Podzimní soustředění KSP 2022 – Seznam přednášek

Tento spisek jest nabídkou přednášek, které byste na soustředění mohli slyšet, čili jakási obdoba matfyzácké Karolínky (ta je ale, pravda, ještě stále o něco tlustší). Přednášek je daleko víc, než kolik se dá za pár dní stihnout, a tak je na vás, abyste si vybrali, o které máte opravdu zájem. Pokud byste rádi slyšeli ještě o něčem dalším, klidně si o to napište (např. na Discord), třeba se najde někdo, kdo by vám o tom rád pověděl. Berte a vychutnávejte!

Údaje o jedné přednášce vypadají asi takto:

Stručný úvod do základů teorie vlkodlaků *("Za dne ukryt v hloubi lesa, děs temný zvečera se plazí...")* **LYK**
RNDr. Á. Cula
Úvod do moderní teorie vlkodlaků, čili též praktická dæmonologie a naiadologie.
Předpoklady: Měsíc v úplňku.

Dozvíte se (čteno v obvyklém pořadí): jméno přednášky, v uvozovkách motto přednášky, kód (pro snadnější odkazování na konkrétní předměty), jméno přednášejícího a nakonec stručný obsah přednášky. Hvězdičky znamenají obtížnost.

Základní přednášky

V této kategorii sídlí přednášky, které se dají považovat za základní stavební kameny informatiky, ať teoretické, či praktické.

Algoritmy a datové struktury

Základní algoritmy a jejich složitost *("Čím menší je časová složitost algoritmu, tím větší je složitost kódu.")* **ZAKL**
Pravděpodobně dvoudílná přednáška pro ty, kdo potřebují dohnat základní znalosti nutné pro ostatní přednášky. Zadefinujeme si základní pojmy jako je algoritmus, program, rekurze a jak se počítá jejich časová složitost, bude následovat přehled základních algoritmů – převážně třídění, rychlé hledání k-tého nejmenšího prvku, práce s výrazy a další.

Grafy & algoritmy *("Pojďme si hrát s obrázky.")* **GA**
Lucka Vomelová, Jirka Setnička, Kiki Prokopová
Co to jsou grafy, jak je v programech reprezentovat a hlavně k čemu se dají použít. Prohledávání grafu do šířky i do hloubky. Hledání nejkratších cest: Dijkstrův a Floydův algoritmus. Minimální kostry a Union-Find problem.

Těžké problémy * **HARD**
Martin Koreček, Jirka Kalvoda, Martin „Medvěd" Mareš
V rámci této přednášky se budeme zabývat problémy tak těžkými, že nikdo na světě pro ně neumí vymyslet efektivní (rozuměj polynomiální) algoritmus. Spousta lidí dokonce věří, že to vůbec možné není. Abychom mezi tyto problémy pronikli, seznámíme se s pojmy NP-úplnosti a NP-těžkosti. Především si však konkrétní těžké úlohy ukážeme a naučíme se i některé těžké úlohy rozpoznat. Závěrem si řekneme, jak se s těžkými úlohami vypořádat v praxi.

Toky v sítích *("Když je v grafu povodeň, těsní?")* **TOKY**
Jirka Kalvoda, Michal Kodad, Jirka Setnička, Martin „Medvěd" Mareš
K čemu je dobré, když grafem teče voda. Předvedeme si klasický problém toků v sítích a jeho všelijaké, mnohdy dosti překvapivé aplikace. Jak rozestavět n věží na šachovnici a jak ji místo toho pokrýt dominovými kostkami? Další souvislosti, jako třeba násobná souvislost grafů.
Předpoklady: Umět plavat (zejména v matematice)

Datové struktury pro začátečníky *("Pole oraná a neoraná, stromy ovocné a okrasné.")* **DS1**
Lucka Vomelová, Michal Kodad, Kiki Prokopová
Jak si ukládat data natolik šikovně, abychom je nejen neztratili, ale také našli dříve, než si pro nás přijde Smrť. Klasické struktury jako pole, seznamy, fronta a zásobník, trie, vyhledávací stromy (vyvážené, AVL, a-b, splay), haldy (binární a obecně regulární) a v neposlední řadě hešování.

Datové struktury pro pokročilé * *("Haldy a jiné kupky.")* **DS2**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Důmyslnější varianty vyhledávacích stromů: splay stromy, BB-α stromy, rankové stromy, vícerozměrné stromy. Chytřejší haldy: binomiální, Fibonacciho, rank-pairing. Amortizovaná analýza složitosti. Též několik přátelských randomizovaných datových struktur: skip listy a treapy.

Intervalové stromy * *("Já bych ty intervaly nejradši... dal do stromu!")* **ITREE**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Intervalový strom je datová struktura pracující s intervaly, se kterou se můžeme setkat v mnoha úlohách (zejména soutěžních). Řekneme si, co to intervalový strom je, jaké všechny druhy intervalových stromů existují a jejich použití si ukážeme na úlohách. Na závěr si představíme jednu „magickou" datovou strukturu jménem Fenwickův strom.

Dynamické programování *("Kampak jsem si to jenom schoval?")* **DYNP**
David Klement, Jirka Kalvoda, Lucka Vomelová, Kiki Prokopová
Dynamické programování je programátorská technika využívající velice prostinkého nápadu: Proč něco počítat několikrát, když to mohu spočítat jednou a výsledek si uložit? Na této přednášce si ukážeme, že tento jednoduchý nápad může pomoci efektivně vyřešit i poměrně obtížné úlohy.

Hledání v textu *("»Vyšíváme v seníku!« – kde jsem to jen viděl?")* **TEXT**
Honza Černý, David Klement, Lucka Vomelová, Michal Kodad, Kiki Prokopová
Někdy potřebujeme najít podřetězec ve velkém množství textu. Stromeček trochu připomínající ten biologický aneb trie. Proč se ve vstupu vracet neboli Knuthův-Morrisův-Prattův algoritmus. Hledání více řetězců najednou podle Aha a Corasickové. Okénkové hešování Rabina a Karpa.

Geometrie a počítače *("Nerušte mé kruhy! (ani jiné kvadriky)")* **GEOM**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Základní algoritmy pro řešení geometrických úloh – konvexní obal, dva nejbližší body v rovině, výpočet obsahu nekonvexního mnohoúhelníka, lokalizace bodu, scanline algoritmus a jeho použití, Voroného diagramy a souvislost s persistentními datovými strukturami.

Amortizace *("Celek bývá daleko menší než součet částí.")* **AMORT**
Martin Koreček, Jirka Kalvoda, Martin „Medvěd" Mareš
Spousta algoritmů je mnohem rychlejší, než jak na první pohled vypadají. Šikovný způsob, jak takové chování zkoumat, je amortizovaná časová složitost. Předvedeme několik trochu překvapivých příkladů amortizace: dvojková a jiná počitadla, datové struktury založené na přebudovávání, vyhledávací stromy bez otravného vyvažování, dynamizace datových struktur, udržování historie.

Programovací jazyky a nástroje

Programování v jazyce C **C**
Vojta Káně, Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Jazyk C patří k nejrozšířenějším jazykům, hodí se pro low-level programování i kusy kódu, které mají zejména být rychlé. Představíme si datové typy a běžné programové konstrukce, vysvětlíme si základy práce s ukazateli a také se seznámíme se standardními knihovnami jazyka C.

Exkurze jazykem C++ **CPP**
Michal Kodad
Základní exkurze do jazyka C++ - jazyka, po kterém sáhnete, když skutečně potřebujete výkon. Dává totiž daleko větší kontrolu nad tím, co se kdy bude v systému dít a jak. Není to jazyk pro pohodlné programátory, o to více se jeho znalost cení. Vhodné pro kohokoli.

Objektově orientované programování *("Object-oriented system. If we change it, users object.")* **OOP**
Honza Černý, Vojta Káně, Michal Kodad
Objektově orientované programování přináší jiný náhled na návrh řešení problémů. Vysvětlíme, jak se liší objektové a procedurální programování. Co je to objekt a co třída. Základní vlastnosti objektů (dědičnost, zabalení, polymorfismus). Co je to metoda, překrývání metod, virtuální metody (pozdní vazba) a čistě virtuální (abstraktní) metody. Jak se liší OOP ve statických (C++, C#, Java) a dynamických (Python) jazycích. Jak programovat objektově i bez podpory jazyka, třeba v Céčku.
Předpoklady: Znalosti procedurálního programování, například v Pascalu, v Pythonu nebo v C.

Programování v jazyce Java **JAVA**
Honza Černý
Java je jeden z nejrozšířenějších objektových programovacích jazyků za posledních deset let. Na přednášce se seznámíme s jeho myšlenkou a naučíme základy. Přednáška je dělaná pro posluchače, kteří umí alespoň základy jiného programovacího jazyka.

Programování v jazyce C# *("Co se stane, když strčíme Céčko za mříže?")* **CIS**
Michal Kodad
C# je moderní objektově orientovaný jazyk, jehož tvůrci se inspirovali přednostmi a úskalími ostatních programovacích jazyků, zejména Javy. Je jednoduchý a crossplatformní (tedy snadno v něm vytvoříte i okýnka, která nepoběží jen na okýnkách). Naučíme se základy a možná si i napíšeme jednoduchý prográmek.
Předpoklady: Tušit něco málo o objektovém programování.

Python *("`print("Ffff".decode("rot13"))`")* **PYTH**
Jirka Kalvoda & Michal Kodad
Jak programovat v Pythonu a jak v něm „nepsat Cčko". Syntaxe, datové typy, funkce, třídy, ... Na co si dát pozor, v čem se Python liší od ostatních jazyků a proč je mezi nimi tak oblíbený.

Procesy a vlákna * *("Koupil jsem dalších 15 procesorů, proč je to stále stejně pomalé?")* **THREAD**
David Klement, Jirka Kalvoda
Procesory už zrychlovat moc neumíme, přidávat vlákna však ano. Rychlý přehled dnešních procesorů a co to znamená pro programátora. Jak psát programy pro více vláken. Kdy to jde jednoduše a kdy je potřeba synchronizace.

Logické programování *("Mohu být svým vlastním dědečkem?")* **LOGP**
Honza Černý, Pali Rohár, Jirka Kalvoda
Což takhle projednou neříkat počítači, jak má věci počítat, ale jenom mu zadat podmínky, které má výsledek splňovat? Neprocedurální programování vychází přesně z této myšlenky. Podíváme se na programovací jazyk Prolog, který vychází z formální logiky. Zjistíme, které problémy se v něm neobyčejně zjednoduší a které naopak programování promění v noční můru. Pokud jsi milovník rekurze, budeš u této přednášky nejspíš skoro celou dobu spokojeně vrnět.

