Podzimní soustředění KSP 2013 – 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 (třeba na fórum), 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.

Algoritmy a datové struktury

Jak vypadá řešení *("Jen jeden bod, když jsem napsal 18 stránek?")* **SOL**
Jak má správně vypadat řešení KSP? Na co si dát pozor, co je úplně špatně a za co organizátoři strhávají body a sobě vlasy. Přednáška, která by mohla pomoci i mnohým déle aktivním řešitelům.

Algoritmy a jejich složitost *("Čím menší je časová složitost algoritmu, tím větší je složitost kódu.")* **SLOZ**
Problém, algoritmus a program. Časová a paměťová složitost problémů i algoritmů. Složitost rekurzivních algoritmů, složitost v průměrném případě. Ukázky jednoduchých (obvykle aritmetických a třídicích) algoritmů a výpočet jejich složitosti.

Základní algoritmy **ZALG**
Základní výbavou každého informatika jsou různé standardní algoritmy, zde si ukážeme ty nejdůležitější z nich. Třídicí algoritmy – porovnávací i přihrádkové. Hledání k-tého nejmenšího prvku v lineárním čase. Práce s výrazy a železničářský algoritmus.

Grafy & algoritmy *("Pojďme si hrát s obrázky")* **GA**
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.

Prohledávání do hloubky **DFS**
Karel Tesař, Martin Mareš, Karolína Burešová, Pavel Veselý, Filip Štědronský
Trochu hlubší pohled na prohledávání do hloubky. Jeho (často dost nečekané) aplikace v dalších algoritmech, jako je třeba hledání mostů, topologické třídění, rozklad na komponenty silné souvislosti či kreslení grafu jedním tahem.

Nejkratší a jiné cesty * *("Všechny cesty vedou do Horní Dolní, jen některé přes Řím.")* **CESTY**
Martin Mareš, Jirka Setnička, Lucie Mohelníková, Karel Tesař, Pavel Veselý, Filip Štědronský
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.

Toky v sítích *("Když je v grafu povodeň, těsní?")* **TOKY**
Martin Mareš, Michal Vaner, Jirka Setnička, Karel Tesař, Pavel Veselý
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**
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**
Michal Vaner, Martin Mareš, Karel Tesař
Důmyslnější varianty vyhledávacích stromů: splay stromy, BB-α stromy, vícerozměrné stromy. Chytřejší haldy: binomiální, Fibonacciho, 2-3. Amortizovaná analýza složitosti. Též několik přátelských randomizovaných datových struktur: skip listy a treapy.

Datové struktury pro šílence ** **DS3**
Martin Mareš
Ještě důmyslnější datové struktury dle přání posluchačů. Možno servírovat například různé dynamické reprezentace grafů (Sleatorovy-Tarjanovy stromy, ET-stromy, Fredericksonovy topologické stromy), geometrické datové struktury, obecné dynamizační schéma, triky pro malé integery, persistentní datové struktury et cetera.

Intervalové stromy * *("Já bych ty intervaly nejradši... dal do stromu!")* **ITREE**
Karel Tesař, Jirka Setnička, Karolína Burešová
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**
Karel Tesař, Jirka Setnička, Karolína Burešová, Filip Štědronský
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?")* **REGEX**
Martin Mareš, Jirka Setnička, Karel Tesař, Filip Štědronský
Jak hledat podřetězce ve velkém množství textu. Proč se ve vstupu vracet neboli Knuthův-Morrisův-Prattův algoritmus. Hledání více řetězců najednou podle Aha a Corasickové. Okénkové hešování Rabina a Karpa. Konečné automaty teoreticky i prakticky, regulární a „regulární" výrazy.

Stringové algoritmy * *("Co se nedá spočítat v lineárním čase, nestojí za to.")* **STRG**
Martin Mareš
Předvedeme všeliké algoritmy na zpracování řetězců, které mají (mimo jiné) společné to, že pracují v lineárním čase: třídění za pomoci kyblíčků, konstrukce suffixových stromů (aneb jak obrátit řetězec naruby) a jejich použití, nebo třeba hledání nejdelšího společného podřetězce dvou řetězců.
Předpoklady: REGEX aspoň zhruba

Suffixové pole a jeho použití v úlohách * **SUFFARR**
Karel Tesař
Možná jste již slyšeli o suffixovém stromě, to je strom, pomocí kterého umíme téměř jakýkoliv řetězcový problém vyřešit v lineárním čase. Bohužel jeho konstrukce je opravdu složitá a vyžaduje nemálo triků. My si představíme suffixové pole plus jeho konstrukci, se kterým dokážeme téměř jakýkoliv řetězcový problém vyřešit v čase $\mathcal{O}(n \log n)$. Takové pole již dokážeme implementovat celkem rychle a můžeme jej použít třeba v programátorských soutěžích.

Úlohy na stromech * **STROM**
Karel Tesař
Ukážeme si dvě úlohy na stromech, které bývají základní myšlenkou pro mnohé další. Pomocí těchto dvou základních myšlenek si pak se stromy budeme umět v mnohých případech poradit. Konkrétně jsou to: hledání nejbližšího společného předchůdce (plus převod na úlohu hledání minima na intervalech posloupnosti) a Heavy-light dekompozice stromu.

Komprese dat *("Jnm idln kpln j nstlčtln.")* **PRESS**
Martin Mareš
Přehled základních kompresních algoritmů: triviální algoritmy (RLE), statistické metody (Huffmanovo a aritmetické kódování), slovníková komprese (LZ77, LZ78, LZW), Burrowsova-Wheelerova transformace (BZIP). Pokud zbude čas, tak i něco o ztrátové kompresi obrázků a zvuku (prediktory, wavelets, JPEG, MPEG, fraktály).

