Bakalářské zkoušky (příklady otázek)

2023-06-27

1 Třídění sléváním (3 body)

1. Formulujte algoritmus třídění posloupnosti n celých čísel sléváním (Mergesort) a zapište ho v pseudokódu.
2. Analyzujte časovou a prostorovou složitost tohoto algoritmu.
3. Navrhněte efektivní algoritmus pro slévání k setříděných posloupností o celkem n prvcích. Upravte Mergesort, aby toto slévání používal. Jak se změní jeho časová složitost? Zapište ji jako funkci proměnných n a k.

2 Převod bezkontextové gramatiky na automat (3 body)

1. Uveďte definici *bezkontextové gramatiky*.
2. Sestrojte bezkontextovou gramatiku G generující následující jazyk:

$$L = \{a^i b^j \mid i, j \geq 0 \text{ a platí } 2i = 3j\}$$

3. Převeďte gramatiku G z předchozí části na zásobníkový automat přijímající prázdným zásobníkem. (Pokud se vám nepodařilo gramatiku sestrojit, sestrojte libovolný zásobníkový automat přijímající jazyk L prázdným zásobníkem.)

3 Lineární algebra (3 body)

Uvažujme dvě matice

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

1. Najděte vektor $x \in \mathbb{R}^2$ takový, aby vektory Ax, Bx byly na sebe kolmé (při standardním skalárním součinu).
2. Najděte matici lineárního zobrazení, které vektor Ax zobrazuje na vektor Bx pro libovolné $x \in \mathbb{R}^2$.
3. Rozhodněte, zda lineární zobrazení $x \mapsto B^{-1}A^3B^{-1}x$ plochy geometrických útvarů zvětšuje či zmenšuje.

4 Pravděpodobnost (3 body)

1. Napište větu o linearitě střední hodnoty (nedokazujte ji).
2. Ve skupině n lidí každý hodí (férovou) šestistěnnou kostkou. Každá dvojice lidí, kteří hodili stejné číslo, získá jeden bod. Označme Y počet bodů získaných jednou předem vybranou dvojicí. Určete střední hodnotu veličiny Y.
3. Označme X celkový počet získaných bodů za všechny dvojice (jeden člověk může skórovat ve více dvojicích). Určete střední hodnotu veličiny X.

5 Skrytý Markovův model (otázka studijního zaměření – 3 body)

Definujte skrytý Markovův model (HMM z anglického Hidden Markov Model). Co je Markovův předpoklad? Jaký je rozdíl mezi filtrací (filtering), predikcí (prediction) a vyhlazováním (smoothing)? Za použití vzorce

$$P(X_t|E_{1:t}) = \alpha P(E_t|X_t) \sum_{X_{t-1}} P(X_t|X_{t-1}) P(X_{t-1}|E_{1:t-1})$$

vyřešte následující příklad:

Předpokládejme, že proměnná X_i vyjadřuje jaké je počasí ve dne i. Každý den prší ($X_i = P$) nebo je slunečno ($X_i = S$). Tuto proměnnou nemůžeme přímo pozorovat, máme k dispozici pouze pozorování E_i, zda si náš kolega přinesl dnes deštník ($E_i = D$) nebo ne ($E_i = N$). Podmíněné pravděpodobnosti změny počasí z předchozího dne na dnešek a závislost pozorování deštníku na skutečném počasí uvádějí následující tabulky.

X_{i-1}	$P(X_i = P\|X_{i-1})$	$P(X_i = S\|X_{i-1})$
P	0.7	0.3
S	0.3	0.7

X_i	$P(E_i = D\|X_i)$	$P(E_i = N\|X_i)$
P	0.9	0.1
S	0.2	0.8

Předpokládejme, že ve dne 0 je pravděpodobnost deště $P(X_0 = P) = 0.5$ a slunečna $P(X_0 = S) = 0.5$. Jaká je pravděpodobnost deště ve druhém dni ($P(X_2 = P) =?$), pokud první i druhý den pozorujeme, že si náš kolega přinesl deštník ($E_1 = D$ a $E_2 = D$)?

6 Testování binárního klasifikátoru (otázka studijního zaměření – 3 body)

Předpokládejme, že k testování binárního klasifikátoru byla použita testovací sada ve složení 100 pozitivních a 400 negativních příkladů.

a) Ukázalo se, že ROC křivka prochází bodem TPR = FPR = 20 %. Vypočítejte hodnotu precision v tomto bodě.

b) Jiný klasifikátor vykazuje konstantní hodnotu precision = 50 %. Znázorněte odpovídající ROC křivku.

Použité zkratky: ROC — Receiver Operating Characteristic curve; TPR — true-positive rate (též: sensitivity neboli recall); FPR — false-positive rate (též: fall-out).

7 Rozhodovací stromy (otázka studijního zaměření – 3 body)

Chceme trénovat rozhodovací strom na datech v tabulce níže, která obsahují dva kategorické atributy x_1 a x_2. Cílem je předpovědět kategorickou hodnotu y.