Haskell *("V téhle proměnné je uložen okolní svět.")* **HASK**
Vojta Káně, Jirka Kalvoda
Základní kurz Haskellu – moderního funkcionálního jazyka. Zkusíme se na chvíli k funkcím programu chovat jako k těm matematickým a uvidíme, že zákaz side-efektů a globálních proměnných může vést k přehlednějšímu a spolehlivějšímu kódu. Přesvědčíme se, že náš program často umíme poskládat ze spousty malých, ale šikovných funkcí. Ukážeme si syntaxi, vysvětlíme typovou kontrolu a typový systém. Rekurze, aneb seznam je tak dlouhý, jako seznam bez prvního prvku plus jedna. Přičichneme k třídám, zrušíme výjimky a zavedeme zcela bezpečná vlákna. Řekneme si, proč v Haskellu nejde komunikovat s okolním světem a proč nám pomůže si okolní svět uložit do proměnné. A že vlastně v Haskellu žádné proměnné nejsou, jen visačky na datech.
Předpoklady: Sklony k algebraickému chápání vesmíru, odvahu tváří v tvář své vlastní tváři a rekurzi.

Programování webových aplikací (nejen) v Pythonu *("Vlastní blog či wiki za 20 minut?")* **WEBAPP**
Jirka Kalvoda
Jak jednoduše vytvářet webové aplikace v Pythonu pomocí webových (mikro)frameworků (Flask, Django, ...): zpracování požadavků a dat z formulářů, generování HTML, práce s databází. Co nejde napsat na pár řádek, nás nezajímá. Polopraktická přednáška o tom, jak si pořídit webovou aplikaci a nestrávit u toho léta.

SQL databáze *("SELECT something FROM knowledge LIMIT 90min")* **SQL**
Vojta Káně, Martin „Medvěd" Mareš
Jak si schovat data do relační databáze a jak je tam zase najít, ideálně rychle. Definice tabulek a indexů. Dotazy a jejich skládání a vnořování. Pohledy, funkce a triggery. Transakce a různé druhy konzistence. Rozdíly mezi dialekty SQL.

Hardware a operační systémy

Principy počítačů *("A opravdu uvnitř počítače běhají malí trpaslíci?")* **HW**
Vojta Káně, Pali Rohár, Jirka Kalvoda, Jirka Setnička
Vydáme se do země skřítků, kteří pohánějí počítače. Počítačové architektury od hodinek po superpočítač od Craye, jejich křivolaká historie i současnost. Co je to procesor, jak se programuje a jak se chová. Různé druhy pamětí a jejich cacheování. Jak procesory komunikují s okolím – sběrnice, čipové sady, vstupní a výstupní zařízení. A co když je procesorů několik, nebo třeba pár tisíc? Přednáška bude praktická: pár počítačů při ní rozebereme a možná i nějaký postavíme.

Od zdrojáku k programu *("Před spuštěním program přeložte. Stačí třikrát podélně?")* **KOMP**
Pali Rohár, Martin „Medvěd" Mareš
Mezi programem, který jste právě dopsali, a tranzistory uvnitř vašeho procesoru leží obrovské území obývané překladači, linkery, knihovníky, operačními systémy, loadery a jinými bájnými bytostmi. Pojďme zjistit, co jsou zač a co všechno s programem provádějí. Co udělá kompilátor za nás a co musíme naopak udělat my za něj.

UNIX *("UNIX gives you enough rope to hang yourself.")* **UNIX**
Jirka Setnička
Unixové operační systémy (zejména Linux) dobývají svět. Jak fungují uvnitř a jaké nabízejí výhody? Unixová filosofie a historie. Proč je systém složený ze spousty malých a jednoduchých kousků stabilnější a bezpečnější? Proč ovládání prostřednictvím textových příkazů je často efektivnější než klikátka? Jaké to je mít svůj systém pod kontrolou a „vidět mu pod ruce"? V čem spočívá moc textových souborů?

Operační systémy *("Mám 3GHz procesor, tak co to už půl hodiny dělá?")* **OS**
Vojta Káně
Jak vypadá architektura dnešních operačních systémů aneb co všechno musí systém zařídit, aby na něm programy fungovaly. Správa procesů a vláken, plánování, synchronizace. Paměť, adresace a její přidělování. Správa souborů, filesystémy. Čemu se říká jádro a proč se spojuje s pudlem.

Programování v Linuxu *("Všechno na světě je tak trochu soubor.")* **PLX**
Pali Rohár, Martin „Medvěd" Mareš
Jak vypadá rozhraní mezi jádrem Linux a uživatelskými programy. Co se doopravdy stane, pokud ve svém céčkovém programu zavoláme `printf` nebo `malloc`. Jak napsat program, který vůbec nepotřebuje standardní céčkovou knihovnu. Co všechno se umí chovat jako soubor a co jako signál.
Předpoklady: Schopnost přečíst a napsat jednoduchý program v C.

Sítě a bezpečnost

Sítě a Internet *("Sítě nejen na ryby.")* **NET**

Vojta Káně, Jirka Setnička, Martin „Medvěd" Mareš

Jak funguje Internet a počítačové sítě vůbec: od elektronů v drátech (fotonů v optických kabelech nebo elektromagnetických vln) přes packety a jejich forwarding až k jednotlivým síťovým službám. Adresace, internetworking a dynamický routing. Jak NAT zachránil i zničil Internet a proč se těšíme na IPv6.

Sítě II – protokoly a síťové útoky *("Jak si přečíst maily... sousedovy maily.")* **NET2**

Vojta Káně, Jirka Setnička, Martin „Medvěd" Mareš

Volné navázání na NET. Budeme si povídat o tom, co za data nám po síti běhá a jaké se k tomu používají protokoly – DNS, FTP, HTTP nebo třeba i mailové SMTP a IMAP. Zaměříme se více na ty nejpoužívanější (metody GET a POST v HTTP), nakousneme cacheování a nadlábneme se cookies. A pokud zbude čas, využijeme zranitelnosti některých protokolů a provedeme síťový útok.

Předpoklady: Základní povědomí o počítačových sítích

Webové stránky **WWW**

Robert Gemrot

Co se děje za oponou, když do prohlížeče zadáte adresu svých oblíbených stránek? A jak si takovou stránku taky pořídit? Přelet nad protokolem HTTP, seznámení s HTML a předvedení kaskádových stylů. Jak fungují dynamické stránky od formulářů až po JavaScript běžící v prohlížeči.

Kryptografie *("Gbgb arav zbp gnwan mcenin.")* **CRYPT**

Martin „Medvěd" Mareš

Kryptografie čili tajuplná nauka o šifrách, jejich konstrukci a hlavně o jejich luštění. Šifrovací systémy jako lego: základními kostičkami nám budou symetrické a asymetrické šifry, jednosměrné funkce a náhodné generátory. Stavět z nich budeme kryptografické protokoly na bezpečný přenos, autentikaci, digitální podpisy a třeba i na házení korunou po telefonu. Předvedeme nerozluštitelnou šifru a dokonce to o ní i dokážeme.

Teoretická informatika

Složitější složitost * **SLOZ2**

Jirka Kalvoda, Martin „Medvěd" Mareš

Trochu hlouběji o složitosti. Přesná definice výpočetního modelu a velikosti vstupu. Složitost v nejlepším, nejhorším a průměrném případě; amortizovaná analýza. Jak dokázat, že úlohu nejde řešit rychleji, aneb dolní odhady. Porovnávání problémů pomocí redukcí, problémy NP-úplné a ještě těžší.

Předpoklady: ZAKL

Strojové učení *("Nechme stroje se samy učit.")* **ML**

Michal Kodad

Co je to strojové učení? Jaké typy strojového učení existují? Začneme u jednoduché lineární regrese, přes perceptron až skončíme u kouzelného slovíčka neuronové sítě. Povíme si rozdílné druhy neuronových sítí a nakonec si odskočíme k algoritmu, který nepotřebuje kromě surových dat nic navíc a dokáže dělat užitečné věci.

Umělá inteligence * **AI**

Honza Černý & Michal Kodad

Ukážeme si, jak počítače přemýšlí při řešení problémů a jakým způsobem hledají řešení. Volně se dostaneme k prohledávání stavového prostoru (který bývá exponenciálně velký) a ukážeme si různé jak informované, tak neinformované techniky pro jeho procházení. Setkáme se třeba s algoritmy, které jsou použity v GPS.

Evoluční algoritmy * *("Já to dělat nebudu, ať to za mě udělají mravenci!")* **EVA**

Honza Černý, Jirka Setnička

Evoluční algoritmy se inspirují strukturami chování v přírodě a na jejich základě pak (optimalizačně) hledají řešení těžkých problémů. Na přednášce určitě zazní genetický algoritmus, zmíníme jeho algoritmy a když zbyde čas tak si obecněji popovídáme o algoritmech pohybujících se ve velkých prostorech řešení.

Modely počítačů *("Nač Pentium? Máme Turingovy stroje!")* **MODEL**

Jirka Kalvoda, Martin „Medvěd" Mareš

V HW se dozvíte, jak fungují „opravdové" počítače, zde pro změnu na čem počítají teoretici. Všechny počítače jsou si rovny, jen některé jsou si rovnější. Turingův stroj obyčejný, vícepáskový, nedeterministický a univerzální. Random Access Machine (RAM) a Pointer Machine. Trocha minimalismu aneb stroj s počítadly. Až nám začne být smutno, pořídíme si klidně N^2 procesorů a spřáhneme je do paralelního počítače (PRAM). Rychlé paralelní slévání a třídění. Pokud zbude čas, ukážeme si buněčné a grafové automaty, nebo třeba dlaždičky v koupelně.

Jazyky, gramatiky a automaty * **AUTO**

Honza Černý, Jirka Kalvoda, Martin „Medvěd" Mareš

O jazycích přirozených, počítačových a matematických, jejich popisu a rozpoznávání. Začneme těmi nejjednoduššími: regulární jazyky a výrazy, konečné deterministické a nedeterministické automaty. Pak budeme stoupat po příčkách Chomského hierarchie, kam až to půjde. Jak výpočetně silný je třeba takový automat na kafe?

Matematické přednášky

Grafy bez algoritmů **GRAFY**

Honza Černý & Jirka Kalvoda

Teorie grafů trochu teoretičtěji. Různé druhy grafů a jejich vlastnosti. Stromy a lesy. Kreslení grafů jedním tahem. Princip sudosti a skóre grafu. Jaké speciální vlastnosti mají rovinné grafy a jak je lze obarvit šesti nebo možná i pěti barvami. Jak poznat, že dva grafy (ne)jsou isomorfní. Mosty, artikulace a ušaté lemma. Párování, střídavé cesty a Hallova věta.