Magické algoritmy * *("Pokročilá technologie není rozlišitelná od magie.")* **MAGIC**
Martin 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.

Přehled umělé inteligence **AI**
Pavel Veselý, Michal Pokorný
Kdy se vyplatí nakoupit akcie? O čem je tenhle článek? Kde jsem nechal klíče? Počítače se musí prát s problémy, které jsou strašně těžké, ale někdo je stejně řešit musí. Jak to dělají? To se dozvíte na této přednášce. Ukážeme si reprezentaci vnějšího světa a automatické uvažování nad jeho obsahem, na což naváže vyhledávání ve stavovém prostoru a úvod do programování s omezujícími podmínkami. Následovat bude pár základních algoritmů strojového učení. Nakonec se můžou podávat speciality z oblasti zpracování přirozeného jazyka, umělého vidění, nebo třeba plánování a rozvrhování.

Herní algoritmy *("Když nemáte na to, abyste vyhráli šachový turnaj...")* **AIGAME**
Pavel Veselý, Michal Pokorný
Povídání o tom, jak programovat počítačové soupeře do šachů a her jim podobným. Základní minimaxový algoritmus a jeho vylepšení neboli α-β ořezávání. Stále pomalé? Několik nápadů na efektivnější ořezávání. Ne u všech her však funguje hrubá síla (minimax) dobře, ukážeme si tedy, jak hru zanalyzovat.

Ďábelské herní algoritmy * *("Pořádně se zamyslet, nebo pořídit výkonější počítač?")* **AIGAME2**
Pavel Veselý
Minimax s α-β ořezáváním nestačí? Pokročilejší techniky na efektivnější ořezávání. Jiné přístupy než minimaxovaný algoritmus aneb Proof-number search a Monte Carlo Tree Search (dnes nejúspěšnější algoritmus pro hraní Go). Jak se poprat s náhodou a jak s tím, že nevidíme soupeři do karet.
Předpoklady: AIGAME

Programovací jazyky a techniky

Programování v jazyce C **C**

Michal Vaner, Karel Tesař, Pavel Veselý, Filip Štědronský, Michal Pokorný

Datové typy jazyka C, programové konstrukce, základy práce s ukazateli. Seznámení se standardními knihovnami jazyka C.

C for wizards * *("`1[x]+++++x[1]`")* **CWIZ**

Martin 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 novou 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.

Objektově orientované programování nejen v C++ *("Object-oriented system. If we change it, users object.")* **OBJ**

Michal Vaner, Michal Pokorný

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. Syntaxe a odlišnosti v jazycích C++, `C#`, Java, Object Pascal, či úplně jiné přístupy v jazycích jako Obj C, Perl, Erlang, ...

Předpoklady: Znalosti procedurálního programování, například v Pascalu, v Pythonu nebo v C.

Černá magie v C++ * *("Je dobré znát, co umí atomová bomba (a její datový typ), abychom ji nechtěli použít.")* **CPP**

Michal Vaner

V C++ jde samozřejmě psát obvyklým způsobem pomocí tříd, polymorfismu a s ruční správou paměti. Ale proč to dělat jednoduše, když to jde složitě? V C++ si můžeme trochu zaprogramovat v době překladu, dělat si seznamy typů, vytvářet lambda třídy, copy-on-write struktury s počítáním referencí... prostě si řekněte, co chcete, napsat to půjde, jen to možná bude práce pro vraha.

Předpoklady: OBJ, TEMPL, staticky alokovaný kyblík

Java **JAVA**

Karolína Burešová

Základy syntaxe, základní typy. Třídy, dědičnost, interface. Práce s objekty, s poli a s řetězci. Povídání o alokaci paměti a garbage collectoru. Zpracování výjimek. Jak na vlákna a jejich synchronizaci.

Programování v jazyce `C#` *("Co se stane, když strčíme Céčko za mříže?")* **CIS**

Honza Škoda, Pavel Veselý

`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 napíšeme i prográmek s grafickým rozhraním.

.NET Framework *("`from p in prednasky where p.Name == ".NET Framework" select p.Content;`")* **DOTNET**

Pavel Veselý

.NET Framework je ohromný soubor technologií umožňující vývoj aplikací pro PC, web, mobily i jiná zařízení. Základní architektura a jak probíhá spouštění programů (aneb Common Language Runtime). Co zajímavého lze najít v knihovnách za funkce. ASP.NET a psaní webových aplikací. Dotazy nad dokumenty XML nebo přímo nad datovými strukturami v paměti podobně jako dotazy nad databází... (zbytek seznamu vymazán šotkem, prý byl moc dlouhý). V neposlední řadě nevýhody .NETu.

Předpoklady: Základy `C#` (CIS), případně jiného objektově-orientovaného jazyka.

Základy psaní pro Android **ANDROID**

Michal Pokorný

Android se v našich mobilech, tabletech, netboocích a ledničkách zabydlel, a asi se v nejbližší době neodstěhuje. Na nějakém jednoduchém zadání si ukážeme, jak se pro něj programuje. Vlastní notebook s nainstalovaným Android SDK se bude hodit.

Předpoklady: Aspoň pasivní znalost Javy.

Generika * *("Má C typovou kontrolu? Ano, ale jen občas.")* **TEMPL**

Michal Vaner

Co je to generická struktura, jak v C napsat spojový seznam, spojovou mřížku, kde se do toho hodí `void *`, dědičnost (ano, v C) a preprocesor. Šablony v C++, aneb neexistuje věc, která by nešla napsat, jen existuje spousta, které se nevyplatí. Jak to řeší jiné jazyky (Java, Haskell, Perl) a jakou za to platíte cenu.