Nápověda: Při řešení můžete potřebovat hodnoty některých logaritmů. Můžete použít následující aproximace: $\log_2 \frac{1}{3} = -1.5$, $\log_2 \frac{2}{3} = -0.6$

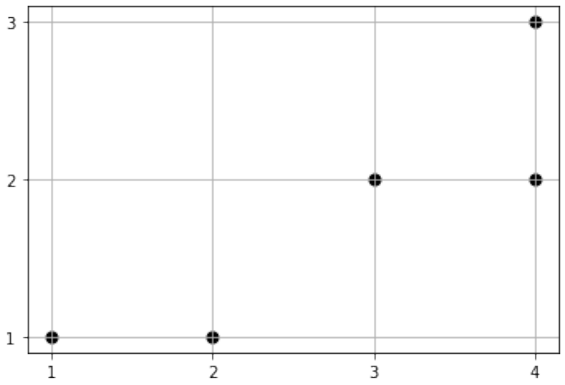
x_1	x_2	y
0	0	0
1	0	1
0	1	1
2	1	0
0	0	1
1	1	1
2	0	0
0	1	0

1. Popište stručně algoritmus trénování rozhodovacího stromu a kritéria pro větvení.
2. Určete, podle jakého atributu se bude strom větvit v kořeni stromu. Proč?
3. Je možné zadaná trénovací data klasifikovat se 100 % přesností (accuracy)? Vysvětlete.

8 Shlukování (otázka studijního zaměření – 3 body)

Uvažujme data zadaná v tabulce níže a zobrazená v obrázku. Budeme na ně chtít aplikovat shlukovací algoritmy.

x_1	x_2
1	1
2	1
3	2
4	3
4	2



1. Nejprve chceme aplikovat algoritmus k-means pro $k = 2$. Na počátku jsme náhodně zvolili jako středy shluků body $(1, 1)$ a $(3, 2)$. Jaké budou nové středy shluků po jedné iteraci algoritmu k-means?
2. Jak se změní polohy středů v dalších iteracích tohoto algoritmu?
3. Popište některý z algoritmů pro hierarchické shlukování. Jaký by byl jeho výsledek na uvedených datech?

9 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)

10 Model odrazivosti a stínování (otázka studijního zaměření – 3 body)

Tato otázka se týká lokálního modelu odrazu světla na povrchu tělesa při zobrazení 3D scény.

1. Popište jednoduchý lokální model odrazivosti světla (jak barva pozorovaná na povrchu tělesa závisí na poloze a intenzitě světelného zdroje a pozici pozorovatele).
2. Jak se započítávají příspěvky více světelných zdrojů, které může 3D scéna zobrazovat? Je vhodné zohlednit vzdálenost světelých zdrojů od tělesa?
3. Jak je možné dosáhnout efektu hladkého tělesa, i když je 3D model sestaven z trojúhelníků? Uvažujte postup: původně je povrch tělesa reprezentován exaktně matematicky jako hladká plocha (třeba v parametrickém tvaru). Pro uložení povrchu v paměti GPU se vytvoří síť trojúhelníků, která původní hladkou plochu aproximuje. Jak se takový povrch stínuje, aby nebyly vidět jednotlivé trojúhelníky?

11 Shadery v moderních GPU (otázka studijního zaměření – 3 body)

Půjde nám o principy programování moderních grafických karet (GPU) pomocí tzv. shaderů.

1. První základní (povinný) shader je tzv. “Vertex shader”. Uveďte, ve kterém místě 3D zobrazovacího řetězce (“3D pipeline”) je zařazen, které má typické vstupy a které typické výstupy?
2. Druhý povinný shader se nazývá “Fragment shader” nebo “Pixel shader”. Co je jeho úkolem, kde je do 3D pipeline zařazen a které vstupy a výstupy má?
3. Popište, kterými cestami/kanály se do shaderů dostávají data z aplikace. Přemýšlejte opravdu o všech možnostech, jak data do GPU dostat. V čem se jednotlivé přístupy liší a jaké mají výhody a nevýhody?

12 Osvětlení a stínování v počítačových hrách (otázka studijního zaměření – 3 body)

Budeme se zabývat rolí osvětlení při zobrazování 3D scén v počítačových hrách.

1. Z hlediska realističnosti zobrazení 3D objektů je možné použití několika rozdílných přístupů, přitom všechny mohou mít své opodstatnění ve vývoji her. Seřaďte podle věrnosti zobrazení ("fotorealističnosti") následující technologie od méně věrných k těm nejvěrnějším: Phongovo stínování, konstantní stínování ("flat shading"), Ray-tracing, drátový model ("wireframe"), Gouraudovo stínování. Která z nich mají podporu v moderních GPU a aspoň u dvou uveďte přibližný princip, jak pracují.
2. Vlastnosti povrchu (materiál) se dají v realtime grafice reprezentovat mnoha různými způsoby. Vyjmenujte co nejvíce metod, jak se pomocí dodatečných dat nebo sofistikovanějších algoritmů dá vylepšovat vzhled 3D objektů ve scéně.
3. Popište alespoň rámcově, jak se v realtime enginu mohou počítat tzv. "vržené stíny", tedy stíny, které vzniknou díky překážkám šíření světla ve 3D scéně. Vzpomenete si, jestli nám zde může pomoci technologie GPU? Zkuste popsat alespoň princip.

