Bakalářské zkoušky (příklady otázek)

jaro 2023

1 Třídění (3 body)

1. Popište, jak pomocí binárního vyhledávacího stromu setřídit (uspořádat vzestupně) posloupnost $x_1, \ldots, x_n$ navzájem různých prvků.
2. Co se změní, pokud se prvky mohou opakovat?
3. Co z toho plyne pro minimální možnou složitost operací s vyhledávacím stromem? Předpokládejme přitom, že klíče ve stromu je možné pouze porovnávat.
4. Jakou bude mít váš algoritmus z bodu 1 časovou složitost, pokud jako strom použijeme AVL strom a prvky posloupnosti budou řetězce délky L?

2 Objektový návrh (3 body)

Předpokládejte, že implementujeme aplikaci pro úpravu bitmapových obrázků. Uživatel bude moci mít otevřených více obrázků najednou. Nový obrázek si uživatel otevře v uživatelském rozhraní aplikace (UI) výběrem jména souboru – aplikace dle přípony souboru rozhodne o jeho formátu. U otevřeného obrázku si uživatel může postupně vybírat některé z nabízených filtrů – v UI aplikace si uživatel vždy vybere filtr a jeho konkrétní nastavení, ten se uloží do seznamu vybraných filtrů. Nakonec bude mít uživatel možnost všechny filtry najednou aplikovat. Každý otevřený soubor si uživatel může kdykoliv uložit buď v původním nebo jiném podporovaném bitmapovém formátu. Pro jednoduchost předpokládejte, že jednomu formátu obrázku odpovídá právě jedna přípona souboru (tj. např. pro JPEG formát jen přípona `.jpg` a nikoliv už `.jpeg`), a že všechny filtry mohou libovolně měnit obrazová data, ale rozměry obrázku v pixelech vždy zůstávají zachované.

Zatím máme v C# připravenou níže uvedenou kostru hlavních tříd našeho programu, kde třída `Image` reprezentuje načtená obrazová data (další metadata jako původní jméno souboru atd. si budeme pamatovat jinde; poznámka: `[,]` v C# znamená dvojrozměrné pole), a metoda `Program.Main` představuje jednoduchý scénář využití uvedených tříd:

```
struct Color { public byte R, G, B; }

static class Filters {
        public static void ApplySharpness(Image image, float amount, float radius) { ... }
        public static void ApplyBrightness(Image image, float amount) { ... }
}

abstract class Image {
        public Color[,] bitmap = null;
        public abstract void Load(string fileName);
        public abstract void Save(string fileName);
}

class PngImage : Image {
        public override void Load(string fileName) { ... }
        public override void Save(string fileName) { ... }
}

class JpegImage : Image {
        public override void Load(string fileName) { ... }
        public override void Save(string fileName) { ... }
}
```

```
static class ImageLoader {
        public static Image Load(string fileName) {
                Image image;
                if (fileName.EndsWith(".png")) {
                        image = new PngImage();
                } else if (fileName.EndsWith(".jpg")) {
                        image = new JpegImage();
                } else {
                        throw new ArgumentException();
                }
                image.Load(fileName);
                return image;
        }
}

class Program {
        public static void Main() {
                var fileName = "a.jpg";
                var image = ImageLoader.Load(fileName);
                Filters.ApplySharpness(image, 9.0f, 1.3f);
                Filters.ApplyBrightness(image, 1.7f);
                image.Save(fileName);
        }
}
```

Přepracujte celý objektový návrh tak, aby lépe umožnoval implementovat, udržovat a dále rozšiřovat uvedenou aplikaci. Novou kostru kódu zapište v C#, C++, nebo Javě. Soustřeďte se na návrh typů a jejich vztahů (vše zapisujte v syntaxi zvoleného jazyka), deklaraci metod a pouze klíčových vlastností (properties) a datových položek. Implementaci většiny metod uvádět nemusíte – pouze upravte implementaci `Program.Main` tak, aby vhodně využívala váš OO návrh. Ve svém řešení uveďte také implementaci kódu, který bude ve vašem návrhu implementovat posloupnost kroků od jména obrazového souboru k výběru formátu souboru a načtení obrázku v tomto formátu.