Předpoklady: Přibližná znalost C, C++ a možná dalších, kyblík

Procesy a vlákna * *("Koupil jsem dalších 15 procesorů, proč je to stále stejně pomalé?")* **THREAD**

Michal Vaner, Martin Mareš

Trochu více praktická přednáška o paralelním programování, než PARAL. Jak vypadá víceprocesorové či vícejádrové PCčko a co to znamená pro programátora. Procesy, vlákna a úskalí komunikace mezi nimi. Jak se snese n kohoutů na jednom smetišti? Synchronizační primitiva: mutexy, semafory, podmínkové proměnné. Spinlocky, deadlocky a livelocky. Jde to i bez

synchronizace: atomické operace, transakční paměť. Které jazyky nám pomáhají a které spíš škodí. Kdy je lepší vlákna použít a kdy ne.
Předpoklady: Trochu představy o hardwaru

Perl *("Jak Pejsek a Kočička vymýšleli programovací jazyk")* **PERL**
Martin Mareš, Michal Vaner, Jan Kubálek, Lucie Mohelníková, Jirka Setnička
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. Malé ochutnání Perlu6, jazyka (snad už nepříliš vzdálené) budoucnosti.

Python *("`print "Ffff".decode("rot13")`")* **PYTH**
Michal Vaner, Jirka Setnička, Honza Škoda, Filip Štědronský
Základy programování v Hroznýši (Pythonu), syntaxe, datové (ne)typy, funkce, třídy, moduly aneb všechno je slovník nebo prvek slovníku (nebo oboje). Výhody interaktivního interpretu.

Pokročilé povídání o Pythonu *("`import antigravity`")* **PYTH2**
Filip Štědronský
Povídání o méně zmiňovaných částech Pythonu. New-style classes, dekorátory, metaclasses, generátory, funkcionální styl programování v Pythonu. Jak napsat quicksort jako lambda funkci. Představení zajímavých modulů nejen ze standardní knihovny.
Předpoklady: PYTH

Ruby *("Jak při psaní webů záplatovat opice a mlátit kachny a přitom být stokrát rozumnější než PHP.")* **RUBY**
Michal Pokorný
Ruby je super jazyk: je ohebný a snadno pochopitelný a navíc má okolo sebe skvělý ekosystém. Uvidíte jeho základy a snad i poznáte, proč si ho tolik programátorů zamilovalo.

Ruby on Rails * *("Webíme bez španělských bot")* **RAILS**
Michal Pokorný
V přednášce navazující na Ruby si ukážeme, jak se programují webové aplikace v jednom z nejpoužívanějších frameworků: jednoduché věci budou jednoduché, složité věci budou často jednoduché taky, a nad věcmi, které by měly být automatické, nebudeme muset ani přemýšlet. Hodí se mít ponětí o MVC.
Předpoklady: Ruby, nějaké zkušenosti s weby

Logické programování *("Detektivem za 90 minut.")* **LOGP**
Michal Pokorný, Honza Škoda
Proč psát dlouhé a složité programy, když stačí dostatečně přesně popsat situaci a pak se prostě zeptat? Toť princip logického programování, který si ukážeme na Prologu.

Funkcionální programování *("Naše prášky nemají vedlejší účinky.")* **FUNC**
Michal Pokorný
Nenáročné povídání o tom, co je na funkcionálním programování nové, co skvělé a co méně skvělé. Side-effects a proč je dobré být líný. Práce s funkcemi, specializace funkcí, currying. Jak se v tom všem neztratit aneb mapy. Monády. Jako další chod doporučujeme HASK.

Haskell *("V téhle proměnné je uložen okolní svět.")* **HASK**
Michal Vaner, Jirka Setnička
Základní kurz Haskellu – moderního funkcionálního jazyka. Ukážeme si syntaxi, vysvětlíme typovou kontrolu a typový systém. 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.

Jazyk SQL *("SELECT something FROM knowledge LIMIT 90min")* **SQL**
Jirka Setnička, Karolína Burešová, Honza Škoda, Martin Mareš, Lucie Mohelníková
Jazyk SQL a jeho aplikace. Jak ušetřit skriptu práci a sobě čas, aneb jak se zeptat rovnou na to, co chci vědět. K čemu se hodí složený dotaz a klíčové slovo `JOIN`. Kam až si můžu dovolit zajít, když nevím, na kterém systému to poběží.

Git a jiné systémy pro správu verzí *("U svatýho tučňáka, kdo sem napsal tohle? Ono to tvrdí, že JÁ?!")* **GIT**
Martin Mareš, Michal Vaner, Lucie Mohelníková
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. Kouzelnické triky: hledáme bugy půlením historie, přepisujeme dějiny. Jak se liší správa zdrojáků v projektech o jednom, deseti a tisíci programátorech. Udržujeme patche k cizímu programu aneb quilt a StGit.

Jak se nestat vepřem *("/* You are not expected to understand this */")* **STYLE**
Michal Vaner, Martin Mareš, Karolína Burešová
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` a jak napsat užitečný komentář nebo dokumentaci. A kdy se vyplatí se na všechna tato pravidla vybodnout.

Make *("make love ... don't know how to make love")* **MAKE**
Michal Vaner
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.

Gdb a jiné ladicí nástroje * *("Jak se ladí kytara, jak křišťálová koule a jak program (řazeno dle obtížnosti)")* **GDB**
Michal Vaner, Martin 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 debugger (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**
Martin 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.