13 Směrování (routing) (otázka studijního zaměření – 3 body)

1. V operačním systému Linux lze obsah směrovací tabulky vypsat příkazem `ip route`. Uvažujte výstup příkazu `ip route`, který v situaci, kdy jsou všechna síťová rozhraní připojená, aktivní a funkční, vypadá následovně:

   ```
   default via 192.168.34.1 dev eth0 proto dhcp src 192.168.34.12 metric 100
   default via 192.168.1.1 dev wifi0 proto dhcp src 192.168.1.134 metric 600
   192.168.1.0/24 dev wifi0 proto kernel scope link src 192.168.1.134 metric 600
   192.168.34.0/24 dev eth0 proto kernel scope link src 192.168.34.12 metric 100
   ```

   Napište stručně účel směrovací tabulky a pro tři adresy zpracované různými způsoby ilustrujte fungování směrovací tabulky na konkrétním příkladu z výpisu výše.

2. Příkaz `traceroute 8.8.8.8`, který sleduje postup směrování paketu sítí, vypsal následující:

   ```
   traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
    1  _gateway (192.168.1.1)  23.944 ms  23.908 ms  23.893 ms
    2  192.168.34.1 (192.168.34.1)  24.070 ms  24.356 ms  24.344 ms
    3  10.30.252.1 (10.30.252.1)  77.091 ms  77.256 ms  78.076 ms
    4  217.195.160.209 (217.195.160.209)  79.568 ms  80.024 ms  80.170 ms
    5  mx204.nej.cz (217.195.167.217)  84.551 ms  85.014 ms  85.001 ms
   ...
   zbytek výpisu není pro otázku důležitý
   ```

   Je tento výpis konzistentní s výše uvedenou směrovací tabulkou a proč? Pokud ne, napište, k jaké změně v konfiguraci pravděpodobně došlo.

3. Další spuštění příkazu `traceroute 8.8.8.8` vypadá následovně:

   ```
   traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
    1  _gateway (192.168.34.1)  0.523 ms  0.552 ms  0.726 ms
    2  10.30.252.1 (10.30.252.1)  3.038 ms  3.075 ms  3.016 ms
    3  217.195.160.209 (217.195.160.209)  4.453 ms  3.970 ms  4.433 ms
    4  mx204.nej.cz (217.195.167.217)  10.283 ms  10.377 ms  10.367 ms
   ...
   zbytek výpisu není pro otázku důležitý
   ```

   Načrtněte podle tohoto a předchozího výpisu schéma zapojení sítě nejbližšího okolí počítače, jehož směrovací tabulku znáte z první otázky. Kde je to možné, tak síťovým rozhraním přidejte IP adresy.

14 Synchronizace (otázka studijního zaměření – 3 body)

1. Načrtněte rozhraní semaforu a popište sémantiku jeho operací.

2. Pomocí rozhraní z prvního bodu implemenujte datovou strukturu a funkce pro řešení problému producent-konzument, jejichž skica je uvedena níže. Není potřeba implementovat vlastní datovou strukturu pro udržení seznamu vyprodukovaných (ale nespotřebovaných) dat, stačí popsat její rozhraní. To ale samo o sobě nesmí poskytovat žádné záruky ohledně synchronizace.

```
typedef struct {
    ...
} prodcons_t;

void prodcons_init (prodcons_t *prodcons, int capacity) { ... }

void prodcons_produce (prodcons_t *prodcons, void *item) { ... }
void *prodcons_consume (prodcons_t *prodcons) { ... }
```

V odpovědi používejte programovací jazyk C nebo C++, ale kromě rozhraní z prvního bodu nepoužívejte žádné další nástroje pro synchronizaci.

15 Stránkování (otázka studijního zaměření – 3 body)

Předpokládejte procesor s virtuálními a fyzickými adresami velikosti 4 B, který pro překlad virtuálních adres používá softwarově plněný TLB s pevnou velikostí stránky 4 MB.

Položka v TLB má velikost 4 B a obsahuje (po řadě od nejvyšších bitů) potřebnou část virtuální a fyzické adresy, operačním systémem přiřazený ASID (identifikátor adresového prostoru) a bity dirty a valid. Zbylé dva nejnižší bity jsou rezervované jako 0.

