Bakalářské zkoušky (příklady otázek z informatiky)

léto 2021

1 Chomského hierarchie (3 body)

Uvažujme jazyk $L = \{w \mid w \in \{a,b\}^*, -1 \leq |w_a| - |w_b| \leq 1\}$ slov nad abecedou $\Sigma = \{a,b\}$, kde se počet a a b liší maximálně o 1. Zařaďte jazyk L do Chomského hierarchie a najděte automat odpovídající dané třídě (konečný, zásobníkový, lineárně omezený, Turingův stroj) přijímající jazyk L.

Popište a pojmenujte všechny složky definice automatu. Formální důkaz neexistence automatu užší třídy vyžadován není.

2 Haldy (3 body)

1. Popište implementaci datové struktury halda (obvyklá minimová, binární).
2. Napište pseudokód operace *ExtractMin* na popsané implementaci haldy.
3. Napište efektivní pseudokód pro Heapsort, tj. třídění haldou. Napište a rozeberte jeho asymptotickou složitost.

3 Relační databáze (3 body)

Uvažujte následující relační databázové schéma:

Product(ProductID, ProductName, PriceCategory, Price)
Discount(DiscountID, DiscountName, NewPrice)
DiscountProduct(DiscountID,ProductID)
DiscountProduct.DiscountID $\subseteq$ *Discount*.DiscountID
DiscountProduct.ProductID $\subseteq$ *Product*.ProductID

Podtržení označuje klíče relací. Souvislé podtržení více atributů za sebou znamená složený klíč přes více atributů.

1. Nakreslete odpovídající konceptuální model (UML nebo ER model).
2. Pomocí SQL dotazů najděte
   (a) všechny produkty, na které se žádná sleva nevztahuje,
   (b) všechny slevy, které nějaký produkt zdražují oproti jeho základní ceně.
3. Rozhodněte, zda by bylo vhodné upravit výše uvedené relační databázové schéma, pokud víte, že sloupec *Product*.Price funkčně závisí na sloupci *Product*.PriceCategory. Pokud ano, vysvětlete proč a uveďte jak. Pokud ne, vysvětlete proč.

4 Pořadí v týmové soutěži (3 body)

Představte si, že navrhujete systém pro organizaci sportovních soutěží, kde se utkávají týmy systémem "každý s každým", tedy postupně, v libovolném pořadí, se odehrají zápasy mezi každou možnou dvojicí týmů. V tomto zadání se soustředíme na tu část systému, která má za úkol vyhodnotit aktuální pořadí týmů – konkrétně v libovolné fázi soutěže z informací o odehraných zápasech spočítat pro každý tým skóre, podle kterého půjde všechny týmy seřadit.

Podle druhu soutěže může mít skóre více složek – například ve fotbale může být první složkou skóre pro jeden zápas rozdíl gólů a druhou složkou počet udělených žlutých karet (zjednodušeno, reálné hodnocení je samozřejmě složitější), každá složka se počítá podle konkrétního kritéria. Celkově se tedy (1) pro každý tým a zápas a kritérium vypočítá konkrétní složka skóre, (2) složky téhož týmu a kritéria se sečtou, (3) týmy se seřadí lexikograficky podle vektoru složek skóre.

Vyberte si jeden z jazyků C++, C# nebo Java a navrhněte v něm níže uvedené části systému pro určování pořadí:

- Datová struktura reprezentující množinu odehraných zápasů – zohledněte přitom skutečnost, že informace vyžadované pro určení pořadí závisejí na konkrétním sportu (např. skóre jednotlivých setů ve volejbale, výsledky z prodloužení v hokeji nebo počty žlutých karet ve fotbale). Snažte se udělat systém rozšiřitelný (rozšířením zdrojového kódu) na libovolný sport založený na zápasech dvou týmů, bez nutnosti zásahu do zdrojového kódu univerzální částí struktury.

  Při návrhu této struktury zvažujte pouze její použití pro určování pořadí, neřešte persistentní uložení ani načítání dat.