High-Performance Computing *("Jak krotit terabyty a jak trilobyty?")* **HPC**
Martin Mareš
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.

Dynamický web a PHP *("Pepíčku, napíšeš mi é-šopík?")* **PHP**
Jirka Setnička, Karolína Burešová, Honza Škoda
Základy praktické tvorby dynamického webu. Úvod do jazyka PHP a Javascriptu, čtení dat z odeslaných formulářů, přesměrování, databáze, generování obrázků a další.

Programování v assembleru **PASM**
Martin 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.

Programování na grafické kartě * *("Řídí se to jako raketa – létá rychle, ale nemá volant.")* **GPU**
Michal Vaner
Dnes již není grafická karta jen placka převádějící digitální pixely na analogový signál. Dá se na ní počítat kde co. Zde si představíme trochu OpenCL a zmíníme, že tento ďábelský kus HW umí počítat zatraceně rychle, ale pokud tam uděláme malou chybičku, tak také zatraceně pomalu. Zmíníme, proč tomu tak je, jaké druhy paměti můžeme v programu používat a co je to multiprocesor.

Jazyková Zoo *("Na co GO TO? Máme COME FROM.")* **JZOO**
Martin 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). Praděděček 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.

Kompilátory * *("Jak se dělají kompilátory (a nebo komplikátory?)")* **KOMP**
Martin Mareš, Michal Vaner
Povídání o tom, jak překladače fungují uvnitř – jak se program parsuje, jak se optimalizuje kód atd. Co je to front end, back end, „middle end", mezikód a jiná arkána umění kompilátorového. Jak psát programy tak, aby kompilátoru chutnaly, co optimalizovat ručně a co naopak udělá kompilátor lépe než my.
Předpoklady: Základní povědomí o tom, co to je procesor a co dělá.

Hardware a operační systémy

Principy počítačů *("A opravdu uvnitř počítače běhají malí trpaslíci?")* **HW**
Martin Mareš, 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.

Vstupní a výstupní zařízení **IO**
Michal Vaner
Co jde k počítači připojit? To nejsou jen klávesnice a myši. Existují různá další udělátka, jako trackbally, tablety (v původním významu slova), volanty, joysticky, mikrofony a kamery. K výstupním se budou řadit bedničky, monitory, tiskárny a třeba i ty volanty. Mezi méně obvyklé i čtečky čárových kódů, infračervené senzory a dotykové tabule. Jak je to celé připojené a jak to funguje v softwaru. Jak fungovaly myši kdysi a jak dnes. Bude i několik praktických ukázek.

Digitální elektronika a hradla **DIGI**
Martin Mareš
Jak fungují digitální elektronické obvody, ze kterých jsou postavené (nejen) počítače. Nuly a jedničky jako napěťové úrovně; kombinační obvody (transistory, hradla, multiplexery), sekvenční obvody (klopné obvody, registry, čítače) a asynchronní obvody. Troška matematiky okolo aneb logické formulky a De Morganovy zákony; proč stačí jenom jeden typ hradel. Třístavová hradla a sběrnice… zde plynule přecházíme v HW.

Filesystémy *("Opravdu je FAT tabulka tlustá?")* **FS**
Martin Mareš
Povídání o tom, co leží mezi nulami a jedničkami na plotnách disku a přátelskou adresářovou strukturou našeho OS. Jak funguje FAT a jeho varianty (VFAT, FAT32). Tradiční Linuxové filesystémy od EXT2 k EXT4. Nadějný nový BtrFS, který je možná za pár let nahradí. Co se hodí na SSD.

UNIX *("UNIX gives you enough rope to hang yourself.")* **UNIX**
Martin Mareš, Honza Škoda, Filip Štědronský
Kamarád u černobílého textového okna září blahem. Chcete poznat, proč? Jak UNIX vznikl, k čemu je dobrý a k čemu třeba není. UNIXová filosofie. Kouzlo skriptů. Kouzlo speciálních souborů. Kouzlo propojování programů. Kouzlo nechtěného. UNIX byl napsán v C a C vzniklo pod UNIXem.

Skriptování v shellu *("man 1 woman … man 2 woman … man group")* **SHELL**
Honza Škoda, Filip Štědronský
Spíše motivační přednáška pro ty, kteří shell ještě neviděli, případně jen z dálky. Ukážeme si na spoustě příkladů, jak nám může automatizace všedních činností ulehčit život a jak silné nástroje pro ni unixový shell (který navzdory svému názvu existuje i pro Windows) svou jednoduchostí a flexibilitou poskytuje. Budeme přejmenovávat epizody (legálně ;-)) stažených seriálů dle názvů z Wikipedie, za deset minut zbastlíme zvukem řízenou samospoušť či ušetříme několik set korun za učebnici angličtiny. True story. Některé činnosti vyžadují lidskou nápaditost a vhled. Ty ostatní bychom měli přenechat strojům.

Programování v Linuxu **PLX**
Michal Vaner, Martin Mareš, Filip Štědronský
Jak si program pod Linuxem povídá s operačním systémem, když chce otevřít soubor, přečíst soubor, půjčit trochu paměti a jiná šprťouchlata. Předvedeme si, jaká existují v Linuxu systémová volání. Naučíme se namapovat si soubor rovnou do paměti, posílat a odchytávat signály, uspávat a probouzet proces, plodit děti a další. Pokud zbyde čas, můžeme si napsat démona a klienta a povídat si po síti.
Předpoklady: Schopnost přečíst a napsat jednoduchý program v C.