1. Načrtněte formát položky TLB včetně počtu bitů jednotlivých polí.
2. Předpokládejte následující obsah TLB:

   ```
   0x35C0C124
   0x35C0613C
   0x35C08120
   0x36005134
   0x36009128
   0x3600D12C
   0x3640E124
   0x3640A120
   0x3640413C
   ```

   Předpokládejte, že aktuální ASID je nastaven na hodnotu `0x12` a dále se nemění.

   Co se stane při přístupech na následující virtuální adresy ? V případě, že dojde k úspěšnému překladu, napište konkrétní fyzickou adresu, ke které procesor přistoupí. V případě, že překlad nebude moci procesor provést, ve stručnosti popište činnost procesoru a případnou (typickou) reakci operačního systému pro napravení situace.

   Jednotlivé přístupy uvažujte izolovaně, tedy vycházejte vždy z obsahu TLB uvedeného výše, i kdyby v některém kroku případná reakce operačního systému mohla vést ke změně TLB.

   - zápis na `0x35803580`
   - čtení z `0x35CD35CD`
   - čtení z `0x36003600`
   - zápis na `0x36403640`

   Odpovědi přiměřeně komentujte tak, aby bylo možné vidět, jak jste dospěli k výsledku.

16 Řízení překladu (otázka studijního zaměření – 3 body)

Mějme následující popis pro řízení překladu jednoduchého C programu ze dvou zdrojových souborů:

```
all: app.bin

app.bin: main.o lib.o
```

```
	$(LD) -o app.bin $(LDFLAGS) main.o lib.o

%.o: %.c
	$(CC) -o $@ -c $(CFLAGS) $<
```

Spuštění programu `make` nad výše uvedeným souborem vypíše následující

```
gcc -o main.o -c -g -O2 main.c
gcc -o lib.o -c -g -O2 lib.c
ld -o app.bin  main.o lib.o
```

V případě změny souboru `main.c` je výpis zkrácen na následující:

```
gcc -o main.o -c -g -O2 main.c
ld -o app.bin  main.o lib.o
```

1. Vysvětlete, proč je v uvedené situaci druhý výpis kratší než první. Napište, jaké jsou hlavní přednosti výše uvedeného popisu pro řízení překladu ve srovnání s přímočarým skriptem, který by obsahoval přímo výše vypsané příkazy.
2. Na příkladu dvojřádky níže vysvětlete hlavní koncepty použitého nástroje (uvedené koncepty identifikujte také na dané dvojřádce):

   ```
   app.bin: main.o lib.o
   	$(LD) -o app.bin $(LDFLAGS) main.o lib.o
   ```

3. Odhadněte vnitřní fungování programu `make`, zejména jakou datovou strukturu si (pravděpodobně) vystaví a jakým způsobem určí, v jakém pořadí (a jestli vůbec) mají být jednotlivé příkazy spuštěny.

17 Překlad adresy (otázka studijního zaměření – 3 body)

Předpokládejme hypotetický procesor s paměťovou architekturou, kde virtuální i fyzický prostor používá 16-bitové adresování. Pro překlad adres se používá 1-úrovňové stránkování s 64 B stránkami. Záznamy ve stránkovací tabulce (které mají také 16 bitů) využívají offsetové bity pro přidružené příznaky, nejméně významný bit je použit jako příznak platnosti (tj. 1 = záznam je platný, 0 = záznam není platný).

Stránkovací tabulka, jejíž výpis následuje níže, se nachází na adrese `0x1d40`. Výpis zobrazuje pro jednoduchost přímo 16-bitové položky stránkovací tabulky, tedy není nutné řešit pořadí bytů v reprezentaci čísel.

