Bakalářské zkoušky (příklady otázek z informatiky)

léto 2020

1 Automaty a gramatiky (3 body)

1. Definujte bezkontextovou gramatiku a Chomského normální formu bezkontextové gramatiky.
2. Mějme gramatiku $G = (\{E, B, I, J\}, \{0, 1, +, (,)\}, \mathcal{P}, E)$, kde $\mathcal{P}$ sestává z následujících pravidel (λ označuje prázdné slovo):

$$\begin{aligned}\mathcal{P} = \{E &\rightarrow E \mid B \mid I,\\ B &\rightarrow (E + E),\\ I &\rightarrow 0 \mid 1J,\\ J &\rightarrow 0J \mid 1J \mid \lambda\}\end{aligned}$$

   (a) Sestrojte levou derivaci slova $w = ((0 + 1) + 10)$ z gramatiky G. Zakreslete ji také ve formě derivačního stromu.

   (b) Najděte Chomského normální formu gramatiky G.

2 Algoritmy a datové struktury (3 body)

1. Napište pseudokód algoritmu prohledávání orientovaného grafu do hloubky (DFS).
2. Jaká je časová složitost algoritmu DFS, pokud je graf reprezentován maticí sousednosti?
3. Popište DFS klasifikaci hran orientovaných grafů.
4. Rozhodněte, zda v orientovaném grafu může existovat kružnice složená pouze ze zpětných hran.

3 Databáze (3 body)

1. Uvažujte transakční rozvrh $T_{123} = R_1(A), W_2(B), R_3(C), R_3(B), W_1(C), W_3(B), COMMIT_3, ABORT_2, COMMIT_1$. V zápisu rozvrhu $R_i(X)$, resp. $W_i(X)$ odpovídá čtení a zápisu proměnné X v i-té transakci.
   - Je rozvrh T_{123} konfliktově uspořádatelný (conflict-serializable)? Odpověď zdůvodněte.
   - Je rozvrh T_{123} zotavitelný (recoverable)? Odpověď zdůvodněte.

   Pokud rozvrh některou z vlastností nesplňuje, opravte jej tak, aby ji splňoval. Pokud to lze, dodržte vzájemné pořadí čtení a zápisů. Pokud to nelze, vysvětlete proč ne.
2. Nad tabulkami $Prodavac(\underline{ProdavacID}, Jmeno, Pobocka, DenNastupu)$ a $Prodeje(\underline{ProdavacID, DenProdeje}, Castka)$, najděte pomocí SQL všechny prodavače, kteří v období '1.5.2020' až '31.5.2020' nic neprodali. Pokud v daném dni prodavač nic neprodal, nemá pro daný den v tabulce *Prodeje* žádný řádek.

4 Programování (3 body)

Navrhněte objektovou datovou strukturu (implementovanou v jazyce C++, C# nebo Java), reprezentující matematický výraz obsahující konstanty (reprezentované jako double), proměnné (jejichž jména jsou řetězce písmen), operátory $+$, $-$, $*$, $/$ a rozšiřitelnou množinu matematických funkcí (jejichž jména jsou rovněž řetězce písmen). Uveďte veřejná i interní rozhraní umožňující následující funkčnost:

1. Tisk daného výrazu (v programátorské notaci; neřešte přitom přehlednost tištěného výrazu, jako např. nadbytečné závorky).
2. Konstrukce nového výrazu reprezentujícího symbolickou derivaci daného výrazu podle zadané proměnné. (Datová struktura reprezentující originální výraz musí při derivování zůstat nedotčena.) Dbejte na korektnost vytvořené struktury, např. v případě, kdy se při derivování duplikuje podvýraz (např. u derivace podílu), a uveďte v rozhraní i pomocné funkce, které jsou pro zajištění korektnosti nutné. Jako demonstraci použití těchto rozhraní napište implementaci funkce realizující derivaci matematické funkce *tg* podle vzorce

$$\frac{\mathrm{d}}{\mathrm{d}x} tg(f) = \frac{\frac{\mathrm{d}}{\mathrm{d}x} f}{sqr(cos(f))}$$

3. Mechanismus (bez větších nároků na rychlost, použitelný např. při načítání výrazu z textového vstupu), který pro zadaný řetězec obsahující jméno matematické funkce zkonstruuje objekt reprezentující volání této funkce s danými parametry. Tento mechanismus musí umožňovat snadnou rozšiřitelnost množiny matematických funkcí, s minimálním množstvím zásahů do společného kódu. Kterému návrhovému vzoru se řešení podobá?