- Datová struktura definující pravidla dané soutěže pro určení pořadí týmů. Pravidla jsou dána posloupností kritérií, která počítají jednotlivé složky skóre - prvním kritériem bývá počet bodů ze zápasů (i ten se však v různých soutěžích určuje různě), další kritéria (aplikovaná v případě, že předchozí pravidla nerozhodla) mohou být založena na dalších informacích uložených u zápasů (např. počtu vstřelených branek, výsledků jednotlivých setů, žlutých karet). Všechna kritéria však budou založena na sečtení celých čísel odvozených z jednotlivých zápasů a porovnání těchto součtů mezi týmy. Uveďte objektové rozhraní reprezentující takové kritérium a třídu reprezentující posloupnost takových kritérií.

  Množina kritérií musí být snadno (rozšířením zdrojového kódu) rozšiřitelná, přičemž implementace kritérií by měly být použitelné pro různé soutěže a nejlépe i pro různé sporty všude, kde to přichází v úvahu (např. kritérium rozdílu skóre pro každý sport založený na počítání branek). Činnosti společné všem kritériím (např. sčítání přes množinu zápasů) nemají být součástí implementace kritéria.

- Nad výše uvedenými rozhraními ukažte detailní implementaci (univerzálního) kritéria rozdílu skóre a (fotbalového) kritéria menšího počtu žlutých karet, včetně souvisejících částí záznamu o zápase pro tyto sporty.

5 Architektura počítačů a operačních systémů (3 body)

Mějme následující třídu v pseudokódu pro C# a Javu, v komentářích je varianta pro C++:

```
1  class Node {
2  public:
3          Node    next; // Node *next; for C++
4          // some other data payload
5  };
6
7  class CountedList {
8  public:
9          CountedList() {
10                 root = null; // nullptr for C++
11                 count = 0;
12         }
13
14         int getCount() {
15                 return count;
16         }
17
18         void insertNode(Node n) { // (Node *n) for C++
19                 n.next = root; // n->next for C++
20                 root = n;
21                 ++count;
22         }
23
24 private:
25         Node    root; // Node *root; for C++
26         int     count;
27 };
```

Dále předpokládejme, že máme multiprocesorový systém a program si spustí několik vláken, která budou mít všechna přístup ke sdílené proměnné typu `CountedList`. Tato vlákna budou volat podle své potřeby funkce `getCount` a `insertNode`.

1. Je možné, že v programu nastane jev zvaný *race condition*? Pokud ano, napište rozsahy řádek nebo vyznačte přímo v uvedeném kódu řádky, které představují kritickou sekci.

2. Je možné, aby nějakým způsobem nastala situace, kdy při volání funkce `insertNode` z různých vláken budou prvky spojového seznamu správně pospojovány, ale počet prvků v proměnné `count` bude špatně spočítán? Zdůvodněte.

3. Pokud se domníváte, že v uvedeném kódu může nastat race condition, pokuste se tento problém odstranit úpravou programu (včetně přidání nějakých nových řádek).

6 Směrování (3 body)

Mějme směrovač (router) realizující standardní směrování na základě následující směrovací tabulky:

Destination	Netmask	Interface	Gateway	Metric
195.113.19.0	255.255.255.0	(A) 195.113.19.20	on-link	1
172.217.23.0	255.255.255.0	(B) 172.217.23.55	on-link	1
216.58.0.0	255.255.0.0	(C) 216.58.201.78	on-link	1
185.17.100.0	255.255.255.0	(B) 172.217.23.55	(R) 172.217.23.111	10
185.17.200.0	255.255.255.0	(B) 172.217.23.55	(S) 172.217.23.222	10
104.0.0.0	255.0.0.0	(C) 216.58.201.78	(T) 216.58.90.100	10

Pro zjednodušení jsou jednotlivá síťová rozhraní a adresy dalších směrovačů označené písmeny. Odpovídající MAC (EUI-48) adresy směrovačů jsou následující: (R) FC-77-74-19-41-1E, (S) C8-F7-50-22-91-BA a (T) 00-15-5D-C7-B9-EA.