Úvod do teorie čísel **NUT**

Martin „Medvěd“ Mareš

Co a k čemu je teorie čísel. Počítání v kongruenci, Euklidův algoritmus a jeho použití. Konečná tělesa a Malá Fermatova věta. Prvočísla a Eratosthenovo síto. Čínská zbytková věta a její algoritmická verze. Jak si odvodit kritéria dělitelnosti.

Kombinatorika *(“Nemám rád faktoriály. Faktoriály nemám rád. Rád nemám faktoriály. . . ”)* **KOMB**

Jirka Kalvoda, Martin „Medvěd“ Mareš

Při navrhování algoritmů a počítání jejich složitosti narazíme na celou řádku zajímavých a ne úplně triviálních kombinatorických problémů, a tak se naučíme, jak na ně. Základní triky s faktoriály a kombinačními čísly, sčítání konečných a občas i nekonečných řad, rekurentní rovnice a princip inkluze a exkluze. Možná se také potkáme s Dlouhým, Širokým a poněkud zmatenou šatnářkou.

Jak vypadá zrcadlo v číslech **LIN1**

Honza Černý, David Klement, Martin „Medvěd“ Mareš, Michal Kodad, Kiki Prokopová

Na přednášce se podíváme na to, jaká matematika stojí za grafickými transformacemi obrázků. Ukážeme si, jak můžeme otáčet předmět pomocí jeho zobrazení osovou souměrností, a budeme si hrát s dalšími transformacemi. Dozvíte se, jak vypadají matice a jak je násobit s vektory. Povíme si, co všechno stojí na lineární algebře a jaké problémy dokáže vyřešit.

Lineární algebra * **LIN2**

Honza Černý, David Klement, Martin „Medvěd“ Mareš, Michal Kodad, Kiki Prokopová

Naučíme se hledat řešení soustav rovnic na papíře tak, abychom toho museli napsat co nejméně (Gaussova eliminace). Zabředneme hlouběji do vektorových prostorů a budeme hledat jejich hezké ortogonální a ortonormální báze.
Předpoklady: Je dobré vědět, co je to matice a jak se násobí (přednáška LIN1).

Rozšiřující přednášky

Mezi rozšiřujícími přednáškami se dají nalézt různé specifičtější obory a zájmy, jakožto i těžší přednášky navazující na předchozí díly ze základních přednášek. Mezi nabízenými přednáškami si tak můžete vybrat obor svého zájmu a tomu se dál věnovat.

Algoritmy a datové struktury

Nejkratší a jiné cesty * *("Všechny cesty vedou do Horní Dolní, jen některé přes Řím.")* **CESTY**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
O problému hledání cest v grafech trochu podrobněji. Obecné relaxační schéma, Bellmanův-Fordův a Dijkstrův algoritmus a jejich zrychlení pomocí různých datových struktur. Potenciálová redukce a heuristiky (třeba A^*), zaokrouhlování délek hran. Souvislosti s násobením matic: transitivní uzávěr, Seidelův algoritmus, Kleeneho algoritmus a regulární výrazy.

Datové struktury pro ještě pokročilejší ** *("log log log log . . . glo glo glo . . . ")* **DS3**
Jirka Kalvoda, Martin „Medvěd" Mareš
Na přednášce si ukážeme některou z méně známých složitějších datových struktur. Pokud Ti ostatní přednášky přijdou moc jednoduché, tato je ta pravá pro Tebe.

Stringové algoritmy ** *("Jak obrátit řetězec naruby?")* **STRG**
Jirka Kalvoda, Martin „Medvěd" Mareš
Ukážeme, jak spoustu zajímavých problémů týkajících se řetězců vyřešit v lineárním čase. Bude se nám k tomu hodit datová struktura jménem suffixový strom. Podíváme se, jak strom vypadá, k čemu se hodí a jak ho sestrojit. Též prozkoumáme několik příbuzných zvířátek, jako třeba suffixové pole a suffixový automat.

Stromové algoritmy * *("Půjdeme na to od lesa.")* **TREES**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Stromy jsou jednou z nejtypičtějších (a nejjednodušších) odrůd grafů. Ledacos pro ně umíme řešit mnohem rychleji než pro obecné grafy, tak se pojďme podívat, jak se to dělá. Předvedeme několik obecných technik pro práci se stromy: DFS očíslování, „vandalskou indukci", intervalové reprezentace. Různé rozklady: heavy-light, Fredericksonův, centroidový a ST-stromy.

Magické algoritmy * *("Pokročilá magie není rozlišitelná od technologie.")* **MAGIC**
Jirka Kalvoda, Martin „Medvěd" Mareš
O algoritmech značně magických a nečekaných. Jak násobit n-ciferná čísla rychleji než v kvadratickém čase. Kouzlo na slévání setříděných posloupností v konstantním prostoru. Isomorfismus stromů pomocí přihrádkového třídění. Bitové kejklířství. Hledání největší díry.

Toky v sítích pro pokročilé * *("Když Edmons-Karp nestačí.")* **TOKY2**
Jirka Kalvoda, Jirka Setnička
Předvedeme si několik rychlejších algoritmů pro problém maximálního toku. Dinicův algoritmus a jeho mnohá vylepšení. Různá zobecnění maximálního toku: assignment problem, aneb hledání nejlevnějšího bipartitního párování. Maximální tok minimální ceny, aneb co když za průtok trubkami musíme platit? Ukážeme si algoritmus založený na postupném vylepšování a předvedeme si na něm obecnou myšlenku, kterou můžeme použít u optimalizačních problémů. Pokud zbyde čas, řekneme si, co se změní, když budeme hledat maximální tok, který jde rozložit do „krátkých" cest (problém L-omezeného toku), nebo když budeme chtít vedle ropy v jedné síti zároveň přepravovat i čaj a Kofolu (problém multikomoditního toku).
Předpoklady: TOKY

Splay stromy *("Lepší než uklízení je organizovaný chaos.")* **SPLAY**
Jirka Kalvoda, Martin „Medvěd" Mareš
Zapomeňte na pracné vyvažování vyhledávacích stromů. Místo toho zavedeme triviální pravidlo: pokaždé, když pracujeme s nějakým prvkem, vytáhneme ho do kořene stromu. Ukážeme, že toto pravidlo stačí na dosažení logaritmické složitosti, tedy aspoň amortizovaně. Také dokážeme, že Splay strom je nejhůře konstanta-krát horší než libovolný jiný strom, a možná i spousta dalších magických vlastností.

Programovací jazyky

C for wizards * *("`1[x]+++++x[1]`")* **CWIZ**
Jirka Kalvoda, Martin „Medvěd" Mareš
Ponořme se do hlubin Céčka, snad až na samé dno. Typový systém: elementární typy, typové výrazy, automatické konverze a rozpad typů (pole vs. ukazatel). Pořadí vyhodnocování kontra pořadí side-efektů (priority, synchronizační body a volatile). Triky s preprocesorem. Návěští a příkaz switch. Všelijaké zrady (velikosti typů, zarovnání, $(a+b)+c \neq a+(b+c)$, . . .). Dialekty Céčka od K&R až po nejnovější normu C11 a různá nestandardní rozšíření jazyka. Proč jsou objekty potřebnější v mysli programátorově než v jazyce a proč je C lepší než C++ ☺
Předpoklady: Povšechná znalost jazyka C.

Programování v assembleru **PASM**

Martin „Medvěd“ Mareš

Jak programovat procesor přímo, aniž by vám do toho mluvily překladače, linkery a podobná verbež. Začneme obecně, ale soustředíme se hlavně na procesory rodiny x86. 32-bitová a 64-bitová instrukční sada, FPU a panoptikum vektorových instrukcí. Rozdíly mezi intelovskou a AT&T syntaxí. Jak spojit assembler s vyššími programovacími jazyky. Optimalizace kódu. Stručný úvod do systémových architektur IA32 a AMD64.

Jazyk Go **GOLANG**

Vojta Káně, Jirka Setnička

Go je moderní kompilovaný jazyk (vyvinutý původně v Google), který se pokouší být takovým Cčkem na steroidech. Umí být podobně rychlý, zaručuje větší typovou bezpečnost, ale má i prvky dynamicky typovaných jazyků. Jeho velkou silou je velmi snadné provázání na existující kód v C/C++, systém balíčků distribuovaných převážně přes Github, funkce vracející libovolný počet hodnot nebo třeba vestavěná podpora Unicode a vestavěná hashovací tabulka. Také se silně dbá na coding-style pro zajištění snadné čitelnosti programů.

Jazyk Lua *(“Co není potřeba, to zahodíme.”)* **LUA**

Jirka Setnička, Martin „Medvěd“ Mareš

Lua je minimalistický jazyk, jehož interpret se vejde do jednoho megabajtu, takže je horkým kandidátem na skriptovací jazyk pro zařízení, která mají málo paměti. Zároveň je velmi často používaný třeba pro skriptování pluginů do různých programů, skriptování herní logiky nebo botů ve hrách nebo třeba ve své JIT verzi i ke zpracování requestů v populárním webovém serveru nginx (kde je při správném použití skoro stejně rychlý, jako nativní kód v C). Naučíme se základní principy jazyka, ukážeme si na jaké věci si v něm dát pozor, pokud nám jde o výkon, ale také to, jakým způsobem jej propojit s jinými jazyky, především s jazykem C.

Perl *(“Jak Pejsek a Kočička vymýšleli programovací jazyk.”)* **PERL**

Pali Rohár, Martin „Medvěd“ Mareš

Jednoho dne se Larry Wall rozhodl, že nasype do jednoho velkého kotle spousty programovacích jazyků a unixových utilit, za stálého míchání povaří, posléze přecedí, přikoření a implementuje. Tak vznikl Perl, jazyk původně určený hlavně na zpracování textu, ovšem jak se ukázalo, též šikovný na spoustu dalších věcí. Asociativní pole, libovolně složité datové struktury za pomoci referencí, balíčky a objekty zdarma a hlavně regulární výrazy zde a všude. Zkrátka jazyk, který lze jedině milovat nebo nenávidět, nic mezi tím. Co se Perl 5 přiučil od Perlu 6.