Ve svém návrhu zohledněte nejen snadnou možnost implementace všech aspektů výše načrtnutého scénáře používání aplikace, ale také to, že do budoucna budeme chtít přidávat velké množství různých filtrů – implementaci nového filtru ale vždy plánujeme doplňovat do hlavních zdrojových kódů aplikace; nepředpokládáme, že bychom nové filtry dodávali ve formě pluginů. Tedy ve svém řešení zohledněte pouze možnost batchové aplikace předvybraných filtrů, ale nemusíte řešit žádnou komplikovanou infrastrukturu pro jejich obecný popis.

Formou pluginů naopak budeme chtít přidávat podporu pro nové obrazové formáty. Předpokládejte tedy, že každý nově podporovaný obrazový formát budeme implementovat ve formě nového pluginu naší aplikace (který budeme za běhu načítat jako dynamicky linkovanou knihovnu) – chceme tedy, aby proces načítání obrázků ze souboru byl dostatečně flexibilní a umožňoval nám za běhu dodat odkaz na nový formát souborů, který bychom nalezli v nějakém pluginu (samotné načítání pluginů a podporu pro načítání dynamicky linkovaných knihoven, ani přímo hledání popisu formátu v pluginu neřešte).

3 Lineární zobrazení (3 body)

1. Definujte *matici lineárního zobrazení vzhledem k zadaným bazím.*
2. Formulujte tvrzení o matici složeného zobrazení, tj. jak se matice lineárního zobrazení mění při skládání zobrazení.
3. Uvažme lineární zobrazení $g : \mathbb{Z}_5^2 \to \mathbb{Z}_5^2$ splňující $f \circ g \circ h = id$, přičemž lineární zobrazení f a h jsou dána následovně:

$$f((1,0)^T) = h((0,1)^T) = (3,4)^T,$$
$$f((0,1)^T) = h((1,0)^T) = (4,1)^T.$$

   Najděte matici g vzhledem ke kanonickým bazím.

4 Rovinné grafy (3 body)

1. Uveďte Eulerovu formuli pro rovinné grafy (o počtu vrcholů, hran a stěn).
2. Rozhodněte, zdali může existovat rovinný graf G na 40 vrcholech s 80 hranami a 5 komponentami souvislosti takový, že neobsahuje C_3 jako podgraf.

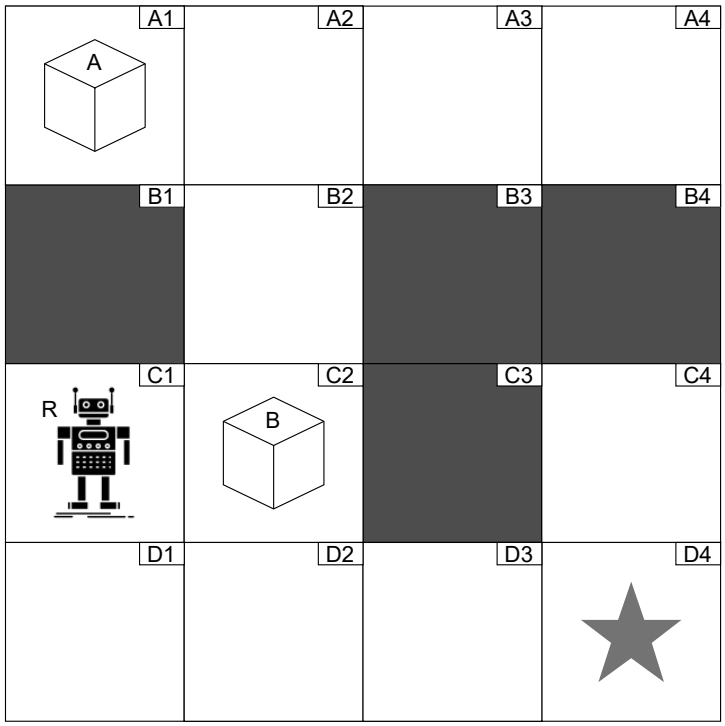
3. Určete, jaký nejvyšší možný počet hran může mít rovinný graf, který má 40 vrcholů, 5 komponent souvislosti a neobsahuje C_3 jako podgraf.