Nejprve vysvětlete význam jednotlivých položek (sloupců) ve směrovací tabulce.

Na síťovém rozhraní (A) 195.113.19.20 postupně přijmeme následující standardní IPv4 datagramy:

1. Datagram pro příjemce 185.17.200.50
2. Datagram pro příjemce 195.113.19.20
3. Datagram pro příjemce 74.6.231.21
4. Datagram pro příjemce 216.89.23.11
5. Datagram pro příjemce 255.255.255.255
6. Datagram pro příjemce 216.58.255.255

Pro každou z předchozích situací určete, jakým způsobem bude daný datagram zpracován. Vysvětlete a odůvodněte. Pokud bude datagram posílán dále, jaká IP adresa příjemce bude vyplněna v příslušné položce (Destination Address) hlavičky IP datagramu a analogicky jaká MAC (EUI-48) adresa příjemce bude vyplněna v příslušném L2 rámci, do kterého se datagram vloží?

7 Optimalizační metody (3 body)

1. Formulujte Minkowského-Weylovu větu pro polytopy a ilustrujte příkladem na polytopu s alespoň pěti vrcholy.
2. Je dán bipartitní graf G=(U,V,E). Uveďte lineární program popisující konvexní obal perfektních párování grafu G.
3. Dokažte, že lineární program z předchozího bodu opravdu popisuje požadovanou množinu.

8 Splnitelnost (otázka studijního zaměření – 3 body)

1. Na problému splnitelnosti (MAX SAT) vysvětlete (ideálně pomocí tzv. Biased SAT algoritmu) využití pravděpodobnosti při hledání dokazatelně dobrých přibližných řešení NP-težkých problémů. Dokažte, jaký je aproximační poměr vámi popsaného algoritmu.
2. Problém rozhodnout, zda instance 2-SAT je splnitelná, je řešitelný v polynomiálním čase. Uvažte standardní lineární relaxaci pro MAX SAT – pro instanci SAT s proměnnými $x_1, \ldots, x_n$ a klauzulemi $C_1, \ldots, C_m$ o vahách $w_1, \ldots, w_m$, vypadá stadardní relaxace takto:

$$\max \sum_{j=1}^{m} w_j z_j$$

$$\sum_{i:x_i \in C_j} y_i + \sum_{i:\bar{x}_i \in C_j} (1 - y_j) \geq z_j \quad \text{pro} \quad \forall j = 1, \ldots, m$$

$$0 \leq y_i \leq 1 \quad \text{pro} \quad \forall i = 1, \ldots, n$$

$$0 \leq z_j \leq 1 \quad \text{pro} \quad \forall j = 1, \ldots, m$$

Rozhodněte, zda pro každou instanci 2-SAT najde tato relaxace optimální řešení, a svou odpověď dokažte.

9 Kombinatorická geometrie (otázka studijního zaměření – 3 body)

Dokažte, že neexistuje 6 kompaktních konvexních množin $A_0, A_1, B_0, B_1, C_0, C_1$ v rovině, které splňují následující podmínky:

- $A_0 \cap A_1 = \emptyset$, $B_0 \cap B_1 = \emptyset$, $C_0 \cap C_1 = \emptyset$
- Pro každé $i, j, k \in \{0, 1\}$ je průnik $A_i \cap B_j \cap C_k$ neprázdný.

Nápověda: Použijte větu o oddělování pro dvojice množin X_0, X_1, kde X je A, B nebo C. Pokud úlohu neumíte vyřešit, tak alespoň formulujte a dokažte větu o oddělování (konvexních kompaktních množin nadrovinou v obecné dimenzi).