Raku alias Perl 6 *(“Slečno, mohu vám ukázat svou sbírku operátorů?”)* **RAKU**

Martin „Medvěd“ Mareš

Je to Perl, a přitom to Perl není. Co je to? Aneb jak to dopadne, když se pokusíme navrhnout programovací jazyk budoucnosti a inspirovat se přitom filosofií Perlu. Typový systém, pokud zrovna chcete. Objekty, třídy a metatřídy. Periodická soustava (meta)operátorů. Definování jazyka v sobě samém. A co se to stalo s regulárními výrazy? Jak vypadají implementace P6 a kdy je prozatím lepší programovat na papíře. Praktické cvičení ve stavbě vzdušných zámků a bydlení v nich.

Pokročilé povídání o Pythonu *(“`import antigravity`”)* **PYTH2**

Jirka Kalvoda, Michal Kodad

Povídání o méně zmiňovaných částech Pythonu. Dekorátory, metaclasses, generátory, funkcionální styl programování v Pythonu. Jak napsat quicksort jako lambda funkci. Představení zajímavých modulů nejen ze standardní knihovny. Další témata dle přání účastníků: paralelní programování (asyncio, multiprocessing), síťová komunikace, GUI, matematické výpočty, propojení Pythonu s C, ...

Předpoklady: PYTH

Jazyk Rust **RUST**

Vojta Káně

Rust je moderní programovací jazyk zaměřený na typovou a paměťovou bezpečnost, nekompromisní výkon, multithreading bez pastí. Zároveň přejímá nějaké vlastnosti funkcionálních jazyků se zachováním výborného výkonu, například iterátory jsou stejně rychlé jako for cykly. Ukážeme si základní principy jazyka. Jak se spravují objekty bez garbage collectoru a co nám to umožňuje (a co zakazuje).

Předpoklady: Schopnost programovat, tušení o nízkoúrovňových paměťových věcech.

Praktický úvod do Jekyllu **JEKYLL**

Honza Černý

Chtěl sis někdy vytvořit vlastní webovou stránku, ale ručně psát HTML a CSS tě otravuje? Zajímalo by tě, jak z Markdownu generovat pěkně vypadající, databázemi a jinou havětí nezatížené webové stránky? Ukážeme si, jak si pomocí Jekyllu vytvořit, spravovat a v neposlední řadě také publikovat webovou stránku.

Jazyková Zoo *(“Na co GO TO? Máme COME FROM.”)* **JZOO**

Martin „Medvěd“ Mareš

Obecná teorie programovacích jazyků má asi tolik půvabu, jako biologická systematika. Tak se raději pojďme podívat do zoo: poznejme jazyky klasické, experimentální i dočista absurdní. Ada, Céčko a Python (tři pohledy na fungování typů). Pradědeček všech funkcionálních jazyků LISP (program a data jsou totéž). APL (algebraické inspirace, nebo též průvan ve skladišti písmenek). Forth (zásobníkový předchůdce Postscriptu, ale i javovského virtuálního stroje). Lingua::Romana::Perligata (programovací jazyk, který skloňuje a časuje). Shakespeare, Intercal, Oook! a jiné komedie. Samorozšiřitelné a hybridní jazyky.

Nix(OS) *("Milujeme Haskell ale namísto toho sestavujeme balíčky")* **NIX**
Vojta Káně
Reprodukovatelná operace je taková, která vždycky dopadne stejně, ať už ji spouštíme v různých časových okamžicích, nebo na různých počítačích. Kompilování software se naopak vyznačuje zcela opačnými tendencemi – výsledek závisí na dostupných knihovnách, jejich verzích, stavu cache, někdy dokonce i stavu nějakého vzdáleného serveru. Nix s tím poměrně úspěšně bojuje. Je to ryze funkcionální programovací jazyk, kterým popíšeme, co bychom chtěli sestavit a přiložené nástroje se nám o to postarají. Jako velmi šílený (ale úspěšný) bonus si můžeme uvědomit, že operační systém se svou aktuální konfigurací a nainstalovanými programy je také v nějakém smyslu balíček, který můžeme popsat Nixem. Tím získáme deklarativní distribuci NixOS, jejíž taje také představím.

Programovací nástroje a techniky

Git a jiné systémy pro správu verzí *("U svatýho tučňáka, kdo sem napsal tohle? Ono to tvrdí, že JÁ?!")* **GIT**
Vojta Káně, Pali Rohár, Michal Kodad, Jirka Setnička
Jak vyvíjet program delší dobu a nezbláznit se u toho. Různé systémy pro správu verzí od diff/patch přes CVS a SVN až ke Gitu. Jak Git funguje: stromy, commity, větve, tagy. Merge mezi větvemi nebo mezi různými počítači.

Git pro pokročilé *("In case of fire, commit, push, and exit the building.")* **GIT2**
Vojta Káně, Pali Rohár, Jirka Setnička, Martin „Medvěd" Mareš
Používáte Git pro všechny své programy a k svačině místo novin čtete commit logy svých oblíbených projektů? V tom případě pojďme nahlédnout pod pokličku, jak Git funguje uvnitř. Reprezentace historie pomocí hešování grafů. Pracovní strom, index, commity a jejich adresy, větve. Pack files jako elegantní způsob komprese dat na disku i na síti. Kouzelnické triky: hledáme bugy půlením historie, přepisujeme dějiny, automaticky konvertujeme soubory. Git v praxi: jak se liší správa zdrojáků v projektech o jednom, deseti a tisíci programátorech. Udržujeme patche k cizímu programu aneb StGit.

Make *(" `make love ... don't know how to make love` ")* **MAKE**
Pali Rohár, Martin „Medvěd" Mareš
Hodil by se otrok, který by překládal jednotlivé soubory. Základní syntaxe takového otroka, jak napsat jednoduchý —Makefile—, který řeší překlad Céčkového programu, automatické řešení závislostí. Jak to udělat, aby výsledek neměl několik tisíc řádek. Proč by se hodilo, aby tu bylo něco lepšího, a proč tomu nepomáhají ani automake, cmake, qmake a další.

Gdb a jiné ladící nástroje * *("Jak se ladí kytara, jak křišťálová koule a jak program (řazeno dle obtížnosti).")* **GDB**
Pali Rohár, Martin „Medvěd" Mareš
Kdo píše programy, které vždy hned fungují, ať se přihlásí. A kdo ne, ať se přihlásí na tuto přednášku. Ukážeme si několik nástrojů, jak si pomoci z nejhoršího. Mezi nimi třeba gdb, řádkový debuger (odvšivovač), strace, nebo valgrind. Kdy je použít a kdy se více hodí `printf`. Proč `assert` je tak užitečná věc.

Textový editor Vim *("Víš, jaký je nejlepší textový editor? Vim.")* **VIM**
Honza Černý & Jirka Kalvoda, Martin „Medvěd" Mareš
Odložme na chvíli své myše a pojďme si vyzkoušet textový editor, který umí poslouchat na slovo. Pravda, budeme se ta slova muset chvíli učit, ale výsledek bude proklatě efektivní. Základní příkazy, práce s regulárními výrazy, makra, kouzla. Vimovité ovládání jiných programů, třeba webového prohlížeče.

Jak se nestat vepřem *("/* You are not expected to understand this */")* **STYLE**
Pali Rohár, Jirka Setnička, Martin „Medvěd" Mareš
Tvrdí se, že číst kód je mnohdy těžší, než ho psát – dokonce i po sobě, stačí krátká doba. Je několik obecně uznávaných pravidel, jak kód psát a jak ne, aby byl hezký a dobře čitelný. Od základních (rozumná pojmenovací konvence, systematické odsazování), až po to, kdy opravdu použít `goto`, jak členit program na funkce a jak využít nějaké třídy, moduly a podobně. Jak napsat užitečný komentář nebo dokumentaci. A kdy se vyplatí se na všechna tato pravidla vybodnout.

Testování a kvalita softwaru **QA**
Pali Rohár, Jirka Setnička
Výroba software není zdaleka jenom o programování. Ukážeme si, jak psát kód tak, aby nejenom fungoval, ale aby vyzařoval krásu a pokoj všem programátorům dobré vůle. Povíme si o různých způsobech testování, o tom jak udržet v kódu pořádek a dalších nástrojích, které pomáhají vyvíjet kvalitní software.
Předpoklady: Znát aspoň jeden příčetný programovací jazyk

Linuxový server *("Chci provozovat vlastní linuxový server, ale nevím jak.")* **ADMIN**
Jirka Setnička
Na co se mi hodí vlastní server a jak ho provozovat? Domácí server nebo něco sedícího v cloudu? A když už ho mám, tak jak ho zabezpečit a jaké věci se mi na něm hodí provozovat? Budeme si povídat o SSHčku, klíčích, šifrování, systemd, Apache a Nginxu, mailech, DNS, Let's Encrypt, zálohování a všem dalším, co nás bude zajímat. Ideálně prakticky na nějaké virtuálce. Pokud bude zájem, můžeme zabrousit i do různých způsobů automatického nasazení a konfigurace serverů, například deklarativní popis instalace pomocí Ansible.
Předpoklady: Základní znalost Linuxu.

Docker *("Stavíme služby jako kostičky.")* **DOCKER**
Vojta Káně, Jirka Setnička
Docker umožňuje snadno vytvořit kontejnery – třeba nějaké aplikace a všech jejích závislostí – a tyto kontejnery jednoduše distribuovat a spouštět. Docker se hodí ke snadné instalaci věcí s mnoha závislostmi nebo naopak stejné aplikace na mnoho serverů. Ukážeme si, jak si Dockerový kontejner vyrobit, jak z něj dostat porty a do něj naopak nějaké úložiště a kam kontejner umístit. Na závěr si ukážeme, jak provozovat více provázaných kontejnerů najednou (Compose) a pokud by zbyl čas, tak se možná dostaneme i k provozování kontejnerů ve větších clusterech (Kubernetes), aneb když chceme distribuovat zátěž na mnoho strojů a mít redundanci.

High-Performance Computing *("Jak krotit terabyty a jak trilobyty?")* **HPC**
Vojta Káně, Martin „Medvěd" Mareš, Jirka Kalvoda
Jak vymáčknout z počítače co možná největší výkon. Kdy optimalizovat a kdy raději ne. Jak si program zparalelizovat: aritmetický paralelismus, vektorové instrukce, symetrický i nepříliš symetrický multiprocesing, počítání na clusterech počítačů. K čemu je grafická karta. Lži, zatracené lži a benchmarky a co si z nich vybrat. Jak hledat v terabytovém textu.