Struktura musí umožňovat dostatečně rychlou manipulaci, není přípustné např. vyhledávání názvů funkcí v tabulkách během konstrukce derivace.

5 Architektura počítačů (3 body)

Mějme následující úryvek kódu ze třídy implementující spojový seznam:

```
1  struct SimpleListNode {
2      Data           payload;
3      SimpleListNode *next;
4  };
5
6  class SimpleList {
7  public:
8      // insert node n at the beginning of the list
9      void InsertNode(SimpleListNode *n) {
10         n->next = root;
11         root = n;
12     }
13
14     // insert two nodes at the beginning of the list
15     // the nodes must be neighbors, that is,
16     // first->next==second should be true
17     void InsertPair(SimpleListNode *first,
18                     SimpleListNode *second) {
19         InsertNode(second);
20         InsertNode(first);
21     }
22
23     ...
24
25 private:
26     SimpleListNode *root;
27 }
```

Tento úryvek kódu je v C++, pro ostatní jazyky (C# a Java) stačí smazat `*` a nahradit `->` znakem tečky.

1. Může při běhu funkcí v tomto úryvku kódu dojít k jevu zvaném "race condition"? Pokud ano, popište aspoň jednu takovou situaci. Uvažujte skutečně pouze tento úryvek kódu, nikoliv jeho interakci s dalším kódem třídy.
2. Pokud jsou v úryvku nějaké kritické sekce, napište rozsahy řádek, kde se nacházejí.
3. Využijte nějaký synchronizační nástroj a upravte zadaný úryvek kódu tak, aby fungoval korektně i v paralelním prostředí.

6 Transportní vrstva v TCP/IP (3 body)

1. Jaké úkoly plní transportní vrstva v architektuře TCP/IP?
2. Uvažme následující vlastnosti přenosů: blokový vs proudový, spojovaný vs nespojovaný, spolehlivý vs nespolehlivý, garantovaný vs negarantovaný, s nebo bez možnosti řešit řízení toku a konečně s nebo bez možnosti předcházet zahlcení sítě. Jaké vlastnosti ve srovnání s IP protokolem na síťové vrstvě v tomto smyslu nabízí protokoly TCP (*Transmission Control Protocol*) a UDP (*User Datagram Protocol*)?
3. Jakou funkci plní porty a jaké konvence používáme pro jejich přidělování – jak například konkrétně víme, že pro SMTP se používá port 25 nebo u PostgreSQL port 5432? Jsme v používání konkrétních portů nějak či někým omezováni? Jaké porty se používají pro označení odesílatele ve spojeních s odpovědí?
4. Jak vypadá relativní a absolutní transportní adresa a jak se identifikují (navzájem rozlišují) jednotlivá transportní spojení (komunikace)?

7 Aranžmá nadrovin (otázka studijního zaměření – 3 body)

Mějme aranžmá n různých nadrovin v $\mathbb{R}^d$ ne nutně jednoduché (tj. nadroviny nemusejí být v obecné poloze).

1. Určete maximální možný počet buněk takového aranžmá. Nápověda: Rozmyslete si, že takové maximum se nabývá pro jednoduché aranžmá a dokažte vzoreček pro jednoduché aranžmá. (Alternativně upravte některý z důkazů rovnou pro tuto situaci.) Pokud si budete vědět rady pouze s jedním z kroků z nápovědy, proveďte alespoň jeden z nich (za menší počet bodů).
2. Určete minimální možný počet buněk takového aranžmá. Nápověda: jak by to dopadlo pro $d = 1$.

8 Aproximační algoritmy (otázka studijního zaměření – 3 body)

1. Definujte aproximační poměr algoritmu.
2. Uvažme problém hranově disjunktních cest délky nejvýšše L: Vstupem je graf $G = (V, E)$, nezáporný celočíselný parametr $L \leq |V|$, a posloupnost dvojic vrcholů $(s_1, t_1), \ldots, (s_k, t_k)$. Úkolem je najít množinu $I \subseteq \{1, \ldots, k\}$ a cesty P_i, pro $i \in I$ takové, že P_i spojuje s_i a t_i, cesty jsou hranově disjunktní a každá má délku nejvýšše L; velikostí rešení je $|I|$ a cílem je najít co největší řešení.