10 Kódy (otázka studijního zaměření – 3 body)

1. Definujte blokový kód a jeho parametry a napište definici lineárního kódu.
2. Pro přirozené $n \geq 2$ uvažme binární kód obsahující všechna slova délky n sudé váhy, neboli $C = \{x = (x_1, \ldots, x_n) \in \mathbb{Z}_2^n : \sum_{i=1}^n x_i \equiv 0 \pmod 2\}$. Určete parametry tohoto kódu a ověřte, že je lineární.

11 Relační databáze (otázka studijního zaměření – 3 body)

Mějme cestovní kancelář, která uchovává svou nabídku zájezdů v relační databázi v tabulce *tours*. Tabulka *tours* má následující strukturu:

- **tour_id** (int): unikátní identifikátor zájezdu
- **title** (varchar): název zájezdu
- **country_code** (char[3]): identifikátor cílové země (dle ISO 3166-1)
- **description** (varchar): textový popis zájezdu
- **date_from** (date): termín odjezdu
- **date_to** (date): termín příjezdu
- **price** (int): cena zájezdu
- **price_description** (varchar): textový popis co cena zájezdu (ne)zahrnuje

1. V průběhu práce s databází se zjistilo, že tabulka *tours* neodpovídá dostatečně realitě: Stejné zájezdy (stejný název, popis a cílové místo) jsou obvykle nabízeny ve více termínech. Zároveň každý termín zájezdu může mít přiřazeno několik různých cen (např. 1-lůžkový / 2-lůžkový pokoj). Upravte schéma tak, aby odpovídalo popsaným skutečnostem.
2. Napište dotaz, který bude mít na výstupu stejnou strukturu jako původní tabulka *tours* (stejný seznam sloupců, tj. *tour_id*, *title*,...,*price_description*) a vrátí pro každý zájezd a každý jeho termín nejnižší cenu zájezdu (*price*) a jí odpovídající popisek ceny (*price_description*).
3. Vlivem chyby v navázaném informačním systému se do databáze zanesly nekonzistence, které je třeba odstranit. Napište dotaz, který vrátí všechny *id* zájezdu (*tour_id* ve vámi definovaném schématu), pro které platí, že buď nemají přiřazený žádný termín, nebo pro žádný z přiřazených termínů neexistuje záznam s odpovídající cenou.

Při řešení tolerujeme použití SQL dialektů běžně známých databází (Oracle, MySQL, PostgreSQL, MSSQL, ...).

12 PHP (otázka studijního zaměření – 3 body)

Uvažujme cestovní kancelář jako v otázce 7 o relačních databázích. Máme následující fragment PHP kódu, který zpracovává formulář odeslaný uživatelem (jednoduché vyhledávání zájezdů na základě podřetězce). Pro zjednodušení příkladu počítejte s tím, že:

- textový dotaz uživatele je odeslán ve formulářovém poli se jménem "searchtext" pomocí metody POST,
- připojení do databáze (*mysqli* objekt) lze získat voláním *Database::getInstance()*,
- databázová tabulka *tours* má stejnou strukturu jako v zadání otázky 7 o relačních databázích. Kromě toho v databázi ještě existuje tabulka *orders* obsahující objednávky zájezdů od klientů CK.

```
...
function findTours($queryText) {
  $mysqli = Database::getInstance();
  $sql = "select␣*␣from␣tours␣where␣title␣like␣'%$queryText%'";
  if ($result = $mysqli->query($sql)){
    return $result -> fetch_all();
  }
  return null;
}

function displayOneTour($row) {...}

function displayResults($queryText, $results) {
  echo "<h1>Results␣for:␣$queryText</h1>";
  foreach ($results as $row){
    displayOneTour($row);
  }
}

$res = findTours($_POST["searchtext"]);
displayResults($_POST["searchtext"], $res);
```

1. Zobrazený PHP fragment obsahuje několik bezpečnostních rizik. Stručně je popište a kód upravte tak, aby byla tato rizika eliminována. Funkci *displayOneTour()* (bez zobrazené implementace) můžete považovat za bezpečnou. [2 body]
2. Na příkladech vysvětlete, jak konkrétně by mohly být nalezené bezpečnostní hrozby zneužity, pokud by nebyly opraveny. Uveďte příklady uživatelských akcí nebo uživatelských vstupů, které by mohly narušit bezpečnost. [1 bod]

Drobné detaily syntaxe PHP nebudou součástí hodnocení. Použijete-li knihovní funkci, u které si nejste jisti názvem, vysvětlete do komentáře její sémantiku.