	+0x0	+0x2	+0x4	+0x6	+0x8	+0xA	+0xC	+0xE
0x1d40	0xdecd	0x0149	0x21c9	0x2b19	0x8f85	0x65d9	0x68a1	0x2151
0x1d50	0x2fb1	0x8249	0xabc5	0x3219	0xeacd	0x2651	0x9111	0x4249
0x1d60	0x5085	0x8e4d	0xe731	0xb109	0xb751	0x7fe2	0x3d6a	0xcf24
0x1d70	0x9c9c	0x6011	0x3d11	0xd589	0xc14d	0x748d	0xa019	0x41c9
0x1d80	0x0320	0x0e2a	0xd2d6	0xb0a8	0x8b44	0x3aa2	0xe4ce	0xcde0
0x1d90	0x5068	0xfbf8	0x427e	0x007c	0x7bba	0x578e	0x5770	0x4f76
0x1da0	0x85b6	0xe920	0x7768	0xcaf0	0xe3c0	0x76fa	0xc84a	0xdc66
0x1db0	0x94d4	0x3fb2	0xb986	0xe52c	0x1ddc	0x957e	0x8ccc	0x82a2
0x1dc0	0x4f3c	0xf6d2	0x8fc2	0xee90	0x9792	0xa43a	0xc590	0x9180
0x1dd0	0x1386	0x2b92	0xdcf0	0x62a0	0x47d8	0x9ede	0x9614	0x8d9c
0x1de0	0xda10	0x16d0	0xc68c	0xa108	0x45c8	0x1f60	0x4a18	0xe9ce
0x1df0	0x4c4f	0xbbe9	0x9535	0x74eb	0x6b19	0x7aad	0x36b1	0x85fd
0x1e00	0xcfe3	0x7dd3	0x4917	0x54ad	0x9391	0x92c5	0x0ab9	0x9fa7
0x1e10	0xd390	0xddfa	0x6330	0x33a2	0xad22	0xabbe	0xe21a	0x04dc
0x1e20	0x43f8	0x4188	0xfe3c	0xba68	0xb082	0x2040	0x126e	0xdecc
0x1e30	0xc206	0xb9a2	0xbe10	0x0cf8	0x81c8	0x564c	0x8140	0x923a
0x1e40	0x5bda	0xbd12	0xb794	0x910c	0x05f2	0x7636	0x9bd8	0xdede
0x1e50	0x85f2	0x21ca	0xf3c8	0x1bc2	0xde4e	0xa87c	0x9a5e	0xc348
0x1e60	0x80e0	0x37ea	0x39a8	0x7ef6	0xd8ca	0x79f4	0x286c	0x7f00
0x1e70	0xb046	0x0fe8	0x5256	0x72e8	0xe0cc	0xf6aa	0x39d8	0x8c4e
0x1e80	0xdd44	0x5088	0x789a	0x68d0	0x1748	0xfbb0	0x9130	0x8e9e
0x1e90	0x13bf	0xe44f	0x525f	0x8715	0x5389	0x849f	0xaee1	0xd451
0x1ea0	0x584a	0xe278	0x6776	0x32b0	0x82b0	0x8b1c	0x704c	0x2e8a
0x1eb0	0xabce	0x4020	0xd434	0xaeea	0x6548	0xdf10	0x3ad8	0x388a
0x1ec0	0xb134	0x8d2c	0xe422	0x6ec8	0x101d	0xa595	0x0b55	0x7cf1
0x1ed0	0x69b9	0x431b	0xfbf7	0x293d	0x0075	0xd8d7	0x9059	0x1569
0x1ee0	0x4c73	0x8577	0x6efb	0x2e09	0xa58d	0xd858	0x39d2	0x00d8
0x1ef0	0x5eb8	0x92e6	0x5798	0x2314	0xb82c	0x34c2	0xbe16	0xbed6
0x1f00	0x4b12	0x4660	0xb082	0x2eee	0x3ee4	0x9140	0x4b42	0xef9e
0x1f10	0x8318	0x5a74	0x7aa2	0xda8a	0x1504	0x5cf4	0x9876	0x3632
0x1f20	0xb5be	0x802e	0xab94	0x0c46	0xd588	0xb138	0xc378	0x5a8a
0x1f30	0x94ea	0x1254	0x0bac	0xaef2	0x2c6a	0xc5de	0x5b12	0xca64

Přeložte virtuální adresu `0x328f` (`0b0011001010001111` binárně) na fyzickou adresu:

1. Napište číslo stránky a offset pro danou virtuální adresu.
2. Napište fyzickou adresu položky ve stránkovací tabulce, která odpovídá zadané virtuální adrese.
3. Napište výslednou fyzickou adresu, pokud existuje překlad. Jinak uveďte, že neexistuje.

V odpovědi důsledně uvádějte číselnou soustavu - čísla bez prefixu jsou v desítkové soustavě, prefix `0x` označuje šestnáctkovou soustavu, prefix `0b` označuje dvojkovou soustavu. (To, jakou soustavu použijete, se ovšem samo o sobě nehodnotí.)

18 Komponentový systém (otázka studijního zaměření – 3 body)

Naprogramujte v C++, C# nebo Javě všechny níže zmíněné a další potřebné typy (včetně zmíněných metod) pro implementaci následujícího komponentového herního frameworku. Ve hře vytvořené pomocí vašeho frameworku má každý objekt ve scéně (např. ležící kámen, nebo odpočívající trol) být reprezentovaný instancí typu `GameObject` (dále jen GO). Samotný GO má reprezentovat množinu komponent – kde komponenta je pro nás libovolný vhodný typ (dle vašeho návrhu), jehož instance přidává k instanci GO nějaká silně typovaná data, a/nebo nějakou další funkcionalitu. Od každého druhu komponenty může být na instanci GO připojena právě 0 nebo 1 instance daného druhu komponenty. Každá instance nějaké komponenty může být připojena na maximálně 1 GO.

1. Na GO by měla být vhodná metoda pro připojení instance nějaké komponenty ke GO – pokud GO komponentu stejného druhu již má, tak nová instance nahradí instanci původní.