   Navrhněte a analyzujte hladový algoritmus na řešení tohoto problému, jehož aproximační poměr pro $L < \sqrt{|E|} - 1$ je lepší než $\sqrt{|E|}$.

9 Perfektní kódy (otázka studijního zaměření – 3 body)

Definujte pojem perfektní (binární) kód.

Rozhodněte, zda existuje perfektní binární kód

1. délky 5 s minimální vzdáleností 3,
2. délky 7 s minimální vzdáleností 3.

Odpovědi přiměřeně zdůvodněte.

10 Relační databáze a JSON (otázka studijního zaměření – 3 body)

Mějme běžný relační databázový systém, ve kterém je databáze a z ní nás zajímá jedna tabulka:

```
users (id: int, name: varchar, settings: text)
```

Sloupec `id` je primární klíč, `name` celé jméno uživatele a `settings` obsahuje JSON se strukturou obsahující konfiguraci uživatele (této struktuře rozumí jen webová aplikace používající databázi). Uveďme příklad takové konfigurace:

```
{
  "window": {
    "x": 42,
    "y": 54,
```

```
  },
  "theme": "dark",
  "notifications": true,
}
```

Konfigurace je tedy serializovaná struktura s parametry, jejichž hodnoty jsou řetězce, celá čísla, booly, nebo vnořené struktury (rekurzivní vnoření není dovoleno, položky uvnitř struktury `window` musí být již pouze skaláry). Dále je garantováno, že všechny názvy položek obsahují pouze takové znaky, aby bylo možné je přímo zapsat v JavaScriptu jako identifikátory, délky názvů nepřesáhnou 64 znaků a délky řetězcových hodnot 255 znaků.

1. Kterou normální formu výše uvedené schéma porušuje a proč? Jaké praktické výhody a nevýhody má toto porušení normální formy? (Uveďte stručně jednu výhodu a jednu nevýhodu.)
2. Navrhněte úpravu schématu (případně dat celkově) tak, aby databáze danou normální formu neporušovala.
3. Napište SQL dotaz, pomocí kterého může aplikace pracující s databází získat kompletní nastavení uživatele ve vámi navrženém upraveném schématu. Technické detaily formátování případně spojování řetězců neřešte (beztak se často liší dle použitého dialektu SQL), ale zaměřte se na to, aby váš dotaz vrátil z databáze všechna potřebná data, ze kterých si může aplikace původní JSON sestavit.

11 Bezpečnost (otázka studijního zaměření – 3 body)

1. Stručně vysvětlete význam digitálního certifikátu (X.509) v souvislosti se zajištěním bezpečnosti webových aplikací. Uveďte nejdůležitější položky, které certifikát obsahuje.
2. Navrhněte způsob, jak ukládat hesla do databáze, abychom minimalizovali riziko bezpečnostního incidentu (tj. aby je nemohl přečíst např. ani administrátor). Stručně popište algoritmus ověření hesla uživatele při autentizaci.
3. Uveďte dva různé typy tzv. *injection* útoků. Ke každému typu uveďte příklad, kde popíšete zranitelnost, vektor útoku a alespoň jednu metodu prevence.

12 Zpracování dat z HTML formuláře (otázka studijního zaměření – 3 body)

Mějme HTML formulář v tradiční (CGI-like, tedy bez skriptů na straně klienta) webové aplikaci na přidávání položek do databáze.

```
<form action="index.php?action=addItem" method="POST">...
```

Skript na straně serveru zpracuje data z formuláře a pokud jsou korektní, přidá nový záznam do databáze. Jako odpověď pak vygeneruje HTML stránku, na které je tabulka se všemi záznamy v databázi.

1. Stručně vysvětlete, jak se data přenesou z klienta na server (protokol, kódování). Uveďte příklad API, které zpřístupňuje data z formuláře samotnému skriptu `index.php`.
2. Uživatel vyplnil formulář, odeslal jej na server a zobrazila se mu stránka s aktuální tabulkou všech záznamů. Co se stane, když uživatel klikne na tlačítko pro znovunačtení stránky (*Refresh*) v prohlížeči? Je takové chování žádoucí z pohledu uživatele? Pokud ne, navrhněte příslušnou úpravu aplikace.
3. Uživatelé požadují novou funkci – součástí vkládaných dat je nově také jeden soubor, který se uploaduje na server při odeslání formuláře. Stručně uveďte, jaké všechny úpravy bude ve formuláři (tj. v HTML) nutné provést, případně vysvětlete proč.