13 CSS (otázka studijního zaměření – 3 body)

Uvažujte následující fragment HTML kódu:

```
<div id="container">
  <div class="grid" id="grid1">GRID 1</div>
  <div class="grid" id="grid2">GRID 2</div>
  <div class="grid" id="grid3"><h1>GRID 3 <span>Text after</span></h1></div>
  <div class="grid" id="grid4"><a href="#">GRID 4</a></div>
  <div class="grid" id="grid5"><a href="https://cuni.cz">GRID 5</a></div>
  <div class="grid" id="grid6" title="Last">GRID 6</div>
</div>
```

K HTML stránce obsahující zmíněný fragment jsou přilinkované následující CSS definice:

```
div{
  border: 1px solid black;
  font-size:1.5em;
}
#container{
  display:grid;
  grid-template-columns: [left] 200px [center] auto [right] 20vw [end];
  grid-template-rows: [header] 1fr [main] 3fr [footer] 1fr [last];
}
#grid3{
  background-color:grey;
  grid-column: 2 / end;
  grid-row:2 / span 2;
}
#grid2{
```

```
  grid-row-start: 3;
  grid-column-start: 3;
  background-color:green;
}
#grid6{
  grid-column-start: 1;
  grid-column-end: right;
  grid-row-start:3;
  background-color:yellow;
}

div div.grid{color:red;}
div a[href^="http"]{color:violet}
div #grid5 + div.grid {color:green;}
div#grid6[title]{color:blue;}
#grid3 span{color:brown;}
```

1. Načrtněte, jak přibližně bude vypadat zobrazení daného HTML fragmentu s danými CSS styly v prohlížeči. Kde je to možné, uveďte rozměry jednotlivých elementů. [1 bod]
2. Vysvětlete princip určení specificity CSS selektorů (na čem specificita záleží, jak konkrétně se počítá). Dále pak odpovězte na následující otázky a svou odpověď stručně zdůvodněte (např. uveďte, který selektor se použije). [2 body]
   - Jaká je barva textu "GRID 3"?
   - Jaká je barva textu "GRID 4"?
   - Jaká je barva textu "GRID 6"?
   - Jaká je viditelná barva pozadí buňky v poslední řádce prostředního sloupce gridu?

14 Extremální teorie grafů (otázka studijního zaměření – 3 body)

1. Zformulujte Turánovu a Ramseyovu větu.
2. Ukažte, že pro každé přirozené k existuje N takové, že platí následující tvrzení: Uvažme obarvení hran úplného grafu s $n \geq N$ vrcholy třemi barvami (červená, modrá, zelená) takové, že nanejvýš $n^2/\log n$ hran je zelených. Pak pro nějakou podmnožinu K vrcholů tohoto grafu velikosti k platí, že buď všechny hrany mezi vrcholy K jsou červené, nebo všechny tyto hrany jsou modré.

15 Kempeho řetězce (otázka studijního zaměření – 3 body)

1. Popište, co jsou Kempeho řetězce.
2. Nechť v je vrchol stupně 4 v grafu G, a předpokládejme, že jeho sousedi indukují 4-cyklus v G a že graf $G - v$ lze obarvit čtyřmi barvami. Ukažte, že buď G lze obarvit čtyřmi barvami, nebo G obsahuje podrozdělení K_5.