   Na GO by měla být vhodná metoda pro vrácení instance komponenty zvoleného druhu (požadovaný druh si nějak vhodně může zvolit volající). Pokud na GO žádná instance požadovaného typu komponenty připojená není, tak to má funkce nějak rozumně indikovat (funkci budeme chtít běžně používat pro detekci, zda daný typ komponenty na GO je, nebo není).

Na GO má být metoda `Update` s parametrem `timeDelta` jako 32-bitové float číslo, které reprezentuje čas v sekundách od předchozího volání metody `Update` na stejném GO. Metoda `Update` na GO má projít všechny instance připojených komponent, a pro komponenty, které mají metodu `Update` se stejným parametrem, se má tato zavolat. Pořadí volání metod `Update` na komponentách GO není důležité. Ve svém objektovém návrhu vhodně zohledněte, že autor komponenty má mít možnost se rozhodnout, zda komponenta bude metodu `Update` mít nebo nikoliv.

2. Napište implementaci komponenty `Position`, která ke GO přidává 32-bitové float souřadnice X a Y reprezentující polohu GO na 2D herní ploše. Napište implementaci komponenty `Health`, která ke GO přidává 32-bitové float hodnoty maximálního a aktuálního zdraví (aktuální zdraví je vždy v rozsahu 0 až max včetně).

   Napište příklad kódu, který vytvoří instanci GO pro kámen, který má komponentu `Position`, a instanci GO pro trola, který má komponenty `Position` a `Health` (max = 30, aktuální = 20).

3. Doplňte vhodné komponenty reprezentující efekt otravy a efekt krvácení – oba efekty vyžadují, aby GO, ke kterému jsou připojené, měl komponentu `Health` (pokud GO takovou komponentu nemá, tak efekt nic nedělá). Dále jsou oba efekty parametrizované 32-bit float hodnotou, která reprezentuje, kolik aktuálního zdraví efekt odečítá každou sekundu. Pokud je efekt připojený k nějakému GO, tak při zavolání `Update` na GO má dojít k úpravě zdraví GO dle parametru efektu. Na jednom GO může být maximálně 1 efekt otravy a nezávisle maximálně 1 efekt krvácení (tedy GO může být bez efektu, nebo mít 1 otravu a žádné krvácení, nebo žádnou otravu a 1 krvácení, nebo 1 otravu a 1 krvácení). Do budoucna budeme přidávat další podobné efekty.

   Napište implementaci funkce, která dostane seznam GO a hodnotu zranění za sekundu – funkce projde všechny předané GO a pokud GO má komponentu `Health`, tak ke GO připojí novou instanci efektu otravy nebo krvácení parametrizované předanou hodnotou zranění za sekundu. Volající musí mít navíc možnost si nějak vhodně zvolit, zda funkce aplikuje na předané GO efekt otravy nebo efekt krvácení.

   Napište příklad kódu, který na výše vytvořený kámen a trola zkusí aplikovat efekt krvácení s hodnotou zranění 1.2 za sekundu.

19 Architektury databázových systémů (otázka studijního zaměření – 3 body)

Uvažujte tabulky PROJECT a EMPLOYEE se schématy PROJECT(ID, Name, Budget) a EMPLOYEE(ID, Name, Position, Salary, PID), kde PID $\subseteq$ PROJECT.ID.

- Je uvedené schéma ve 3NF, pokud víte, že firemní politika vyžaduje funkční závislost

  EMPLOYEE.Position $\rightarrow$ EMPLOYEE.Salary?

  Pokud ano, zdůvodněte. Pokud ne, upravte schéma, aby bylo ve 3NF.

- Pro výsledné relační schéma nakreslete ekvivalentní konceptuální model pomocí jazyka E-R, případně UML.

- Napište nad uvedeným schématem v jazyce SQL dotaz, který vrátí seznam projektů i s počty zaměstnanců, kteří na nich pracují.

20 Web chat (otázka studijního zaměření – 3 body)

Uvažme jednoduchou webovou aplikaci s přihlašováním, ve které uživatelé sdílejí krátké textové zprávy (chat). Přihlašovací formulář v HTML vypadá přibližně takto:

```
<form action="?action=login" method="POST">
  Login: <input type="text" name="login">
  Password: <input type="password" name="password">
  <button type="submit">Sign in</button>
</form>
```

Front controller webové aplikace (jeho nejpodstatnější části) vypadají následovně.

```
if (($_GET['action'] ?? '') === 'login') {
    if (verify_credentials($_POST['login'], $_POST['password'])) {
        $res = $db->query("SELECT id FROM user WHERE login = '" . $_POST['login'] . "'");
        $userId = $res->fetch_column(0);
        if ($userId) setcookie('user', $userId);
    } else {
        showWrongCredentialsError();
```

```
    }
    redirectAndExit();
}
// ...
if (empty($_COOKIE['user'])) {
    showLoginForm();
} else {
    showChatMessagesForUser($db, $_COOKIE['user]);
}

// ... v pomocných inkludovaných souborech se také nachází
function showChatMessagesForUser($db, $userId) {
    // ...
    $messages = loadMessagesForUser($db, $userId);
    foreach ($messages as $m) {
        echo "<p>[$m->from]: $m->text</p>";
    }
    // ...
}
```

Můžete předpokládat, že chybějící části kódu (včetně funkcí) jsou rozumně implementovány (a neobsahují bezpečnostní chyby). Např. proměnná `$db` obsahuje Mysqli objekt reprezentující spojení k databázi, superglobální proměnné nebyly skriptem modifikovány atd.