13 Teorie množin (otázka studijního zaměření – 3 body)

Úlohu řešte v teorii ZF, bez použití axiomu výběru.

1. Napište definici „množina x má menší mohutnost než množina y“.
2. Dokažte, že pro každou množinu x platí, že x má menší mohutnost než potenční množina x.

14 Hranová barevnost (otázka studijního zaměření – 3 body)

1. Napište definici hranové barevnosti grafu.

2. Jakých hodnot může nabývat hranová barevnost 3–regulárního grafu? Pro každou možnou hodnotu uveďte příklad grafu této hranové barevnosti a ukažte, že žádná jiná možnost není.
3. S pomocí Hallovy věty (nebo jakkoli jinak) ukažte, že každý 3-regulární bipartitní graf má hranovou barevnost 3.

15 Perfektní kódy (otázka studijního zaměření – 3 body)

Definujte pojem perfektní (binární) kód.

Rozhodněte, zda existuje perfektní binární kód

1. délky 5 s minimální vzdáleností 3,
2. délky 7 s minimální vzdáleností 3.

Odpovědi přiměřeně zdůvodněte.

16 Morfologická, syntaktická a sémantická analýza přirozeného jazyka (otázka studijního zaměření – 3 body)

Sémantika se v přirozených jazycích uplatňuje na mnoha úrovních, od významu jednotlivých slov až po význam delších textových úseků. Vysvětlete některé základní sémantické pojmy:

1. Vysvětlete pojem „ontologie“ ve zpracování sémantiky přirozeného jazyka.
2. Popište sémantickou síť Wordnet, její strukturu a historii, a uveďte alespoň dva příklady aplikace této sítě v různých oblastech zpracování přirozeného jazyka.
3. Vysvětlete pojem „anafora“ a uveďte základní kategorie anafory v textu.

17 Jazykové modelování (otázka studijního zaměření – 3 body)

1. Popište základní fakta o tvorbě jazykových korpusů.
2. Rozdělte jazykové korpusy podle typu značkování.
3. Pro každý typ značkování uveďte alespoň jeden příklad korpusu češtiny a jeden příklad nějakého jiného jazyka.

18 Základy teorie informace (otázka studijního zaměření – 3 body)

Házíme hrací kostkou a hozené číslo z množiny $\{1, 2, 3, 4, 5, 6\}$ interpretujeme jako hodnotu náhodné proměnné X. Předpokládejme, že X má uniformní rozdělení. Dále uvažujme náhodnou proměnnou Y s hodnotami *sudé/liché* a náhodnou proměnnou Z s hodnotami *true* (pokud padne číslo větší než 4) nebo *false* (pokud nepadne číslo větší než 4). Obory hodnot náhodných proměnných jsou shrnuty v tabulce

náhodná proměnná	**hodnoty**
X	$\{1, 2, 3, 4, 5, 6\}$
Y	$\{sudé, liché\}$
Z	$\{true, false\}$

1. Která z proměnných X, Y, Z má největší entropii? Odpověď přesně zdůvodněte.
2. Určete vzájemnou informaci $\mathrm{I}(X; Y)$. Výsledek zdůvodněte.

19 Rastrová a vektorová grafika (otázka studijního 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 obrázků (nemusíte uvádět detaily).

20 Anti-aliasing (otázka studijního 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 (rasterizace, např. při kreslení vektorové grafiky)?
3. Jak se anti-aliasing implementuje v prostředí paprskového zobrazovače (např. ray-tracing)?

21 Textury ve 3D grafice (otázka studijního zaměření – 3 body)

1. Jaký je rozdíl mezi 2D a 3D texturou? Uveďte typické aplikace těchto dvou přístupů.
2. Jaké jsou hlavní výhody a nevýhody 3D textur při použití v ray-tracingu?
3. Jak je možné napodobit vnitřní strukturu materiálu (dřevo, mramor) pomocí textury nebo procedurální definice v paprskovém zobrazovači (ray-tracing, path-tracing)? Pozn.: pište o simulaci struktury materiálu, ne o zobrazovači!