16 Teorie množin (otázka studijního zaměření – 3 body)

1. Zformulujte princip maximality (Zornovo lemma).
2. Zformulujte Hypotézu kontinua.
3. Která z následujících tvrzení lze dokázat v teorii ZF (tj. bez axiomu výběru)? Která z následujících tvrzení lze dokázat v teorii ZFC (tj. s axiomem výběru)? Pro každou z otázek stačí napsat seznam, ale odpovědi můžete i krátce zdůvodnit.
   (a) Princip maximality
   (b) Hypotéza kontinua
   (c) Každou množinu lze dobře uspořádat.
   (d) Každou spočetnou množinu lze dobře uspořádat.
   (e) Každou konečnou množinu lze dobře uspořádat.

(f) Sjednocení spočetně mnoha spočetných množin je vždy spočetné.

(g) Je-li f zobrazení množiny a na množinu b, pak existuje prosté zobrazení $g\colon b \to a$ takové, že pro každé $y \in b$ platí $f(g(y)) = y$.

(h) Je-li f prosté zobrazení a do b a g prosté zobrazení b do a, pak a a b mají stejnou mohutnost.

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

Lexikální sémantika se zabývá studiem významu jednotlivých slov nebo výrazů přirozených jazyků. Existuje několik základních způsobů, jakými je možné popsat význam jednotlivých slov. Vysvětlete následující tři způsoby popisu významu slov a uveďte jejich výhody a nevýhody:

1. Ontologie
2. Sémantické sítě
3. Sémantické rysy

18 Základní formalismy pro popis přirozeného jazyka (otázka studijního zaměření – 3 body)

Uveďte základní charakteristiky následujících formalismů pro popis syntaxe přirozených jazyků:

1. Transformační gramatika
2. Unifikační gramatika
3. Funkční generativní gramatika

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

1. Načrtněte graf entropie binární náhodné veličiny. Popište a vysvětlete osy x a y.
2. Uvažujme dvě náhodné veličiny X a Y. Nakreslete diagram, který vizualizuje vztah mezi jejich entropiemi, podmíněnými entropiemi, sdruženou entropií a vzájemnou informací.

20 Scanline vyplňování (otázka studijního zaměření – 3 body)

Budeme se zabývat vybarvením vnitřku rovinného mnohoúhelníka v rastrovém prostředí (čtvercová mřížka pixelů), obrys útvaru není potřeba obkreslovat.

1. Jak by měl být mnohoúhelník zadán a jaké možnosti definice jeho vnitřku mají smysl?
2. Popište základní princip algoritmu řádkového vyplňování 2D mnohoúhelníka (uveďte i případné pomocné datové struktury používané v průběhu vyplňování).
3. Jak by se algoritmus zjednodušil, kdyby měl vyplňovat pouze konvexní útvary?

21 HDR grafika (otázka studijního zaměření – 3 body)

Otázka se týká HDR (High Dynamic Range) grafiky, principů a aplikací.

1. Z jakých důvodů a v jakých oblastech grafiky se používá HDR grafika? Uveďte příklady, kdy nám může pomoci nejvíce.
2. Jak se HDR obraz reprezentuje v paměti počítače a v jakém formátu ho můžeme zapsat na disk? Naznačte postup pořízení HDR obrázku s pomocí fotoaparátu a stativu.
3. Jak se HDR obraz prezentuje pozorovateli na běžném (LDR) výstupním zařízení? (není potřeba uvádět nějaké komplikované postupy, jen popsat princip)

22 Distribuované sledování paprsku (otázka studijního zaměření – 3 body)

Tyto techniky se také nazývají "Monte-Carlo ray tracing" nebo "Monte-Carlo integrace".

1. Vyjmenujte několik příkladů, kde je užitečné do rekurzivního sledování paprsku zapojit stochastický výpočet (Monte-Carlo integraci).
2. Vyberte si jedno konkrétní použití Monte-Carlo a popište ho detailně. Které veličina se integruje? Který nedostatek původního přístupu se odstraňuje nebo potlačuje?
3. Navrhněte vzorkování (sampling), které by se dalo dobře použít v předchozím příkladu. Stačí popsat metodu vzorkování rámcově nebo nakreslit obrázek. Šlo by toto vzorkování implementovat adaptivně?