Hardware a operační systémy

Cache-oblivious algoritmy *("Kešuješ, kešuje, kešujeme.")* **CACHE**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Dnešní procesory mají několik úrovní vyrovnávacích pamětí (cache), což způsobuje, že ačkoliv si jsou všechny části paměti rovny, některé si jsou rovnější. Jak taková cache funguje? Jak se procesor rozhodne, co si v ní zapamatuje a co vyhodí? Jak toho můžeme využívat při programování, aby naše programy běžely rychleji? Předvedeme kousek teorie i několik praktických ukázek s poněkud překvapivým chováním.
Předpoklady: Kešu oříšky

Filesystémy *("Aká až tučná môže byť FAT tabuľka?")* **FS**
Vojta Káně, Pali Rohár, Martin „Medvěd" Mareš
Ako sú dáta uložené na disku? Rozdelenie disku na partície pomocou MBR a GPT schémy. Ako funguje FAT (12, 16, 32) a jej rozšírenia VFAT, TFAT. Krok dozadu v podobe exFAT. Linuxové filesystémy EXT2 až EXT4. Multiplatformový UDF nielen na optické disky. Čo použiť na SSD? A ak ostane čas, tak niečo o UBIFS používaný na flash pamätiach bez radiču.

Audio cez bluetooth *("There are 14 competing bluetooth audio codecs (XKCD 927).")* **BTAUDIO**
Pali Rohár
Prečo je potrebné komprimovať 1,4 Mb/s zvukový stream pred tým než sa začne prenášať vzduchom do bluetooth slúchadiel, keď bluetooth zvláda prenášať 2 až 3 Mb/s? Ako sa prenášajú dáta vzduchom cez bluetooth protokoly ACL, L2CAP, RFCOMM a SCO? Čo podporujú bluetooth profily HSP, HFP a A2DP? A ako výrobcovia slúchadiel klamú o kodekoch CVSD, SBC, aptX, MP3 či LDAC? Ako je riešený bluetooth mikrofón a prečo vo väčšine prípadov jeho používanie rapídne znižuje kvalitu prehrávaného zvuku? Ako je možné, že na niektorých slúchadlách jeden kodek hraje subjektívne lepšie ako druhý a na iných slúchadlách je to presne naopak? Ktorý kodek je vhodný na počúvanie orchestrálnej hudby a ktorý zas na konferenčný VOIP hovor?

PCI Express *("Vysokorýchlostná zbernica, „papierovo" o rýchlosti až 1024 Gb/s, skutočne?")* **PCIE**
Pali Rohár
Dnes sa cez PCIe pripájajú k počítaču takmer všetky nové periférie včítane WiFi kariet, 5G modemov, NVMe SSD diskov, Thunderboltu a pod. Na tejto prednáške sa zoznámime s PCIe zbernicou po hardwarovej aj softwarovej stránke od fyzickej modulácie až po aplikačné posielanie paketov. Na akých princípoch je postavená, ako počítač a zariadenia na nej medzi sebou komunikujú. Aké stromové algoritmy musí software implementovať a ako sú PCIe protokoly podobné so sieťovým TCP/IP vrstevnatým modelom. Ako sa rieši bezpečnosť na zbernici, či ako prideliť grafickú PCIe kartu exkluzívne pre virtualizovaný operačný systém. Prehľad základných softwarových nástrojov, ktoré pomôžu pri debugovaní problémov s rôznymi PCIe/M.2 kartami.

Jak postavit (rychlý) procesor **CPU**
Jirka Kalvoda, Martin „Medvěd" Mareš
Vysypeme z rukávu, jak se dá vyrobit procesor, a půjdeme ho zrychlit. Proč nejde jednoduše zvýšit frekvenci, a jak to teda udělat. Jak ušetřit čas při přístupu do paměti, na co je ta slavná predikce skoků a jak se dá spouštět víc instrukcí najednou.

Bezpečnostní chyby v procesorech *("Sběrnicí obchází Přízrak a krade klíče.")* **CPUBUG**
Jirka Kalvoda, Martin „Medvěd" Mareš
Že jsou v programech bezpečnostní chyby, na to jsme si už zvykli. Ale teprve zvolna si zvykáme na to, že mohou být i v hardwaru, dokonce v samotném procesoru. Poslední tři roky přinesly několik ošklivých překvapení tohoto druhu s veselými jmény, jako je Meltdown a Spectre. Budeme se zabývat fungováním procesoru uvnitř, zejména všelijakými triky na zrychlení výpočtu: superskalárním zpracováním instrukcí, kešováním a predikcí skoků. A ukážeme, co pokazil Intel, co AMD a jak toho jde zneužít.

Linuxové jádro a jak se v něm vyznat *("Jak pořádně otestovat fsck?")* **KERN**

Pali Rohár

Co ten kernel vlastně je, čím se liší programování v kernelu od normálního kódu, jak sobě vlastní kernel postaviti a jak v něm něco opraviti. Kde najít nejnovější zdrojáky a kde najít pomoc, až se něco pokazí. Praktická ukázka na x86 nebo ARM zařízeních (notebook nebo router).

Mikrokontroléry *("Nejlepší debugger je LEDka.")* **MCU**

Martin „Medvěd" Mareš

Srdcem mnoha dnešních technických hraček je mikrokontrolér. To je čip, na kterém je integrovaný nejen procesor, ale i paměť a spousta zajímavých periferií. Ukážeme si, jak se mikrokontroléry programují, jaké periferie typicky obsahují a jak je používat ke komunikaci s okolním světem. Něco si vyzkoušíme i prakticky na STM32.

Předpoklady: Hodí se základní znalost jazyka C.

Správa paměti * *("Když má program sklerózu...")* **MEM**

Vojta Káně

Po chvíli zjistíme, že nám lokální a globální proměnné nestačí a je potřeba paměť alokovat dynamicky. Co všechno si musíme udělat sami a co se děje programátorovi „za zády". Mapování adresního prostoru, ruční alokování a vracení paměti a problémy s tím spojené (chyby programátora), počítání odkazů a daň s nimi spojená (a hele, cyklus), odklízeče odpadu (mark & sweep, kopírovací, generační a jiné triky).

Aktuální stav 3D tisku **3DPRINT**

Honza Černý

3d tisk je tu s námi už nějakou chvíli. Tak se pojďme ohlédnou kam se stihl posunout. Prozkoumáme nové technologie a principy a co vše dnes taková tiskárna dokáže. Také se budeme věnovat aktuálnímu trhu a na co si dát pozor při pořízení 3D tiskárny.

SystemD **SYSTEMD**

Vojta Káně

Internetovými diskutujícími nenáviděný soubor init systému a přidružených démonků je nejrozšířenějším svého druhu. Na každém šprochu pravdy trochu, ale má i své (možná i mnoho) světlé stránky. Navíc je fakt všude, takže se s ním setkáváme, i pokud nám srovna nevoní. Pojďme se blíže podívat zejména na onu init část, napsat si vlastní službu, číst její logy, spouštět ji periodicky a mnoho dalšího.

Sítě a bezpečnost

E-mail *("Drahoušek zákazník.")* **EMAIL**

Vojta Káně, Pali Rohár

Co se stane s e-mailem, když jej odešlete? Kudy chodí a kudy jej čerti nesou? Jaké máte záruky, že přijde; proč občas přijde pozdě nebo vůbec. Problém formátů a kódování, chyby webových i jiných klientů. Protokoly SMTP, POP, IMAP a co se stane, když do nich přimícháme SSL/TLS. E-mailová bezpečnost, SPAM a (nefunkční?) obrana pomocí SPF, DKIM a DMARC. Nakonec se podíváme na ne zrovna triviální grafový problém, který je v emailech skrytý.

IPv6 *("Viac adries znamená aj viac problémov.")* **IPV6**

Pali Rohár

Internetoví provideri nám pomaly začínajú ponúkať pripojenie do sveta IPv6 Internetu. Čo to IPv6 je? Čím sa líši od svojho staršieho kolegu IPv4? Prečo by sa o neho mal zaujímať aj obyčajný užívateľ? Pozrieme sa aké problémy IPv6 rieši, aké má výhody oproti IPv4 ako aj na súčasný stav podpory IPv6 v operačných systémoch. A hlavne s akými problémami sa človek bežne stretne než si bude môcť doma spraviť IPv6 sieť.

Předpoklady: NET

Tunely a lokálna sieť *("VPN? Dnes fičí WireGuard.")* **LOCALNET**

Pali Rohár

Každý z nás má doma nejakú tú „chytrú" škatuľku, pomocou ktorej sa pripája do Internetu. Ako takému zariadeniu poskytovateľ Internetu prideľuje IP adresy? A ako distribuuje IP adresy táto chytrá škatuľka do vnútornej siete? Protokoly používané v našich končinách (DHCPv4/v6, RA, PPPoE a DS-Lite). Preklady IPv4 adries (NAT) a presmerovanie portov. Ako rozdeliť domácu sieť na menšie celky, vytvoriť ďalšie podsiete a odizolovať určité zariadenia aby nemohli medzi sebou komunikovať? Virtuálne oddelené siete VLAN a VPN siete všeobecne. Prístup do domácej siete zvonku Internetu cez šifrované tunely (SSH, IPSec, WireGuard). Pripojenie domácej siete do IPv6 internetu iba po IPv4 linke (SIT, GRE, FOU tunely). A na záver problémy pri prepájaní všetkých druhov tunelov a sietí medzi sebou. Routovanie vs ARP/NDP proxy.

Předpoklady: NET

Hesla *("Byl tam dolar, nebo ampersand?")* **PASS**

David Klement

Kdo si má všechna ta hesla pamatovat? A ještě po mně chcete, abych je měnil každý měsíc? Hesla jsou dnes potřeba skoro všude, většině lidí však působí jen problémy. Povíme si, jak vylepšit své zabezpečení bez nutnosti pamatovat si desítky různých hesel. Většinu odvodíme na základě toho, jak se hesla ukládají na serverech a jak se prolamují. Nakonec ještě zmíníme jiné způsoby přihlašování, které hesla v mnohém překonávají.

Bezpečné programovanie *("V každom programe je aspoň jedna bezpečnostná chyba, tak poďme to napraviť.")* **SECPRG**
Pali Rohár
Dosť často sa programy nepíšu iba na jedno spustenie, ale sa používajú dlhé roky. Veľa z nich naviac bude používať úplne niekto iný. Na tejto prednáške sa pozrieme na časté chyby v programoch a ako im predchádzať. Ako klasifikovať či chyba je závažná a bezpečnostná. Ďalej si povieme, čo robiť v prípade, ak nájdeme nejakú (možno bezpečnostnú?) chybu. Komu ju oznámiť a komu radšej nie.