22 Techniky přenosu na linkové vrstvě (otázka studijního zaměření – 3 body)

Navrhněte hypotetický bytově orientovaný blokový protokol pro linkovou vrstvu, který bude nespojovaný, negarantovaný (tedy Best Effort) a spolehlivý, a to v sítích s přepojováním paketů. Popište strukturu dat a význam jednotlivých položek. Stačí se věnovat jen těm, které jsou nezbytné pro zajištění funkce doručování užitečného nákladu a případných chybových zpráv. Ošetřete transparenci dat. Popište konkrétní způsob zajištění spolehlivosti. Vše pečlivě vysvětlete a odůvodněte. Můžete aplikovat obecné principy stejně jako se inspirovat reálně existujícími protokoly.

23 Propojování na linkové vrstvě (otázka studijního zaměření – 3 body)

Mějme lokální síť používající technologii Ethernet s přístupovou metodou CSMA/CD obsahující jeden **přepínač** (*switch*) propojující následující segmenty:

- segment I se dvěma koncovými uzly majícími hardwarové adresy A a B,
- segment II s jediným uzlem C,
- segment III se dvěma uzly D a E,
- segment IV s uzlem F,
- a nakonec segment V také jen s jediným uzlem G.

Vysvětlete následující aspekty propojování na linkové vrstvě L2:

1. Co to je, jak funguje a k čemu slouží **filtrování** (*filtering*) a **cílené předávání** (*forwarding*)? Ilustrujte na konkrétních příkladech.
2. Jak v tomto kontextu můžeme využít **metodu zpětného učení**? Jak funguje?
3. Je možné filtrování a/nebo cílené předávání realizovat např. v rámci opakovače (*repeater*) na fyzické vrstvě L1? Vysvětlete.
4. Co to je **kolize**? Šíří se kolize skrz přepínač do dalších segmentů?
5. Pokud v segmentu I aktuálně probíhá kolize, může uzel C poslat nějaká data uzlu F, uzel D data pro uzel E a uzel G data pro uzel A? Vysvětlete.
6. Co to je **mikrosegmentace**? Jak by se naše síť musela změnit, abychom jí dosáhli? Jaké přináší výhody?

24 Doručování IP datagramů (otázka studijního zaměření – 3 body)

V rámci domácí sítě mějme jediný směrovač (*router*) s překladem mezi privátními a veřejnými adresami, jeho vnější rozhraní má veřejnou IP adresu 94.112.15.207 s CIDR prefixem 24 a vnitřní rozhraní privátní adresu 192.168.0.1 s maskou sítě 255.255.255.0. Předpokládejte, že překladová tabulka aktuálně obsahuje následující záznamy:

Vnitřní adresa	Protokol	Vnější adresa
192.168.0.11:55001	TCP	94.112.15.207:65401
192.168.0.22:50999	UDP	94.112.15.207:65402

Jak bude probíhat doručení IP datagramů od odesilatele k příjemci v následujících situacích? Popisujte pouze provoz na odesilateli, příjemci a směrovači (tedy na uzlech, o kterých jsou v zadání k dispozici konkrétní informace jako IP adresy), případné další uzly zmiňte pouze okrajově. Vše vysvětlete z pohledu procesu doručování IP datagramů, tedy pro každý IP datagram a každý uzel sítě napište, na základě čeho a jak uzel s IP datagramem naloží, případně jak se změní obsah IP datagramu, případně stav uzlu.

1. Uzel s adresou 192.168.0.11 odešle uzlu s adresou 195.113.27.221 IP datagram, který obsahuje data TCP spojení z portu 55001 na port 80.
2. Uzel s adresou 192.168.0.22 přijme od uzlu s adresou 195.113.19.71 IP datagram, který je odpovědí UDP protokolu na DNS dotaz z portu 50999 na port 53.
3. Uzel s adresou 192.168.0.22 odešle uzlu s adresou 192.168.0.11 IP datagram, který obsahuje data UDP protokolu z portu 50999 na port 5432.

Kdy se jednotlivé položky z překladové tabulky odstraní?

25 XML (otázka studijního zaměření – 3 body)

Uvažujte program multikina. Program je vydáván vždy v sobotu a obsahuje plán promítání na pondělí až neděli další týden. Multikino má jeden nebo více sálů. Program je strukturován po promítacích dnech a pro daný den uvádí pro každý sál seznam promítání. Pro jedno promítání je uveden identifikátor a název promítaného filmu a čas začátku a konce promítání. Vedle samotného plánu promítání obsahuje program také detailní informace o filmech - kromě identifikátoru a názvu filmu také jeho popis, informaci o věkovém omezení a obrázek. Další informace o filmu neuvažujeme.