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

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

Cache-oblivious algoritmy *("Kešuješ, kešuje, kešujeme")* **CACHE**
Martin Mareš, Michal Vaner
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

Mobilní zařízení *("Co je malé, to je hezké, a když ne, tak toho aspoň není moc.")* **MOBI**
Jirka Setnička
Jak se liší PDA, MDA, Smartphone a obyčejný mobilní telefon. Jak vlastně takový mobilní telefon funguje a jak vypadá struktura základnových stanic dovolující, aby fungovat mohl. Spíše hardwarový přehled toho, co vlastně v dnešním telefonu je a co to umí – dotykové technologie, bezdrátové sítě, vlastnosti různých baterií, metody šetření elektřinou. Diskuze o třech (čtyřech) hlavních platformách dneška a ukázka konkrétního využití, aneb s telefonem se dá mnohem více, než jen telefonovat.

Sítě a bezpečnost

Sítě a Internet *("Sítě nejen na ryby.")* **NET**
Martin Mareš, Michal Vaner, Honza Škoda
Jak funguje Internet a počítačové sítě vůbec: od elektronů v drátech (fotonů v optických kabelech nebo elektromagnetických vln) přes packety a jejich routing až k jednotlivým síťovým službám. Adresace, internetworking a dynamický routing. Jak NAT zachránil i zničil Internet a proč se těšíme na IPv6.

Sítě II – aneb aplikační protokoly TCP/IP *("Pokud jste se zamotali do sítí, tak se vás pokusíme vymotat.")* **NET2**
Martin Mareš, Michal Vaner, Honza Škoda
Tato přednáška navazuje na „Sítě a Internet" a zaměří se na konkrétní aplikační protokoly nad TCP/IP. Zajímá vás, jak funguje web, pošta, DNS, FTP, nebo třeba Jabber? Poodhalíme roušku tajemství těchto protokolů a když zbude čas, přidáme ještě třeba SIP (protokol pro internetovou telefonii).
Předpoklady: NET

Web uvnitř *("Error 402: Payment Required. Please insert a coin.")* **HTTP**
Martin Mareš, Jirka Setnička
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 protokol CGI. Mezi klientem a serverem aneb DNS a virtuální servery. Nakonec do toho všeho přimícháme SSL/TLS a máme HTTPS.

E-mail *("Drahoušek zákazník.")* **EMAIL**
Michal Vaner
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, viry, phishing, BFU a kde koupit levnou viagru. Nakonec státní datové schránky a proč je to zlý ošklivý nepěkná věc. A jak se správně podepsat. A jako bonus sociální síť, ve které je každý a ani o tom neví.

DNS *("Neviditelný stín v pozadí Internetu")* **DNS**
Michal Vaner
DNS je starý, a přesto moderní protokol. Stojí na něm infrastruktura celého Internetu. Slouží k překladu adres, ale nejen k tomu. Jak zajišťuje spolehlivost, jak bezpečnost. A proč si na něj ani Anonymous netroufnou. V čem je ČR první a co je to digitální lukostřelba.

Kryptologie *("Gbgb arav zbp gnwan mcenin.")* **CRYPT**
Martin Mareš, Jirka Setnička
Kryptologie čili tajuplná nauka o šifrách, jejich konstrukci a hlavně o jejich luštění. Přísně tajné. Šifrovací systémy jako lego: základními kostičkami nám budou symetrické a asymetrické šifry a jednosměrné funkce, 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.

Kryptologie II * *("6140 a184 c9a6 41f1 de99 e733 354a f451")* **CRYPT2**
Martin Mareš
Pokročilejší (dešifruj: zběsilejší) partie vědy kryptologické: utajené výpočty, zero-knowledge proofs, sdílení tajemství, podprahové informace a kvantová kryptografie. Aplikace v reálném životě: digitální peníze, volební systémy. Různé metody útoků na šifry a kryptografické protokoly. Problémy distribuce klíčů a proč se jí raději vyhnout (a jak: Diffie-Hellman key agreement, komutativní šifry). Stručný přehled souvisejících partií matematiky a teorie složitosti.
Předpoklady: Základní povědomí o šifrování (CRYPT) a víra v existenci náhodných čísel

Síťová (ne)bezpečnost *("Kdo si čte vaše emaily a jak?")* **NEBEZP**
Honza Škoda, Jirka Setnička
Stručný úvod do fungování sítě. Praktické ukázky hackování běžné sítě od MITM po převzetí (třeba emailového) účtu. Pomůže nám HTTPS? Jak je to na WiFi? Zastaví nás heslo? Na jak dlouho? A konečně – jak se tomu všemu lze bránit, pokud to vůbec jde.

Počítačová bezpečnost prakticky * *("K čemu je 4096bitový klíč, když si ho napíšu na papírek pod klávesnici?")* **NSA**
Michal Vaner
Nejslabším článkem bezpečnosti téměř vždy bývá člověk. Na co si dát pozor, když si chráníte data a když posíláte zprávu. Co je to trust model a komu můžete (ne)věřit. Jak udělat bezpečné přihlášení. Co dělat ve chvíli, kdy jsou nějaká data kompromitovaná. A proč zamykat dveře a školit zaměstnance, aby za sebou zavírali a nikoho nepouštěli. Kam data neukládat a jaká data neukládat vůbec.
Předpoklady: Paranoia

Sledování sítě * *("Nedivte se, že to padá, všichni koukaj na Dalas.")* **SCAN**
Michal Vaner
Jak zjistit, co vám uživatelé dělají na síti. Jak chytit spamera, jak odhalit DDOS, a co jsou jen živelné katastrofy. Jak to dělat doma, v malé síti a jak na 100GBit lince. A co si ještě může administrátor dovolit sledovat v ČR a co v Americe.
Předpoklady: NET, tušení, co je statistika.