Aplikace kryptografie * *("6140 a184 c9a6 41f1 de99 e733 354a f451")* **CRYPT2**
Martin „Medvěd" Mareš
Pokročilejší a občas nečekané aplikace základních kryptografických primitiv. Jak přesvědčit server, že známe heslo, aniž bychom mu ho posílali? Jak zajistit, aby útočník nemohl dešifrovat komunikaci, ani když dodatečně získá soukromý klíč? Jak funguje BitCoin (decentralizovaná digitální měna) či Tor (protokol znemožňující komukoli po cestě vědět, kdo s kým komunikuje)?
Předpoklady: Základní povědomí o šifrování (CRYPT) a víra v existenci náhodných čísel

Praktická kryptografie *("A proč jsou všechny ty zámky na papírových dveřích?")* **PCRYPT**
Martin „Medvěd" Mareš
Programátoři si často myslí, že pro bezpečnou komunikaci stačí vybrat si z knihovny osvědčenou silnou šifru. Jak naivní! Navrhnout bezpečný protokol není maličkost a dá se při tom ledacos zpackat. Replay útoky (jak otevřít auto krabičkou za 30 dolarů), útoky na padding a na blokovou strukturu. Čí že je ten podpis? Jak nepoužívat RSA a jak nehešovat hesla. Jak náhodná jsou vaše čísla? Postranní kanály: časování, spotřeba, záření. K čemu se crackerům hodí termoska s tekutým dusíkem.

Eliptické krivky a kryptografia **ECC**
Pali Rohár
Kryptografia založená na eliptických krivkách sa vo veľkom nasadzuje a rozširuje po svete od smart kariet až po HTTPS servery. Na tejto prednáške si ukážeme, čo je sú to eliptické krivky, ako vyzerajú, ako sa s nimi dá počítať a ako ich použiť v kryptografii na šifrovanie alebo podpisovanie správ. Zároveň sa zoznámime s algebraickými pojmami ako sú abelovská grupa, či konečné teleso. Úvodná prednáška určená pre tých, ktorí ešte o eliptických krivkách nič nevedia.

Web uvnitř *("Error 402: Payment Required. Please insert a coin.")* **HTTP**
Jirka Setnička, Martin „Medvěd" Mareš
Většina webu je dnes založena na protokolu HTTP, pojďme se podívat, jak funguje uvnitř. Metody GET, POST, ale třeba i PUT. Dohadování o typu dat. Cacheování, revalidace a transformace dat. Křupavé sušenky. Jak se vypořádat s dynamicky generovaným obsahem aneb protokoly CGI, WSGI apod. Mezi klientem a serverem aneb DNS a virtuální servery. Nakonec do toho všeho přimícháme SSL/TLS a máme HTTPS. Malá ochutnávka HTTP/2.0 a 3.0.

Telefonovanie cez internet *("Postavme si vlastnú telefónnu ústredňu.")* **VOIP**
Pali Rohár
Pod skratkou VOIP sa označuje prenos hlasu cez internet. Na tejto prednáške sa budeme venovať protokolu SIP a pridruženým protokolom SDP a RTP, ktoré sú azda najviac rozšírené. Takmer všetky dnešné pevné a internetové linky, včítane mobilných VoLTE a VoWiFi pripojení, ktoré poskytujú telefónni operátori zákazníkom, sú práve na protokole SIP. Ako prebieha priame volanie na IP adresu telefónnu, ako sa volá cez prostredníka (proxy server) a ako do PSTN siete. Ako sa riešia problémy s NAT a prečo je SIP ALG taký zlý. Akým spôsobom sa dá prenášať FAX, textová správa, krátka správa (SMS), video hovor alebo konferenčný audio/video hovor. DNS záznamy pre verejné telefónne čísla (ENUM) a prečo sa to neujalo.
Předpoklady: NET

Grafika a typografie

Barevné systémy *("Co je na konci duhy?")* **COLOR**
Martin „Medvěd" Mareš
O podstatě světla a barevného vidění a různých pokusech o reprezentaci barev v počítačích, fotoaparátech, televizích a podobných zařízeních. Systémy RGB, CMY(K), HSV, XYZ, Lab s jejich výhodami i neduhy. „Systém" Pantone. Reálné kontra imaginární barvy aneb proč nejde vyfotit duha.

Typografie *("What You See Is all What You've Got!?")* **TYPO**
Martin „Medvěd" Mareš
Jak na počítači text nejen napsat, ale také vysázet tak, aby pěkně vypadal a aby (což je důležitější) se i příjemně četl. Jak se sází pohádka, jak báseň a jak vzorové řešení KSP plné komplikovaných vzorců. Jak jde dohromady staleté umění typografické a moderní technika. Přineste knihy i letáky, zkritizujeme sazeče, co se do nich vejde.

TEX *("No pages of output. Ask a TEXnician.")* **TEX**
Jirka Kalvoda, Jirka Setnička, Martin „Medvěd" Mareš
Z předchozí přednášky máme představu o tom, jak vypadá pěkná sazba. K její výrobě nám pomůže typografický systém TEX. Praktická přednáška s ukázkami použití TEXu od hladké sazby knihy až po zběsilosti hraničící s programováním. Jak do TEXu vkládat obrázky a jak to raději nedělat. Kde shánět další informace: TEXbook, TEXbook naruby a další zajímavá literatura. Praktické rozdíly mezi různými dialekty TEXu. Všelijaká rozšíření: pdfTEX, eTEX, LuaTEX.

TEXnické detaily ** *("TEX capacity exceeded. Ask a wizard to enlarge me.")* **TEX2**
Jirka Kalvoda, Martin „Medvěd" Mareš
Pokročilejší přednáška o TEXu pro ty, kdo ho už nějaký čas používají. Budeme v TeXu programovat, kreslit obrázky, otáčet text, používat různé podivné fonty a třeba si i vysázíme odstavec ve tvaru kolečka.

Asymptote *("Vy obrázky kreslíte? My je programujeme!")* **ASY**
Jirka Kalvoda, Martin „Medvěd" Mareš
Rádi byste své řešení KSP ozdobili hezkými obrázky? Dají se nakreslit ručně, ale často je snazší obrázky programovat. Předvedeme Asymptote, což je programovací jazyk určený na kreslení 2D a 3D obrázků. Také se zastavíme u jeho předchůdců MetaPostu a MetaFontu a knihovny pro vektorové kreslení Cairo.

Formát PDF **PDF**
Martin „Medvěd" Mareš
Jeden z nejrozšířenějších formátů na předávání dokumentů má za sebou spletitou historii i dokumentaci. Ukážeme si, jak vypadá uvnitř a co se do něj dá uložit: grafické objekty, text, fonty, odkazy, všelijaké anotace a meta-data, a dokonce i kryptografické podpisy. Zmíníme se o profilech, třeba PDF/X a PDF/A. Při troše štěstí si vytvoříme jednoduchý PDF soubor ručně a možná půjde i otevřít.

Unicode *("Jaký kód má sněhulák s kudrnatými vlasy?")* **UNI**
Pali Rohár, Martin „Medvěd" Mareš
Jak funguje znaková sada Unicode, která se snaží zapsat všechny jazyky světa? Codepointy versus glyfy. Kombinující znaky, čtvero normálních forem a pátá lehce nenormální. Typografické a neviditelné znaky. Co všechno prozradí Unicode Character Database. Uložení v paměti: formáty UCS-2, UCS-4, UTF-8 a UTF-16, nešvar s BOM. Tajemný svět emoji. Jak se s Unicode programuje? A jako vždy: bezpečnostní problémy.

Teoretická informatika

Persistentní datové struktury * *("Datové struktury cestující časem.")* **PERS**
Jirka Kalvoda, Martin „Medvěd" Mareš
Ukážeme si (téměř) obecný způsob, jak naučit datové struktury zapamatovat si celou svou historii. Předvedeme si, jak tuto historii modifikovat a k čemu to je celé dobré.

Třídy složitosti ** *("Kolik sekund stojí jeden bajt?")* **SLOZ3**
Jirka Kalvoda, Martin „Medvěd" Mareš
Složitost opravdu důkladně: nejrůznější třídy složitosti a vztahy mezi nimi. Vztahy mezi časem a prostorem, odstraňování nedeterminismu a Savitchova věta. Jak víme, že všechny třídy nejsou stejné: dolní odhady a věty o hierarchii. Stroje s kvantifikátory, třída PSPACE a polynomiální hierarchie. Pravděpodobnostní třídy složitosti. Orákula a neuniformní složitost.
Předpoklady: SLOZ2

Pravděpodobnostní algoritmy *("Kudy dál? Hoďme si kostkou!")* **PPALG**
Jirka Kalvoda, Martin „Medvěd" Mareš
Když nevíme, jak se v algoritmu rozhodnout, někdy pomůže ponechat to náhodě a prostě si „hodit kostkou". Dokážeme sestrojit algoritmy, které jsou rychlé, i když spravný výsledek vydají jen v 99 % případů. Ale i takové, které odpoví správně vždycky, ale rychlé jsou jen v průměru (třeba QuickSort). Též ukážeme, jak pomocí náhody zabraňovat kolizím v hešování.

Vyčíslitelnost ** *("S Halting problémem na věčné časy!")* **VYCIS**
Martin Koreček, Martin „Medvěd" Mareš
Některé problémy se dají vyřešit snadno, jiné obtížněji a některé dokonce vůbec. Obecněji: Ať si vymyslíte jakýkoliv rozumný programovací jazyk, vždycky existuje problém, který se v něm nedá vyřešit. Jak se ale dokazuje, že něco nejde? Matematický pohled na výpočetní modely a univerzální stroje, rekurzivně spočetné a rekurzivní množiny a funkce. Halting problem a diagonální důkazy. Vždycky může být hůř: Turingovy stupně a aritmetická hierarchie.

Neuronové sítě **NEURO**
Honza Černý, David Klement, Michal Kodad
Základy neuronových sítí: od biologického neuronu a jeho modelu – perceptronu, přes algoritmus back propagation, až po hluboké a konvoluční sítě a jejich použití na rozpoznávání obrázků.

Dlaždičková složitost *("Co je negace koupelny?")* **TILES**
Jirka Kalvoda, Martin „Medvěd" Mareš
Nadefinujeme trochu netradiční počítač založený na dlaždičkách v koupelně. Prostudujeme, jak se různé druhy dlaždičkových počítačů chovají, a zjistíme, že to docela dobře odpovídá klasické teorii složitostních tříd. Jaké problémy má matematik, jehož koupelna je nekonečně velká?