Vytvořte XML schéma popisující reprezentaci programu multikina dle výše uvedeného zadání v XML. K XML schématu napište také příklad validního XML dokumentu (s alespoň jedním promítacím sálem). Napište XSLT transformaci této XML struktury do HTML dokumentu se seznamem filmů, které jsou promítány v pondělí. Chceme zobrazit název filmu a věkové omezení. Ukažte HTML kód, který je výsledkem XSLT skriptu aplikovaného na váš příklad XML dokumentu.

26 JavaScript, CSS a HTML (otázka studijního zaměření – 3 body)

Pomocí vhodné kombinace CSS a JavaScriptu rozšiřte váš HTML kód z předchozí otázky tak, aby uživatel mohl jednotlivé filmy skrývat a zobrazovat. Pro každý zobrazený film bude zobrazeno tlačítko, na které když uživatel klikne, film bude skryt. Dále bude zobrazeno jedno tlačítko, které zobrazí všechny skryté filmy (a zobrazené filmy ponechá zobrazené). Jako odpověď na otázku je očekáváno rozšíření HTML kódu z předchozí otázky, potřebný kód v jazyku JavaScript a CSS styly.

27 Základy technologií sémantického webu (otázka studijního zaměření – 3 body)

Uvažujte program multikina z předchozích otázek. Ukažte na něm, jak by vypadala reprezentace multikina v datovém modelu RDF tak, aby odpovídala čtyřem principům Linked Data.

28 Synchronizace na multiprocesorech (otázka studijního zaměření – 3 body)

1. Definujte spinlock, tedy napište jeho stav a metody včetně jejich významu.
2. Implementujte spinlock s pomocí následujících funkcí:

```
// Read a value of an integer variable atomically.
int atomic_read (int *address);

// Write a value of an integer variable atomically.
void atomic_write (int *address, int value);

// Atomically, if the address contains value old,
// set the address to value new and return TRUE,
```

```
// otherwise return FALSE.
bool atomic_compare_and_swap (int *address, int old, int new);
```

3. Implementujte spinlock s pomocí následujících funkcí:

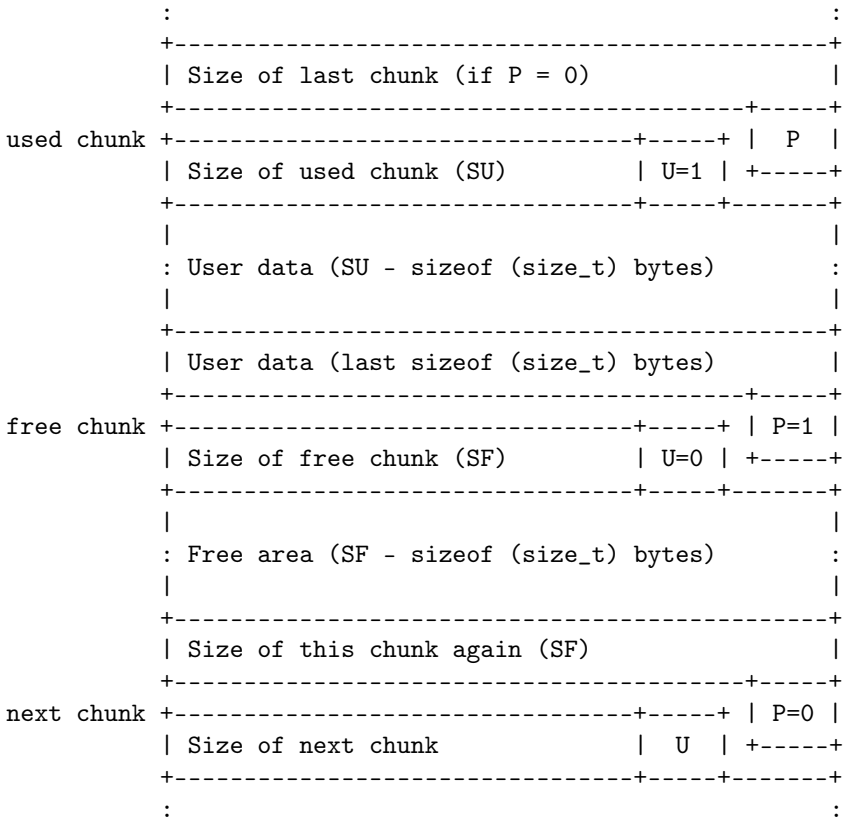
```
// Read a value of an integer variable atomically.
int atomic_read (int *address);

// Write a value of an integer variable atomically.
void atomic_write (int *address, int value);

// Atomically, read a value of an integer variable
// and start watching for further writes to address.
int load_link (int *address);

// Atomically, if the address is watched from past
// call to load_link, and if there was no write to
// the address since that call, write the value
// to the address and return TRUE, otherwise
// return FALSE.
bool store_conditional (int *address, int value);
```