Real-time osobní komunikace po Internetu *("Zítra v 5 v matfyz@conf.netlab.cz")* **VOIPIM**
Michal Vaner
Probereme sbírku různých komunikačních systémů, které usnadňují život na internetu. Jak si posílat zprávičky, počínaje starým dobrým talkem, přes IRC až po XMPP (to, na čem běží všechny jabbery, google talky i facebook chaty). Přejdeme i k telefonování, zmíníme SIP, Skype a Jingle. Jak to funguje uvnitř? Čemu se vyhnout, pokud chceme komunikovat i za 20 let, co se dá odposlouchávat a k čemu se hodí otevřenost? A proč to tak moc vadí telekomunikačním společnostem?
Předpoklady: Základní povědomí o fungování počítačových sítí

XMPP v dobrém i zlém *("Rozšíření pro rozšíření pro rozšíření")* **XMPP**
Michal Vaner
V čem je protokol XMPP dobrý a v čem špatný. Ukážeme si jeho použití pro instant messaging, synchronizaci kde čeho, sociální sítě či ovládání zařízení na dálku. Jak přenést soubor a jak zvuk, co je to XEP a proč jich je tolik. A jakým chybám se při návrhu podobného protokolu vyhnout. A jak je to celé zabezpečené.
Předpoklady: Hrubá znalost XML.

Grafika a typografie

Počítačová grafika *("Namaluj mi beránka...")* **GFX**
Martin Mareš, Pavel Veselý
Kreslení a zpracování obrazu na počítači. Souřadnice (rovinné, prostorové i barevné) a jejich transformace. Základní grafická primitiva: body, úsečky, kružnice, elipsy, Bézierovy křivky a jejich rasterizace. Vyplňování n-úhelníků a křivkou ohraničených oblastí, flood fill. Pár triků navíc: maticové filtry, anti-aliasing a dithering. Grafické formáty a komprese obrázků. Základy trojrozměrného promítání a vykreslování scény.

Geometrie a počítače *("Nerušte mé kruhy! (ani jiné kvadriky)")* **GEOM**
Martin Mareš, Jirka Setnička, Karel Tesař
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.

Barevné systémy *("Co je na konci duhy?")* **COLOR**
Martin 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.

PostScript a PDF *("Vy obrázky malujete? To my je programujeme...")* **PS**
Martin Mareš
Jemný úvod do jazyka určeného k tisku grafiky a textu. Základní principy, řídící konstrukce a datové struktury, cesty a kreslení objektů, transformace souřadnic, DSC komentáře. PDF (Portable Document Format) coby mladší a deklarativnější bráška PostScriptu. Různé druhy fontů (např. Type1, TrueType) a jak fungují.

MetaFont, MetaPost *("Teď ten obrázek takhle zkroutím a pak ho přeložím.")* **MF**
Lucie Mohelníková
Lehké nakousnutí jazyka, ve kterém můžete opravdu kreslit planimetrické obrázky, ale i třeba písma nebo piktogramy do zadání a řešení KSP. Jak vypadají CM fonty (ty, které používá TEX) a jak se autorovi povedlo, že se z jediného „obrázku" dá vygenerovat tlusté, tenké, rovné, skloněné, šišaté písmenko.

Typografie *("What You See Is all What You've Got!?")* **TYPO**
Martin 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**
Martin Mareš, Lucie Mohelníková, Jirka Setnička
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.

OpenGL * *("Je libo krychličku nebo díru do ní?")* **GL**
Michal Vaner
Základy OpenGL, jak vytvořit 3D svět. Kreslení trojúhelníků a jiných úhelníků. Barvy, textury a světlo. Průhlednost a triky s ní. A triky bez ní. Jak udělat žulový náhrobek s vyrytým reliéfním nápisem na 6 obdélníků a hladkou kouli na 16. Jak udělat oheň či vodotrysk jako živý. Náznaky, co vše může dnešní grafická karta dělat a jak moc jsou výrobci her lemry líné.
Předpoklady: GFX

Čárové kódy *("Jak naučit počítače číst láhve od Coly")* **BAR**
Martin 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, ...

Teoretická informatika

Složitější složitost * **SLOZ2**
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: SLOZ

Třídy složitosti * **SLOZ3**
Martin Mareš, Michal Vaner, Pavel Veselý
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

Modely počítačů *("Nač Pentium? Máme Turingovy stroje!")* **MODEL**
Martin Mareš, Michal Vaner
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. Rekursivní funkce, přepisovací pravidla a Minského registrové stroje. Paralelní verze klasických modelů, buněčné a grafové automaty. Trocha esoteriky: reverzibilní algoritmy a dlaždičky v koupelně.

Vyčíslitelnost ** *("S Halting problémem na věčné časy!")* **VYCIS**
Martin Mareš, Pavel Veselý
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.

Omezené třídy grafů ** *("Nejdelší cesta ve stromu? A co je za problém?")* **OTG**
Michal Vaner, Karel Tesař, Pavel Veselý
Některé problémy nad grafy jsou těžké, ale pokud si omezíme grafy, které můžeme dostat, hned je to jednodušší. Co je to graf omezené stromové šířky, a jak na něm najít hamiltonovskou kružnici v polynomiálním čase. Silně teoretická přednáška.
Předpoklady: GA, SLOZ2

Logické úlohy a informatika **LOGINF**
Karel Tesař
Povíme si, jak logické úlohy souvisí se základy informatiky. Ačkoliv to tak na první pohled nevypadá, tak ve většině logických úloh jsou skryty informatické postupy, jako je například kódování informací, hledání správné cesty nebo splňování podmínek.