Kvantové počitání ** *("`return 0.5*dead + 0.5*alive;`")* **QC**
Martin „Medvěd" Mareš
Stručný úvod do kvantového počítání. Kvantová superpozice stavů výpočtu a její kolaps při měření. Základní kvantové operace: negace, řízená negace, permutace, Hadamardovo hradlo, Tofolliho hradlo. Groverův algoritmus na hledání v odmocninovém čase. Kvantová Fourierova transformace a Shorův algoritmus pro faktorizaci.
Předpoklady: Znalost komplexních čísel je nutností, znalost lineární algebry výhodou.

Buněčné automaty a Game of Life *("Čtverečkovaný svět, co není Minecraft")* **LIFE**
Martin „Medvěd" Mareš
Game of Life je dvojrozměrný svět, ve kterém se buňky vyvíjí podle průzračně jednoduchých pravidel. Už desítky let v tomto světě objevujeme další a další zajímavé jevy. Tak do něj také nahlédneme, prozkoumáme souvislosti s evoluční biologií i s algoritmy. Též uvidíme, jak Život zapadá do obecnějšího světa buněčných automatů.

Aplikace informatiky

Čárové kódy *("Jak naučit počítače číst láhve od Coly.")* **BAR**
Martin „Medvěd" Mareš
Čárové kódy dnes potkáváme na každém kroku, ale jak doopravdy fungují? Prozkoumáme klasické jednorozměrné kódy (UPC, EAN, Code39, Code128), jakož i novější dvojrozměrné (QR, Aztec, DataMatrix). Kódovací a dekódovací algoritmy plus trocha matematiky okolo zabezpečení proti chybám. Další počítačem čitelné značky: RFID, bíle křížky na asfaltu, ...

Komprese dat *("Jnm idln kpln j nstlčtln.")* **ZIP**
Jirka Setnička, Martin „Medvěd" Mareš
Pokud jsou data příliš velká, můžeme je zkusit zkomprimovat. Předvedeme základní kompresní algoritmy: triviální (RLE), slovníkové (LZ77), statistické (Huffmanovo a aritmetické kódování) a některé pokročilejší techniky, jako třeba Burrowsovu-Wheelorovu transformaci (BZIP). Zmíníme se o kompresi zvuku, obrazu a videa (prediktory, wavelety, všelijaká ztrátová komprese).

OpenStreetMap **OSM**
Jirka Setnička
Otevřené mapy OpenStreetMap fungují trochu jinak, než klasické kreslení map. Je to vlastně obrovská databáze správně otagovaných bodů, cest a relací, ze kterých se pak dají vykreslit zajímavé mapy (třeba jak by vypadala Česká Republika, kdyby stoupla hladina světových moří o 300 metrů), ale zároveň i počítat ještě zajímavější věci. Navíc do OSM může přispívat každý a zmapovat si třeba malou stezku v v horách v Makedonii nebo novou lavičku u babičky ve vesnici. Ukážeme si vnitřní strukturu, typické tagy a to, jak provést svoji první editaci v OSM.

Matematické přednášky

Pravděpodobnost **PAST**
David Klement, Michal Kodad
Jak pracovat s pravděpodobností matematicky. Ukážeme si pravděpodobnosti jevů, nezávislé jevy střední hodnotu, náhodné proměnné a další. Také si vše procvičíme na několika příkladech. Pokud zbyde čas, tak si také ukážeme, jak se dá pravděpodobnost využít v informatice.

Teorie množin a matematika nekonečen * *("Kdo je nejvyšším z kardinálů?")* **TEMNO**
Martin „Medvěd" Mareš
Historie matematiky je dlážděna trampotami s nekonečnem. Začalo to roztomilým problémem s želvou pana Zénona a vedlo až k poněkud děsivým paradoxům 18. století. V moderní době jsme se proti tomu obrnili teorií množin, na níž je dnes takřka celá matematika postavena. Jak se taková teorie buduje a jak se pomocí ní popisují nekonečné objekty. Množiny a jejich velikosti. Cantorův diagonální trik. Ordinály a houšť kardinálů. Potenciální kontra aktuální nekonečno. Jak si pořídit přirozená čísla a jak ta reálná. Potíže s axiomem výběru.

Teorie čísel a RSA * *("$2^{67} - 1 = 193\,707\,721 \cdot 761\,838\,257\,287$")* **NUT2**
Martin „Medvěd" Mareš
Pokračování teorie čísel, které nás dovede až k RSA – asi nejpoužívanějšímu asymetrickému šifrovacímu algoritmu dnešní doby. Počítání modulo složené číslo a Eulerova věta. Jak RSA funguje, proč funguje a jestli bude ještě fungovat. Generování klíčů, faktorizace kontra testování prvočíselnosti. Časová složitost aritmetiky.

Fourierova transformace ** **FFT**
Honza Černý & Jirka Kalvoda, Martin „Medvěd" Mareš
Jak rychle umíte násobit n-ciferná čísla? My to umíme lineárně. Hodí se k tomu chytrý trik pana Fouriera, který už dávno patří k matematické a fyzikální klasice. Ukážeme, co je Fourierova transformace zač, jak ji rychle spočítat a k čemu je dobrá: rychlé násobení polynomů i čísel, digitální zpracování zvuku a obrazu (spektrální analýza či třeba komprese).
Předpoklady: Základy komplexních čísel

Meta-matematika * *("Tato věta sem nepatří.")* **METAM**
Martin „Medvěd" Mareš
Pokud budeme v životě věřit všemu, co je „přeci zřejmé", dostaneme se brzy do potíží a v matematice to platí dvojnásob. Přírodní vědy si vymyslely opakovatelné pokusy a matematici axiomatický přístup. Ukážeme, jak z jednoduché sady axiomů vybudovat takřka celou matematiku. Dokonce tak, že správnost důkazů za nás ověří počítač, aspoň když mu trochu pomůžeme. Nadšení trochu ochladí Gödelova věta: ať děláme, co děláme, vždy zbude nějaké nerozhodnutelné tvrzení. Pomůže přidávat axiomy? Asi ne, ale za odměnu získáme mnoho různých matematik. A dá-li bůh, stihneme dokázat jeho existenci i neexistenci ☺.

Teorie (vesměs samoopravných) kódů *("f y cn rd ths, y wll b gd cmptr prgrmmr!")* **KODY**
David Klement, Martin „Medvěd" Mareš
Jak komunikovat po lince, která průměrně každý k-tý bit přenese špatně? K tomu se hodí teorie samoopravných kódů, která nás naučí: vzdálenost slov a jejich souvislost s detekcí a opravou chyb, paritní a lineární kódy, perfektní kódy, Reed-Solomonovy a vůbec polynomiální kódy a několik dolních odhadů nádavkem. A jak s teorií kódů souvisí třeba čeština?

Lineární programování jako blackbox * **LPBB**
Martin Koreček
Lineární programování je ohromně užitečná optimalizační technika. V přednášce se nebudeme zabývat teorií, a raději si ukážeme co nejvíce praktických využití této techniky zejména pro návrh efektivních algoritmů. Bude se hodit předem znát definici NP-těžkého problému.

Řešení těžkých problémů ** **HARDSOL**
Martin Koreček, Jirka Kalvoda
NP-těžkých problémy se v praxi objevují velmi často a zdatný algoritmik si s nimi musí umět poradit. Ukážeme si pár aproximačních algoritmů, které nenajdou optimální řešení, ale nějaké jemu blízké, a také techniky návrhu exponenciálních algoritmů, které mohou za určitých okolností být přijatelně rychlé.
Předpoklady: HARD

Komplexní a komplexnější čísla *("$1 = \sqrt{1} = \sqrt{(-1)(-1)} = \sqrt{-1}\sqrt{-1} = i \cdot i = i^2 = -1$. Huh?")* **CPLX**
David Klement
Jak se nám matematika změní, když připustíme, že se záporná čísla také dají odmocňovat? Čísla imaginární a komplexní a jejich různé podoby. Součtové vzorce pro sin a cos dostaneme téměř zdarma. K čemu se hodí v matematice a k čemu ve fyzice. Proč se zastavit u dvou složek aneb kvaterniony, oktoniony a Cliffordovy algebry. Remember, life is complex.

Úvod do matematické analýzy * *("$\lim_{8\to 9} \sqrt{8} = 3$")* **MA**
David Klement, Honza Černý
Jak zjistit, jaký tvar má graf nějaké funkce? Jak najít její minimum? Jak spočítat délku spirály nebo objem sudu (třeba i čtyřrozměrného)? Jak spočítat $\sin x$ nebo třeba π? Na to všechno se hodí limity, derivace a integrály. Nejprve si o nich vybudujeme jednoduchou geometrickou představu, pak je nadefinujeme pořádně a naučíme se s nimi počítat.

Catalanova a Fibonacciho čísla * *("1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, ?")* **CAT**
Martin „Medvěd" Mareš
Kolik existuje binárních stromů? Kolika způsoby jde uzávorkovat výraz? A kolika způsoby projít čtvercovou mřížku, aniž bychom překročili úhlopříčku? Kam oko pohlédne, všude se skrývají Catalanova čísla. Kromě případů, kdy za ně zaskakují čísla Fibonacciho. Povídání o dvou zajímavých posloupnostech a jejich početném příbuzenstvu. Dlouhá cesta od hezkého vzorečku k rychlému algoritmu.

Grupy a tělesa *("Síla abstrakce.")* **GROUP**
David Klement, Honza Černý
Už před dlouhou dobou někoho napadlo, že místo čísel můžeme počítat s písmenky, která nám nějaká konkrétní čísla zastupují, a plno věcí si tím usnadníme. Proč se ale omezovat jen na čísla? Předvedeme si, že sčítat můžeme libovolné dvě věci, tedy aspoň když si jejich součet nadefinujeme tak, aby dával smysl. Přidejme ke sčítání i odčítání a dostaneme grupu. Ukážeme si, že grupy jsou všude kolem nás a že nám umožní zkoumat vlastnosti věcí, u kterých nevíme, jak vypadají. Jak grupy souvisí s prvočísly, jak s šifrováním a jak s osovou souměrností. Nakonec grupy rozšíříme na tělesa, které zobecňují nám dobře známé číselné obory.