Identifikujte všechny bezpečnostní chyby ve výše uvedených příkladech. U každé chyby stručně popište možný způsob zneužití (v případě injection útoků uveďte zejména, jaký řetězec by byl injektován) a způsob opravy (načrtněte krátký kód, nebo stručně popište, jaké změny je třeba v kódu provést).

Pokud v kódu žádné zranitelnosti nejsou, popište stručně jeden možný typ injection útoku (na tuto konkrétní aplikaci) a identifikujte bezpečnostní mechanismy v kódu, které tomuto útoku zabrání.

21 Přehled SQL (otázka studijního zaměření – 3 body)

Uvažujte tabulky PROJECT a EMPLOYEE se schématy PROJECT(ID, Name, Budget) a EMPLOYEE(ID, Name, Position, PID), kde PID $\subseteq$ PROJECT.ID. Tabulka PROJECT obsahuje záznamy ([1, 'SIS', 500000], [2, 'FIS', 400000], [3, 'CARS', 400000]). Tabulka EMPLOYEE obsahuje záznamy ([1, 'Alex', 'Consultant', 1], [2, 'Peter', 'Programmer', 3], [3, 'John', 'Programmer', 2], [4, 'Paul', 'Designer', 2]).

Zapište výsledek následujících dotazů, v případě chyby v SQL dotazu místo s chybou vyznačte.

1. Select Min(Budget) From PROJECT
2. Select Name From EMPLOYEE Inner Join PROJECT On (Name = Name)
3. Select Name From PROJECT Where Budget = Min(Budget)
4. Select Name From PROJECT Where ID Not In (Select PID From EMPLOYEE)
5. Select PID, Count(ID) From EMPLOYEE Where Position='Programmer' Group By PID Having Count(ID) > 1

22 Transakční zpracování (otázka studijního zaměření – 3 body)

Uvažujte transakce T_1: $W(B)$ $W(C)$ $COMMIT$, T_2: $R(A)$ $W(B)$ $COMMIT$ a T_3: $W(A)$ $W(C)$ $COMMIT$.

Je rozvrh S: $W_3(A)$ $R_2(A)$ $W_1(B)$ $W_2(B)$ $W_1(C)$ $W_3(C)$ $COMMIT_1$ $COMMIT_2$ $COMMIT_3$ konfliktově uspořádatelný (conflict-serializable)? Pokud ano, uveďte příklad konfliktové ekvivalentního sériového rozvrhu. Pokud ne, vysvětlete proč.

Je rozvrh S zotavitelný (recoverable)? Pokud ano, proč? Pokud ne, jak by bylo potřeba jej upravit, aby zotavitelný byl?

Operace R reprezentuje čtení dané proměnné, operace W reprezentuje její zápis.

23 Datové formáty (otázka studijního zaměření – 3 body)

Uvažujme následující JSON-Schéma:

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "properties": {
    "version": { "type": "string" },
    "content": {
      "type": "array",
      "items": {
        "oneOf" : [
          { "type": "string" },
          { "type": "number" }
        ]
      }
    }
  }
}
```

a JSON dokument:

```
{
  "node": "My document",
  "content": [ "Ailish", 2, [3] ]
}
```

1. Rozhodněte, zda je daný JSON dokument validní podle uvedeného JSON-schématu. Pokud není validní, popište co do rozsahu změny minimální variantu úpravu JSON dokumentu aby validní byl.
2. Naznačte, jakým způsobem by bylo nutné JSON dokument rozšířit, aby bylo možné na něj nahlížet jako na RDF data. Není dovoleno měnit existující klíče ani hodnoty v JSON dokumentu.

24 Základy indexování (otázka studijního zaměření – 3 body)

Uvažujme následující neredundantní 3-rní B-strom. Kořenem stromu je uzel [5,12]. Potomky stromy jsou pak zleva doprava uzly [1,] [6, 7] [13, 14].

1. Graficky znázorněte výsledný stav B-stromu po vložení nové hodnoty 20.
2. Mohl by výše uvedený příklad B-stromu být validním redundantním stromem? Odpověď zdůvodněte.
3. Vysvětlete rozdíl mezi B-stromem a B*-stromem.

25 Kombinatorika (otázka studijního zaměření – 3 body)

Nechť $a(x) = \sum_{n=0}^{\infty} a_n x^n$ a $b(x) = \sum_{n=0}^{\infty} b_n x^n$ jsou vytvořující funkce posloupností $(a_0, a_1, \dots)$ a $(b_0, b_1, \dots)$.

1. Určete členy posloupnosti $(d_0, d_1, \dots)$, jejíž vytvořující funkcí je $d(x) = \sum_{n=0}^{\infty} d_n x^n = a'(x)$ (derivace funkce $a(x)$).
2. Určete členy posloupnosti $(c_0, c_1, \dots)$, jejíž vytvořující funkcí je $c(x) = \sum_{n=0}^{\infty} c_n x^n = a(x) \cdot b(x)$.
3. Použitím vytvořujících funkcí vyjádřete členy a_n, které jsou zadány následující rekurencí: $a_{n+2} = \frac{a_{n+1}+a_n}{2}$ pro každé $n \geq 0$, přičemž $a_0 = 1$ a $a_1 = 2$,

26 Optimalizace (otázka studijního zaměření – 3 body)

1. Definujte pojem konvexní obal množiny vektorů.
2. Je dán orientovaný graf $G = (V, E)$ a dva vrcholy $s, t \in V$, $s \neq t$ takové, že v G existuje orientovaná cesta z s do t. Uvažte následující lineární program P:

$$\min \sum_{e \in E} x_e$$
$$\sum_{u:su \in E} x_{su} - \sum_{u:us \in E} x_{us} = 1$$
$$\sum_{u:tu \in E} x_{tu} - \sum_{u:ut \in E} x_{ut} = -1$$
$$\sum_{u:vu \in E} x_{vu} - \sum_{u:uv \in E} x_{uv} = 0 \ \forall v \in V \setminus \{s,t\}$$
$$0 \leq x_e \leq 1 \ \forall e \in E$$

Nechť $P_1(G,s,t)$ označuje množinu všech optimálních řešení P.

Nechť $\mathcal{P}$ označuje množinu všech nejkratších cest z s do t v G. Pro $p \in \mathcal{P}$ nechť x_p označuje charakteristický vektor cesty p, a nechť $P_2(G,s,t)$ označuje konvexní obal množiny $\{x_p \mid p \in \mathcal{P}\}$.

Rozhodněte a zdůvodněte, která z následujících tvrzení platí pro každý graf G:

(a) $P_1(G,s,t) \subseteq P_2(G,s,t)$

(b) $P_2(G,s,t) \subseteq P_1(G,s,t)$

(c) $P_1(G,s,t) = P_2(G,s,t)$.

27 Geometrie (otázka studijního zaměření – 3 body)

1. Jaký je maximální počet hran konvexního mnohostěnu v $\mathbb{R}^4$ s 2023 vrcholy? (Stačí vzoreček pro výpočet.)
2. Popište konstrukci nějakého takového mnohostěnu, například pomocí souřadnic vrcholů.
3. Co nejlépe zdůvodněte, proč mnohostěn zkonstruovaný v části 2 má počet hran uvedený v části 1. (Alternativně vyřešte analogii otázek 1. a 2. v $\mathbb{R}^3$.)

28 Pokročilá diskrétní matematika (otázka studijního zaměření – 3 body)

1. Definujte pojem "homeomorfismus" a rozhodněte, zda jsou prostory $\mathbb{R}$ a $[0,1]$ homeomorfní.
2. Definujte pojem "rod plochy" a určete rod toru (anuloidu).
3. Rozhodněte, zda existuje nakreslení (bez křížení hran) Petersenova grafu na toru.

29 Pokročilé algoritmy a DS (otázka studijního zaměření – 3 body)

1. Definujte vyhledávací automat z Knuthova-Morrisova-Prattova algoritmu pro vyhledávání podřetězce („jehly“) v textu („seně“).
2. Nakreslete, jak bude automat vypadat pro jehlu `kockoskocka`.
3. Zapište v pseudokódu algoritmus pro vyhledávání pomocí tohoto automatu.
4. Jakou složitost má konstrukce automatu a jakou hledání pomocí automatu? Obojí zapište jako funkci délky jehly J a délky sena S. Není třeba dokazovat.
5. Jak obě složitosti závisí na velikosti abecedy?

30 Pokročilá MA (otázka studijního zaměření – 3 body)

Nechť $f(x,y,z,u,t) = ax^4 + by^4 + cz^4 + du^4 + et^4$ je funkce pěti proměnných x, y, z, u, t definovaná na $\mathbf{R}^5$, přičemž a, b, c, d, e jsou reálné parametry.

1. Vypočítejte parciální derivace $\partial^3 f / \partial z^3$ a $\partial^3 f / \partial x^2 \partial t$.
2. Nalezněte všechny hodnoty parametrů, pro něž f má v počátku (tj. v bodě $(0,0,0,0,0)$) ostré lokální minimum. Poznámka: tuto úlohu lze řešit pomocí parciálních derivací i bez nich.

Své řešení zdůvodněte.