Pravděpodobnost a algoritmy *("Nejen že Bůh hraje v kostky, ale ještě při tom občas švindluje!")* **PPALG**
Martin Mareš
K čemu jsou při programování dobrá náhodná čísla a jak je generovat. Algoritmy pravděpodobnostní a randomizované, průměrná časová složitost a její koncentrace. Míchání karet. Proč používat a proč nepoužívat Quicksort. Inkrementální algoritmy (třeba na konvexní obal), vyhledávání v poli v konstantním čase za pomoci hešování, konstrukce perfektního hešování, randomizované datové struktury (skip listy a treapy). Interaktivní protokoly aneb jak vyhrát nad falešným hráčem.

Paralelní výpočty *("Největším nepřítelem lidstva je trojrozměrný prostor.")* **PARAL**
Martin Mareš, Michal Vaner
Když nestihne problém vyřešit jeden procesor, proč jich nepoužít víc? Zkusme na chvíli zavřít oči a představit si, že máme stroj, který umí například pro sečtení N čísel zapnout N procesorů... nebo rovnou N^2, všechny se společnou pamětí a společným programem – teoretikové takovému počítači říkají PRAM. Ukážeme si rychlé paralelní algoritmy všeho druhu: aritmetiku, slévání a třídění, grafové algoritmy, vše v (poly)logaritmickém nebo dokonce konstantním čase. Po probuzení do reality všedního dne trocha praxe: SMP, NUMA, Connection Machine, clustery, koordinované screen savery, FPGA.

Programování s omezujícími podmínkami *("Celé prázdniny budu plánovat a řešit sudoku.")* **CSTR**
Michal Vaner
Trochu jiný přístup k obtížným úlohám. Některé úlohy sice vypadají, jako by se za dobu existence vesmíru nedaly vyřešit, nicméně pro rozumně velké vstupy to přesto potřebujeme. Jak backtrackovat rychleji a radostněji – backjumping, backmarking, limited discrepancy search, a jak neprobírat úplné nesmysly – hranová konzistence, konzistence po cestě, bodová konzistence.

Jazyky, gramatiky a automaty * **AUTO**
Martin Mareš, Michal Vaner, Karel Tesař, Jirka Setnička, Michal Pokorný
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

Logika *("Tato věta sem nepatří.")* **LOGI**
Martin 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

Pravděpodobnost *("Většina lidí má nadprůměrný počet rukou.")* **PP**
Martin Mareš, Karel Tesař, Lukáš Folwarczný
Toto je přednáška o základech teorie pravděpodobnosti a statistiky. Dozvíte se, co to je podmíněná pravděpodobnost, rozdělení, střední hodnota nebo rozptyl, jak se to všechno počítá a k čemu je to dobré. Součástí přednášky bude i několik zajímavých příkladů z praxe a krátký kurs přežití ve světě plném chybných statistik.

Úvod do Ramseyovy teorie * *("Dejte mi dostatečně velký objekt a já v něm najdu nějaký řád.")* **RAMS**
Martin Mareš, Jirka Setnička, Karel Tesař, Pavel Veselý, Lukáš Folwarczný
Hříčka: ve společnosti šesti lidí vždy existují tři, kteří se navzájem znají, nebo neznají (ověřte ručně). Obecněji, pro libovolné „tři“ existuje „šest“ tak, že shora uvedené tvrzení platí. To je jedna z Ramseyových vět, které říkají, že v každém dostatečně velkém objektu vždy existuje nějaký stejnorodý podobjekt. Jednoduchá tvrzení Ramseyova typu, Ramseyova věta pro grafy dvou a více barev, pro systémy p-tic, nekonečná verze a aplikace. Populárně řečeno, chaos to má těžké.

Teorie množin a matematika nekonečen * *("Je Vlk nedosažitelný kardinál?")* **TEMNO**
Martin Mareš, Lukáš Folwarczný
Teorie množin tvoří páteř veškeré matematiky. Pomocí množin se totiž modelují veškeré objekty, které se v matematice vyskytují. Celou teorii prostupuje magický pojem *nekonečno.* Jakým způsobem se tohoto, pro spekulativní mysl ošidného, termínu zhostila moderní matematika? Množiny a jejich velikosti. Cantorův diagonální trik. Ordinály a houšť kardinálů. Potenciální kontra aktuální nekonečno. Myslíte si, že máte dobrou představu o tom, co jsou přirozená čísla? Možná vás z ní vyvedeme. A co teprve reálná čísla. Problematika volby axiomů determinovanosti versus výběru.

Základy algebry **GALG**
Michal Pokorný
Hledáme eleganci každodenní matematiky. Množina s jednou binární operací a přirozenými axiomy: od grupoidu přes monoid a pologrupu k celogrupě. Počítání modulo x aneb cyklické grupy. Eulerova funkce φ a proč jsou prvočísla krásná. Dělitelnost jak ji neznáme a Euklidův algoritmus. Plus mínus krát rovná se okruh, okruh plus děleno rovná se těleso, a těleso je buď nekonečné, nebo elegantně jednoduché. Obecná algebra: zobecnění zobecnění. RSA.

Grafy bez algoritmů **GRAFY**
Martin Mareš, Lucie Mohelníková, Jirka Setnička, Karel Tesař, Pavel Veselý
Teorie grafů trochu teoretičtěji. Různé druhy grafů a jejich vlastnosti. Stromy a lesy. Kreslení grafů jedním tahem. Princip sudosti a skóre grafu. Jak poznat, že dva grafy (ne)jsou isomorfní. Mosty, artikulace a ušaté lemma. Párování, střídavé cesty a Hallova věta.