5 Plánování (3 body)

Mějme robota, který může provádět následující akce

- Jdi – přesun na sousední políčko, pokud na něm není zeď; pokud robot nese krabici, nesmí navíc na políčko s jinou krabicí
- Seber – sebrání krabice na pozici robota; robot nemůže sebrat další krabici, pokud už nějakou krabici nese
- Polož – položení nesené krabice na pozici robota

Předpokládejme, že počáteční stav světa je jako na obrázku níže. Robot je značen písmenem R, dvě krabice jsou značeny písmeny A a B. Tmavá políčka označují zdi, světlá políčka jsou volná místa. Cílem je dostat krabici A na políčko označené hvězdičkou. Můžete dále předpokládat, že stav je popsaný pomocí predikátů `soused`(X, Y) vyjadřující, že X a Y jsou sousedi a na Y není zeď (tedy na Y je možné se z X přesunout), `na`(X, K) vyjadřující, že na pozici X je krabice K, a `robot`(X) vyjadřující, že robot je na políčku X. Značky v pravém horním rohu každého políčka uvádějí jeho jméno, které můžete použít, pokud se na konkrétní políčko potřebujete odkázat v řešení.



1. Formalizujte zadaný problém jako plánovací problém. Pro definování akcí a počátečního a cílového stavu použijte predikáty popsané výše, případně další predikáty, které si nadefinujete. Predikáty, které se nemění (`soused`), nemusíte explicitně vypisovat v definici počátečního stavu.

   Nápověda: Budete potřebovat definovat dvě verze akce pro pohyb - podle toho, jestli robot nese nějakou krabici nebo ne.

2. Popište, jak funguje dopředné a zpětné plánování.
3. Napište nějaký plán, který zadaný problém řeší.

6 Učení pomocí kolektivních prediktorů (otázka studijního zaměření – 3 body)

1. Vysvětlete princip trénování metodami *bagging* a *boosting*.
2. Popište trénovací algoritmus *nahodilých lesů*.
3. Uveďte konkrétní příklad algoritmu typu *boosting*. Stručně vysvětlete hlavní myšlenku.

7 Evaluace binárního klasifikátoru (otázka studijního zaměření – 3 body)

Předpokládejme, že binární klasifikátor vykazuje sensitivitu 0.8 a specificitu 0.9. Při jeho testování bylo použito 10000 příkladů, z nichž 500 bylo positivních.

Poznámka: Sensitivita bývá též nazývána *recall* nebo *true positive rate* a specificita se někdy nazývá též *selectivity* neboli *true negative rate*.

1. Na základě uvedených údajů spočítejte celou matici konfúze.
2. Uvažujme nahodile zvolený testovací příklad, který je predikován jako positivní. Určete pravděpodobnost, že tento příklad je skutečně positivní.
3. Jaká je přesnost (*accuracy*) daného klasifikátoru?

8 Model šumového kanálu v NLP (otázka studijního zaměření – 3 body)

Popište model šumového kanálu a ukažte, jak je v něm využita Bayesova věta. Vysvětlete, proč je aplikace Bayesovy věty výhodná (nápověda: argmax). Ilustrujte využití modelu šumového kanálu na alespoň dvou aplikacích NLP.

9 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)?

10 Homogenní souřadnice a maticové transformace (otázka studijního zaměření – 3 body)

1. Jak se pomocí matic transformují objekty v 3D počítačové grafice?
2. Proč zavádíme homogenní souřadnice a matice 4×4? Co nám umožňují realizovat?
3. Uveďte několik elementárních maticových transformací (translace, rotace, škálování – pokuste se matice napsat prvek po prvku) a jeden praktický příklad složené transformace (nemusíte konstrukci dotáhnout do konce, stačí naznačit postup).