Logika *("Tato věta sem nepatří.")* **LOGI**
Honza Černý, Martin „Medvěd" Mareš
Pokud budeme v životě věřit všemu, co je „přeci zřejmé", dostaneme se brzy do potíží a v matematice to platí dvojnásob. Ale co s tím? Přírodní vědy si vymyslely verifikovatelné experimenty a matematici logiku a dokazování. Co je to výrok, co jeho důkaz a proč se axiomy nedokazují. Jenže jak si je zvolit? A jak se z toho všeho postaví celá matematika? A bude vůbec matematika někdy celá? Studená sprcha pana Gödela coby sebevražedné dovršení snahy získat dokonalý jazyk. Logika coby hra a problém líného profesora. Důkazy boží existence a neexistence.
Předpoklady: LOGI nebo její negace

Barevnost grafů * *("Bílá, modrá, červená, co to pro graf znamená?")* **BAGR**
Jirka Kalvoda
V teorii grafů zaujímá významné místo problém barevnosti grafu, tedy přiřazení co nejmenší počtu barev vrcholům tak, aby se hranami dotýkaly pouze různobarevné vrcholy. Aplikace problému v informatice je nasnadě. Ukážeme si několik zajímavých teoretických výsledků. Obarvení některých druhů grafů, $L_{2,1}$ barevnost aneb problém vysílačů, vybíravost, kruhová barevnost a další.

Rovinné grafy *("Kdo nakreslí pět souvislých států tak, aby každý sousedil s každým, má u mě čokoládu.")* **ROG**
Jirka Kalvoda, Martin „Medvěd" Mareš
Povídání o grafech, které jde nakreslit na papír bez křížení hran. Co všechno pro takové grafy platí a jak je poznáme, aniž bychom je museli kreslit. Existuje pouze 5 pravidelných mnohostěnů, a my se o tom pomocí teorie grafů přesvědčíme. Barvení rovinného grafu šesti a možná i méně barvami. Když zbyde čas, zkusíme grafy kreslit i na jiné plochy: kupříkladu Möbiovu pásku, pneumatiku nebo ušatou kouli.

Křivky v počítačové grafice * *("Jak se měří elegance křivky?")* **BEZI**

Martin „Medvěd" Mareš

Jak se na počítači kreslí křivky, které „vypadají hezky", třeba tvar karoserie auta nebo tvar písmenka? Kružnice a jiné kuželosečky se k tomu moc nehodí, tak se poohlédneme po obecnějších křivkách. Základy matematiky okolo Bernsteinových polynomů, Bézierových křivek a spline funkcí. Práce s křivkami pomocí rekurzivního rozkladu a de Casteljauova algoritmu. Matematické modelování estetiky.

Předpoklady: Pro část přednášky se hodí vědět, co je derivace, a nebát se ji použít.

Fyzikální přednášky

GPS *("Čekám na signál...")* **GPS**

David Klement, Jirka Setnička

GPS je vstutku magická technologie, a ačkoli se to nezdá, využívá jedny z nejdůležitějších výsledků moderní vědy. Vysvětlíme si, na jakém pricipu GPS funguje, a proč by to jinak nešlo.

Půlnoční přednášky

Aneb přednášky přednášené (nejen) o půlnoci na různá zajímavá témata nejen o informatice. Pokud nějaká z nich nebude oficiálně vypsaná, je možné si konkrétního organizátora ve volné chvíli chytit a přesvědčit ho k přednášení.

Organizování a práce v týmu *("Ten dělá to a ten zas tohle aneb co obnáší organizátorem býti.")* **ORG**
Jirka Setnička, Jirka Kalvoda, Martin „Medvěd" Mareš
Volné povídání o tom, co se všechno skrývá za organizováním různých seminářů a podobných akcí, primárně pak KSPčka. Jaká práce, jaké radosti a jaké starosti s sebou organizování nese, co se přitom člověk může naučit a také pár cenných rad do života. Jak se z toho nezbláznit a pár bláznivých příhod k tomu.

Základy první pomoci *("Jak někomu zachránit život a jak málo k tomu stačí.")* **ZDRAV**
Jirka Setnička
Pobavíme se o základech první pomoci. Jak správně vyhodnotit situaci a kdy je potřeba volat pomoc? Jak se postarat o člověka v bezvědomí, jak kontrolovat životní funkce a jak člověka stabilizovat do příjezdu pomoci? Ukážeme si, jak málo stačí k záchraně života a naučíme se nebát se první pomoci. A také, že naše bezpečí je v každé situaci na prvním místě.

Lockpicking *("Jak si odemknout, když si náhodou my (nebo soused) zapomeneme klíč :-)")* **PICK**
Jirka Setnička, Michal Kodad, Honza Černý
Jak fungují dnešní zámky, co jsou to stavítka a jak vlastně fungují klíče. A jak se pomocí jednoduchých nástrojů dají využít výrobní nedokonalosti zámků k jejich odemčení. Použití planžet, napínáků, praktické ukázky odemykání, nastínění technik bumpingu a dalších postupů, jak se dostat přes zamčené dveře.

Klávesové zkratky *("`Ctrl+Shift+Alt+Super+J`")* **KEYB**
David Klement, Jirka Kalvoda
Jakmile se rozhodnete ovládat počítač klávesnicí, brzy zjistíte, že anglická abeceda má příliš málo písmenek pro všemožné klávesové zkratky. Nemluvě o tom, jak si všechny zkratky zapamatovat. Přednáška až diskuze o různých přístupech, které výše zmíněné problémy řeší.

Lingvištika *("Přísudek je v této větě podmět.")* **LING**
Martin „Medvěd" Mareš
Převážně nevážné a mírně nepřed-vídatelné po-vídání o jazyku i jazyce. Základní jazykové rodiny a jejich podobnosti i odlišnosti. Co má společného čínština s angličtinou a co nikoliv. Proč jeden jazyk potřebuje 15 pádů, zatímco jiný se bez nich obejde úplně. Jak se jazyky vyvíjejí a jak se navzájem ovlivňují. Kde se berou jazyková pravidla. Kde se vzalo písmo a proč se mluvený a psaný jazyk tolik liší. Jak se na jazyk dívá matematik a jak se na matematiku dívají lingvisté.

Fonetika *("Pojďte, zachrochtáme si spolu!")* **FON**
Martin „Medvěd" Mareš
Malá inventura zvuků, které lidé dovedou vytvářet, a jejich použití v komunikaci. Různé způsoby vytváření a modulace zvuku. Kolik různých B dokážete říci? Fonetické kontrasty a co si z nich různé jazyky vybraly. Rázy, polosamohlásky a jiní obyvatelé polosvěta. Přízvuk kontra délka. Asimilace, přehlasování a další „principy líné huby." Vše prakticky procvičíme.

Orientace **ORI**
Martin „Medvěd" Mareš
Jak ze neztratit v terénu a jak se neztratit na moři. Vývoj umění navigace. K čemu je důležité slunce a hvězdy, ale proč mořeplavcům nestačí, alespoň dokud neobjevíme hodinky. Použití mapy, busoly a GPSky. Orientace bez pomůcek a použití Ariadniny nitě. Bleskový úvod do sférické astronomie a časomíry čili jak (ne)postavit sluneční a třeba i měsíční hodiny. Jak reprezentovat mapu v počítači a jak raději ne. Jak zapisovat polohu místa na Zemi (přestože Země má tvar podivně nakousnuté hrušky) a kolika způsoby to jde. Různé druhy map a jejich (z)kreslení. Jak se neztratit v kartografii. Praktické cvičení v terénu.

Čaj *("Jak vypadá odvar z nezralých pražců?")* **TEA**
Martin „Medvěd" Mareš
Pojďme usednout k šálku lahodného čaje a povídat si o tom, co se v něm skrývá. Kde se čaj vzal, kde se pěstuje, jak se zpracovává a jak ho připravovat. Trocha čajového zeměpisu, dějepisu i čajové chemie a čajové kultury. Též o všelijakých substancích čaji podobných.

Hausdorffův zvěřinec ** *("Jaký objem má π-rozměrná koule?")* **HAUS**
Martin „Medvěd" Mareš
Možná vás už také zarazilo, že některé fraktály nejsou ani dvourozměrné, ani třírozměrné, ale něco mezi tím. Pojďme se podívat, co to znamená. Cestou potkáme různé zajímavé partie matematiky (jako třeba metrické prostory a teorii míry) a různá podivuhodná zvířátka: Cantorovo diskontinuum, von Kochovu vločku a Hilbertovu křivku.

TempleOS *("Když ti bůh řekne, aby jsi mu napsal operační systém.")* **TEMPLEOS**
Honza Černý
Tato přednáška je povídání o operačním systému TempleOS a hlavně jeho autorovi. Tento hořko sladký příběh je plný těžkých osudů, zvláštních náhod a jak se TempleOS stal bizarní a přesto významnou součástí historie internetu.

Hlasitá přednáška *("Jak nehoda při obsluze soustruhu založila hudební žánr.")* **LOUD**

Honza Černý

Tato přednáška je chronologickým a snad i trochu systematickým náhledem na vývoj metalové hudby a žánrů z nich odvozených. Přednáška bude nejen o hudbě, ale i příbězích o jejich interpretech. Přednáška nejen pro posluchače metalové hudby, ale i pro lidi, kteří se s ní nikdy nesetkali.

Nová náboženská hnutí *("Vše co potřebujete vědět, aby jste si mohli založit sektu.")* **CULT**

Honza Černý

Od poloviny minulého století přibylo mnoho nových náboženství a duchovních hnutí. Povíme si o jejich historii a dopadu na společnost. Povíme si i o psycholigii schovanou za sektami a jak vlastně interně fungují. Povíme si i o charismatizaci a jak ji poznat.

Hnutí nového věku *("Jak vyrobit boha na míru.")* **NEWAGE**

Honza Černý

Žijeme v době, kdy můžeme být svědkem velkých změn a to dokonce i ve vnímání duchovna a životních hodnot. Řekneme si o historickém vývoji náboženského směru "new age" a jeho aktuální podobě. Cílem přednášky je pochopit vývoj hodnotových systémů a jak ovlivňuje jednotlivce. Přednáška z velké části vychází z pozorování Karla Gustava Junga.

git-annex *("Git pro velké soubory a netradiční použití")* **ANNEX**

Vojta Káně

„Propadl jsem Gitu a potřebuju terapii." Tak to jsi tu špatně. Neporadím, jak se této závislosti zbavit, pouze jak s ní přežít. git-annex je nadstavba Gitu napsaná v Haskellu, která nám umožní verzovat i vskutku velké soubory, ukládat si je jen v některých klonech repozitáře a přitom si držet přehled, kde, a také k souborům přilepit spoustu metadat. Chcete verzovat rodinné fotoalbum, záznamy přednášek, nebo třeba obrazy disků, ale Git (nebo váš počítač) nepříjemně sípou? git-annex je ta správná medicína

Abecední seznam přednášek

Základní přednášky

Rozšiřující přednášky

Půlnoční přednášky