortonormální báze (3 body)

1. Definujte pojem ortonormální báze.
2. Buď z_1, z_2, z_3, z_4 ortonormální báze prostoru $\mathbb{R}^4$ a nechť vektor $u = z_1 - z_2 + 2z_3 - 2z_4$.
   - Určete $\|u\|$,
   - rozhodněte, zda je vektor $v = z_1 - z_2 + 3z_3 + 4z_4$ kolmý na u,
   - sestrojte projekci vektoru u do ortogonálního doplňku množiny $\{z_1, z_2\}$.

32 Determinanty a podobnost (3 body)

1. Definujte determinant matice a rozhodněte, zda obecně platí $\det(A+B) = \det(A) + \det(B)$. (Dokažte nebo vyvraťte.)
2. Spočtěte determinant matice
$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$
3. Definujte podobnost matic a dokažte, že podobné matice mají stejný determinant.

33 Regulární matice (3 body)

1. Definujte pojem regulární matice a dokažte, že reálné regulární matice řádu n spolu s operací maticového součinu tvoří grupu.
2. Spočítejte kolik existuje regulárních matic v $\mathbb{Z}_2^{3\times 3}$.

34 Taylorův rozvoj (3 body)

Spočtěte Taylorův rozvoj funkce $\log(1+x)$ v bodě $x = 0$.

Vypočítejte limitu
$$\lim_{x\to+\infty} \left(x^2(\log(x+1) - \log x) - x\right).$$

35 Primitivní funkce (3 body)

Uveďte vzorec pro integraci (tj. nalezení primitivní funkce) metodou per partes.

Nalezněte jím
$$\int e^x \sin x\,.$$

36 Grafy (3 body)

1. Zadefinujte pojmy tah a uzavřený tah v grafu.
2. Zformulujte nutnou a postačující podmínku pro to, aby v grafu existoval uzavřený tah obsahující všechny jeho vrcholy a hrany.
3. Ukažte, že každý graf má orientaci, v níž se vstupní a výstupní stupeň každého vrcholu liší nejvýše o jedna.

37 Vytvořující funkce (3 body)

Nechť $a_0, a_1, a_2 \ldots$ je posloupnost čísel definovaná pomocí následující soustavy rekurencí:

$$\begin{aligned} a_0 &= 2 \\ a_1 &= -1 \\ a_n &= 3a_{n-1} - 2a_{n-2} \text{ pro } n \geq 2. \end{aligned}$$

Najděte vytvořující funkci této posloupnosti. Výsledek vyjádřete vzorečkem v uzavřeném tvaru, tj. bez použití nekonečných součtů. Není třeba hledat vzorec pro n-tý člen posloupnosti.

38 Normální podgrupy (3 body)

Definujte pojem normální podgrupy v grupě.

Pro pevné $n \in \mathbb{N}$ uvažte množinu všech permutací n-prvkové množiny, které mají kladné znaménko, tedy $A_n = \{\pi \in S_n : sgn(\pi) = 1\}$. Ukažte, že tvoří podgrupu symetrické gupy S_n všech permutací a rozhodněte, zda se jedná o normální podgrupu.

39 Logika (3 body)

Vyjádřete následující dvě tvrzení formulemi predikátové logiky nad vhodně zvoleným jazykem (s unárními predikáty pro "*chce*", "*hledá způsoby*", "*hledá důvody*").

1. *Kdo chce, hledá způsoby, kdo nechce, hledá důvody.*
2. *Kdo nehledá způsoby, hledá důvody.*

V nějakém formálním dokazovacím systému (tablo metoda, rezoluční metoda, Hilbertovský kalkul) dokažte, že z prvního tvrzení vyplývá to druhé.