11 Architektura programovatelných GPU (otázka studijního zaměření – 3 body)

Půjde nám o popis architektury moderních programovatelných grafických karet (GPU), budeme na ně nahlížet pouze z hlediska realtime zobrazování 3D grafiky.

1. Popište, z jakých základních částí se GPU skládá. Uveďte je v pořadí, ve kterém se účastní zpracování vykreslovaných 3D dat (tzv. zobrazovací řetězec nebo "pipeline")
2. Které části zobrazovacího řetězce můžete jako autoři aplikace programovat? Co jsou to "konstanty" ("uniforms") a data jakého typu se do nich ukládají?
3. Popište podrobněji poslední (z hlediska umístění v řetězci) programovatelný modul. Co má za úkol spočítat, co má na vstupu a co na výstupu.

12 Animační křivky (otázka studijního zaměření – 3 body)

Pro plynulé pohyby, animace a přechodové scény ("cut-scenes") je potřeba používat animační křivky. Budeme se zabývat křivkami, které se dají použít například pro animaci posunutí (translace) ve 3D scéně.

1. Navrhněte typ křivky, který byste dali k dispozici animátorům, aby s jejich pomocí mohli snadno zadávat spojitý translační pohyb ve scéně. Jaký bude princip zadání trajektorie (můžete uvést i přesnější matematické vzorce navrhované křivky)? Co přesně bude zadávat animátor(ka)?
2. Jakým způsobem půjde dosáhnout maximálně hladkého průběhu animace?
3. Jak by se dal naopak realizovat ostrý zlom v animační křivce (představte si prudký náraz nebo odražení)?

13 Homogenní souřadnice a maticové transformace (otázka studijního zaměření – 3 body)

1. Jak se pomocí matic transformují objekty v 3D počítačové grafice?
2. Proč zavádíme homogenní souřadnice a matice 4×4? Co nám umožňují realizovat?
3. Uveďte několik elementárních maticových transformací (translace, rotace, škálování – pokuste se matice napsat prvek po prvku) a jeden praktický příklad složené transformace (nemusíte konstrukci dotáhnout do konce, stačí naznačit postup).

14 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)?

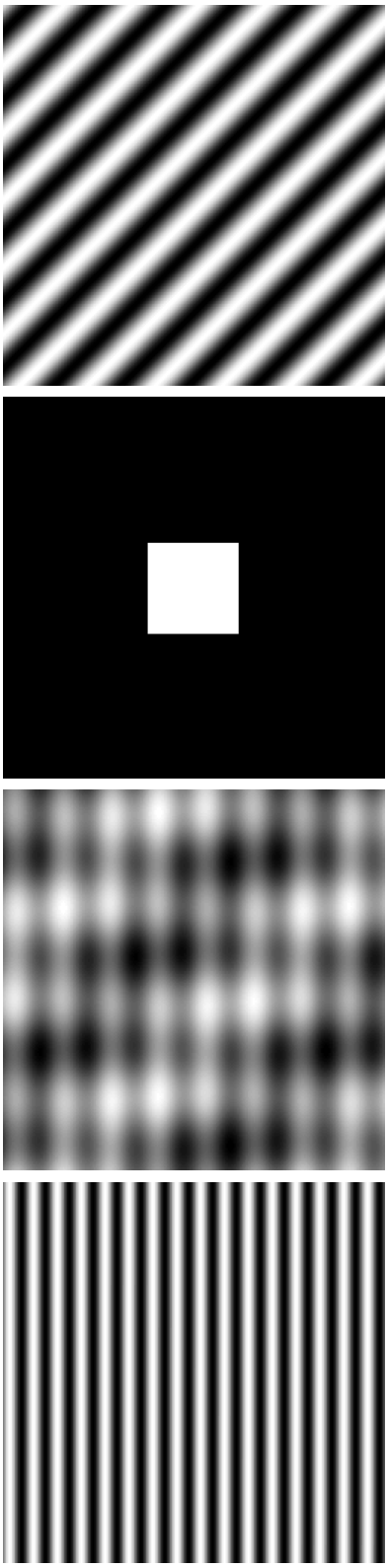
15 Fourierova transformace (otázka studijního zaměření – 3 body)