23 Linkové protokoly (otázka studijního zaměření – 3 body)

Navrhněte hypotetický bitově 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.

24 IPv4 fragmentace (otázka studijního zaměření – 3 body)

Detailně vysvětlete následující aspekty fragmentace IPv4 datagramů:

1. Jaká je podstata a příčiny problému fragmentace na síťové vrstvě? Které uzly mohou provádět fragmentaci a které defragmentaci a proč?
2. Jakými postupy lze zabránit nutnosti provedení fragmentace? Do jaké míry tyto postupy fungují?
3. Jakým způsobem se datagram fragmentuje (jeho hlavička včetně volitelných doplňků a tělo)?
4. Jaké položky hlavičky k fragmentaci potřebujeme a jak fungují? Pokud si je nepamatujete, můžete vymyslet svoje vlastní, ale musíte zaručit potřebnou funkcionalitu.
5. Jak probíhá proces defragmentace?

25 Protokol ARP (otázka studijního zaměření – 3 body)

Detailně vysvětlete fungování protokolu ARP (Address Resolution Protocol):

1. K řešení jakého problému a v jakém kontextu protokol ARP používáme?
2. Jak vypadá struktura ARP zprávy? Pokud si jednotlivé položky nepamatujete přesně, můžete navrhnout vlastní, ale musíte zachovat požadovanou funkcionalitu.
3. Jakým způsobem a jakým uzlům je ARP dotaz rozeslán? Jak na něj jednotlivé uzly reagují?
4. Co je ARP Cache a jak a proč se používá? Jak se liší práce se statickými a dynamickými záznamy?

26 JSON (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.

Zapište příklad programu multikina dle výše uvedeného zadání ve formátu JSON. Příklad musí obsahovat alespoň 2 různé dny v týdnu, alespoň 2 sály a alespoň 2 promítání v každém dnu. Zapište schéma pro reprezentaci multikin ve formátu JSON dle Vašeho příkladu v jazyku JSON Schema.

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

Napište v jazyku JavaScript kód, který vaši JSON strukturu navrženou v předchozím příkladu převede do HTML podoby, kterou lze v prohlížeči zobrazit uživateli.

28 Základy technologií sémantického webu - Linked Data (otázka studijního zaměření – 3 body)

Uvažujte filmy z programu multikina z předchozích otázek. Na příkladu jednoho vybraného filmu vysvětlete datový model RDF a vysvětlete, jak lze dle 4 principů Linked Data propojit vaší RDF reprezentaci filmu s RDF reprezentací herce, který ve filmu hraje.

29 Rozhraní pro synchronizaci (otázka studijního zaměření – 3 body)

1. Uvažujte následující program:

```
1  #include <stdio.h>
2  #include <pthread.h>
3
4  static pthread_mutex_t mutex_one = PTHREAD_MUTEX_INITIALIZER;
5  static pthread_mutex_t mutex_two = PTHREAD_MUTEX_INITIALIZER;
6  static pthread_t thread_one;
7  static pthread_t thread_two;
8
9  void *thread_one_function (void *dummy) {
10     pthread_mutex_lock (&mutex_one);
11     pthread_mutex_lock (&mutex_two);
12     putchar ('A');
13     putchar ('B');
14     pthread_mutex_unlock (&mutex_one);
15     pthread_mutex_unlock (&mutex_two);
16     putchar ('C');
17     putchar ('D');
18     return (NULL);
19 }
20
21 void *thread_two_function (void *dummy) {
22     pthread_mutex_lock (&mutex_two);
23     pthread_mutex_lock (&mutex_one);
24     putchar ('a');
25     putchar ('b');
26     pthread_mutex_unlock (&mutex_two);
27     pthread_mutex_unlock (&mutex_one);
28     putchar ('c');
29     putchar ('d');
30     return (NULL);
31 }
32
33 void main (void) {
34     pthread_create (&thread_one, NULL, thread_one_function, NULL);
35     pthread_create (&thread_two, NULL, thread_two_function, NULL);
36     pthread_join (thread_one, NULL);
37     pthread_join (thread_two, NULL);
38 }
```

   Pokud volání `putchar` okamžitě zobrazí znak předaný jako parametr, jaké výstupy může program zobrazit ?