4. Předpokládejte, že hardware může mezi `load_link` a `store_conditional` sledovat pouze omezený počet adres, například pouze jednu. Vysvětlete, zda a jak takové omezení může ovlivnit chování implementace spinlocku z předchozího bodu.

Vaše implementace musí fungovat na multiprocesorech.

29 Správa paměti (otázka studijního zaměření – 3 body)

Předpokládejte, že váš program je vybavený jednoduchým heap alokátorem, který používá následující struktury hlaviček (tyto shodou okolností pocházejí z alokátoru `dlmalloc` a ilustrují jeden obsazený a jeden volný blok):



Bit `P` indikuje, zda je použitý předchozí blok, bit `U` indikuje, zda je použitý aktuální blok. Pole SU a SF se týkají čisté velikosti dat (SU) nebo prostoru pro ně (SF).

Předpokládejte dále, že `sizeof (size_t) == 4`.

1. Měl by uvedený alokátor zarovnávat velikost alokovaných bloků ? Pokud ano, proč ?
2. Jaká by byla prostorová režie uvedeného alokátoru v programu, který alokuje výhradně bloky délky 40 bajtů ?
3. Jakou hodnotu by vrátilo volání `malloc (40)` v následujícím heapu, pokud by alokátor používal strategii first fit ? Dump obsahuje celý heap, hodnoty jsou little endian, velikosti jsou uváděné striktně v bajtech (toto by u skutečné implementace pravděpodobně neplatilo).

```
10000000   00 00 00 00 32 00 00 00 55 AA 55 AA 55 AA 55 AA
10000010   55 AA 55 AA 31 00 00 00 AA 55 AA 55 AA 55 AA 55
10000020   0C 00 00 00 22 00 00 00 55 AA 55 AA 55 AA 55 AA
10000030   01 01 00 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55
10000040   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10000050   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10000060   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10000070   40 00 00 00 00 00 00 00
```

30 Návrhové vzory (otázka studijního zaměření – 3 body)

Následující kód obsahuje zjednodušené rozhraní třídy `ClassVisitor` frameworku ASM pro analýzu bytekódu jazyka Java:

```
public abstract class ClassVisitor {
    public ClassVisitor () { ... }
    public void visit (String name, String parentName) { ... }
    public FieldVisitor visitField (String name, String type) { ... }
    public MethodVisitor visitMethod (String name, String returnType) { ... }
    public void visitEnd () { ... }
}
```

K dispozici je také zjednodušená třída `ClassReader`, jejíž konstruktor akceptuje jméno souboru s bytekódem a jejíž metoda `accept` navštíví obsažené třídy:

```
public class ClassReader {
    public ClassReader (String classFileName) { ... }
    public void accept (ClassVisitor visitor) { ... }
}
```

Metody `visitField` a `visitMethod` jsou volány mezi voláním `visit` a `visitEnd` v blíže neurčeném pořadí. Metody mohou vracet `null` v případě, že není třeba ve větším detailu navštívit dané pole nebo metodu.

1. Napište kód, který pomocí uvedených tříd vytiskne seznam polí a metod obsažených v daném souboru s bytekódem. Vysvětlete, jak spolu jednotlivé části kódu ve vašem řešení interagují. Formát výpisu volte co nejjednodušší, pořadí informací ve výpisu není důležité.
2. Navrhněte rozhraní třídy `MethodVisitor`, která by v uvedeném příkladu stejným návrhovým vzorem dovolila tisknout ještě jména a typy argumentů jednotlivých metod.