1. Definujte (napište vzorec a popište veličiny) 2D Diskrétní Fourierovu transformaci (DFT).
2. Jak se používá DFT na odstranění šumu v obrazech?
3. U obrázků na následující straně správně přiřaďte obraz a jeho DFT.

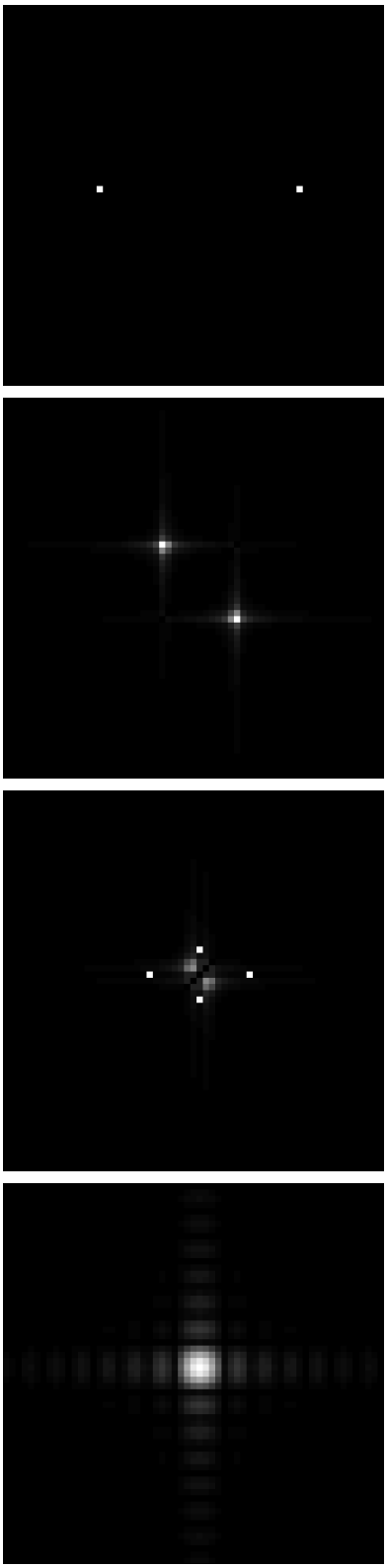
16 Naivní Bayesovský klasifikátor (otázka studijního zaměření – 3 body)

1. Popište princip naivního Bayesovského klasifikátoru. Na základě čeho (vzorec) přiřazuje objekty do tříd?
2. Popište význam jednotlivých pravděpodobností v Bayesově vzorci. V čem spočívá naivnost klasifikátoru?
3. Použitím Bayesovského klasifikátoru určete, zda student půjde do školy, jestliže vstane pozdě, nemá oblíbenou prednášku a prší. Pravděpodobnost určete na základě následujících pozorování.

Den	Vstanu	Je oblíbená přednáška ?	Počasí	Jdu do školy
1	Brzo	Ne	Slunce	Ne
2	Normálně	Ne	Déšť	Ne
3	Brzo	Ne	Slunce	Ano
4	Pozdě	Ne	Slunce	Ano
5	Pozdě	Ano	Slunce	Ano
6	Pozdě	Ano	Déšť	Ne
7	Brzo	Ano	Déšť	Ano
8	Normálně	Ne	Slunce	Ne
9	Normálně	Ano	Slunce	Ano
10	Pozdě	Ano	Slunce	Ano
11	Normálně	Ano	Déšť	Ano
12	Brzo	Ne	Déšť	Ano
13	Brzo	Ano	Slunce	Ano



Obrázek 1: Originální obraz



Obrázek 2: DFT zoom