2. Závisí seznam možných výstupů z prvního bodu na tom, zda má použitý systém pouze jedno jádro (single processor) nebo více jader (multiprocessor) ? Vysvětlete a zdůvodněte.

3. Závisí pravděpodobnost pozorování možných výstupů z prvního bodu na tom, zda má použitý systém pouze jedno jádro (single processor) nebo více jader (multiprocessor) ? Vysvětlete a zdůvodněte.

30 Heap alokátory (otázka studijního zaměření – 3 body)

Uvažujte následující program:

```
1  #include <stdio.h>
2  #include <stdint.h>
3  #include <stdlib.h>
4
5  void main (void) {
6      for (int i = 0 ; i < 256 ; i++) {
7          void *p = malloc (i);
8          void *q = malloc (i);
9          int delta = (uintptr_t) q - (uintptr_t) p;
10         printf ("%i:%i␣", i, delta);
11     }
12 }
```

Jeden možný výstup programu je následující:

```
0:32 1:32 2:32 3:32 4:32 5:32 6:32 7:32 8:32 9:32 10:32
11:32 12:32 13:32 14:32 15:32 16:32 17:32 18:32 19:32 20:32
21:32 22:32 23:32 24:32 25:48 26:48 27:48 28:48 29:48 30:48
31:48 32:48 33:48 34:48 35:48 36:48 37:48 38:48 39:48 40:48
41:64 42:64 43:64 44:64 45:64 46:64 47:64 48:64 49:64 50:64
51:64 52:64 53:64 54:64 55:64 56:64 57:80 58:80 59:80 60:80
61:80 62:80 63:80 64:80 65:80 66:80 67:80 68:80 69:80 70:80
...
```

1. Vysvětlete, co dvojice čísel (pravděpodobně) vypovídají o použitém alokátoru.
2. Vysvětlete, proč je v každé dvojici čísel první menší než druhé. Musí tomu tak být vždy ? Pokud ne, napište příklad situace, kdy by to tak nebylo.
3. Vysvětlete, proč je druhé z čísel ve dvojici vždy násobek 16. Musí tomu tak být vždy ? Pokud ne, napište příklad situace, kdy by to tak nebylo.

31 Základy middleware (otázka studijního zaměření – 3 body)

Při volání vzdálených procedur i zasílání zpráv je potřeba řešit otázku kódování dat. Jednou z možností je použít technologii *protocol buffers*, které se týká tato otázka - pokud uvedenou technologii neznáte, je možné odpovědět také ze znalosti obecných principů.

1. Protocol buffers vyžadují oddělenou specifikaci formátu přenášených dat ve zvláštním specifikačním jazyce, tento popis se pak použije pro generování typů v konkrétním programovacím jazyce, například C++ nebo Javě. Vysvětlete, proč se volí tato cesta a přenášená data se nespecifikují přímo v cílovém programovacím jazyce.
2. Protocol buffers nabízí celkem 10 typů pro ukládání celých čísel (`int`, `uint`, `sint`, `fixed`, `sfixed`, vše v délce 32 nebo 64 bitů). Vysvětlete, proč se volí tato cesta a nedá se přednost řešení s jedním celočíselným typem, jehož velikost se upraví podle obsahu, jako je tomu například v jazyce Python.
3. Srovnejte následující definici datového typu v jazyce protocol buffers a v Javě (ignorujte viditelnost):

```
1  syntax = "proto3";
2  message Node {
3      int32 value = 1;
4      repeated Node neighbors = 2;
5  }
```

```
1  public class Node {
2      int value;
```

```
3       Node [] neighbors;
4   }
```

Jsou obě definice stejně použitelné pro uložení grafu ? Vysvětlete.