Barevnost grafů * *("Bílá, modrá, červená, co to pro graf znamená?")* **BAGR**
Michal Vaner
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**
Martin Mareš, Jirka Setnička, Lucie Mohelníková, Karel Tesař, Pavel Veselý
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.

Lineární algebra **LA**
Martin Mareš, Lucie Mohelníková, Jan Kubálek, Karolína Burešová
Lineární algebra původně vznikla jako elegantní prostředek k popisování geometrie lineárních útvarů (bodů, přímek, rovin, ...) v libovolněrozměrném prostoru, ale ukázalo se, že její kouzlo dosahuje daleko dál. Vektorové prostory, lineární (ne)závislost, báze, lineární zobrazení a matice, determinanty, tenzory. Konečné projektivní roviny.

Lineární algebra pro pokročilé **LAFA**
Jan Kubálek
Volně navážeme na LA. Ukážeme si, co je to spektrum nebo nilpotentní matice. Proč lze ke každé čtvercové matici najít podobnou matici, která bude horní trojúhelníková nebo proč každý matfyzák snídá koláče upečené v bilineární formě. Nebo snad chcete vědět, jak Rusko málem zničilo planetu setrvačníkem?
Předpoklady: Doporučené jsou základní znalosti z LA.

Lineární programování ** **OPT**
Lucie Mohelníková, Karel Tesař, Honza Škoda
Řešení problémů pomocí soustavy lineárních nerovnic (lineárním programováním). Naučíme se tak řešit některé grafové, ale i jiné, problémy. Řekneme, jaké metody se k tomu používají a jak jsou efektivní oproti běžnému přístupu.
Předpoklady: LA

Teorie (vesměs samoopravných) kódů *("f y cn rd ths, y wll b gd cmptr prgrmmr!")* **KODY**
Martin 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?

Komplexní a komplexnější čísla *("$1 = \sqrt{1} = \sqrt{(-1)(-1)} = \sqrt{-1}\sqrt{-1} = i \cdot i = i^2 = -1$. Huh?")* **CPLX**
Martin Mareš, Jan Kubálek, Lucie Mohelníková, Lukáš Folwarczný
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 teorie čísel *("Po malém fermetu mívám čínský zbytkáč.")* **NUT**
Martin Mareš, Karel Tesař, Lucie Mohelníková, Karolína Burešová
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.

Teorie čísel a RSA * *("$2^{67} - 1 = 193\,707\,721 \cdot 761\,838\,257\,287$")* **NUT2**
Martin Mareš, Karolína Burešová
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.

Prvočíselné věty * **NUT3**
Martin Mareš, Lukáš Folwarczný
Věty o rozložení prvočísel jsou tradičně považovány za jednu z nejmysterióznějších oblastí teorie čísel. Zde ukážeme, jak některé z nich odvodit snadným pozorováním vlastností kombinačních čísel: rozbor časové složitosti Eratosthenova síta, Bertrandův postulát („Mezi n a $2n$ je aspoň jedno prvočíslo."), hustota prvočísel.

Fourierova transformace * **FFT**
Martin Mareš
Chytrý trik pana Fouriera patří již dávno k matematické a fyzikální klasice. Převapivě se ale hodí i při programování: rychlé násobení polynomů a dlouhých čísel (dokonce v lineárním čase), digitální zpracování zvuku a obrazu (spektrální analýza či třeba komprese).
Předpoklady: Základy komplexních čísel (CPLX)

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

Vytvořující funkce **VYTF**
Karel Tesař, Lukáš Folwarczný
Vytvořující funkce jsou silný nástroj pro zjišťování počtu možností, obvykle daných nějakou rekurencí, a tvorbu vzorečků. Základní myšlenkou je představit si polynom, jehož koeficienty jsou hledané výsledky, a z něj pak odvodit vytvořující funkci ve tvaru, ze kterého dokážeme zkonstruovat vzoreček. Tímto způsobem můžeme například odvodit vzoreček pro Fibonacciho či Catalanova čísla a mnohé další.

Teorie kombinatorických her *("Život je jen hra... Jakou má vyhrávající strategii?")* **GAME**
Pavel Veselý
Rozličné kombinatorické hry se zápalkami, kamínky, barvičkami či grafy. U některých si ukážeme výherní či obranné strategie, u některých dokážeme, že příslušná strategie existuje, i když nevíme, jak vypadá. Jak hry rozkládat a zase sčítat a jak je ohodnocovat čísly dle výhodnosti pro jednoho či druhého hráče. Zmíníme například: všelijaké piškvorky, různé varianty Nimu, dominování, a další.

Teorie nemožného * *("Neexistence důkazu není důkazem neexistence. Dokažte.")* **NONEX**
Martin Mareš
Existenci slona v Africe snadno dokážete tím, že ho přivedete. Jak ale ukázat, že tam žádný slon není, případně že sice je, jenže ho nejde najít pomocí pravítka, kružítka a jeepu? Přímo se to dělá těžko, ale existuje spousta krásných triků, jak neřešitelnost problémů dokazovat. Nesložitelné hlavolamy, nerozvázatelné uzly, nepopsatelná čísla, neroztřetitelné úhly, nealgoritmické problémy a jiné slasti nekonstruktivní matematiky. Jak naopak ukázat, že něco existuje, aniž bychom věděli, jak to vypadá?

Úvod do klasické analýzy * *("Dej pokoj a už mě konečně zinfinityzuj")* **KA**
Jan Kubálek
Limita, derivace, derivace^{-1}, integrál. Z pohledu formální matematiky velmi podobné si jsou. My si ukážeme, jak jednoduše lze tyto pojmy pochopit a že vyzrát na ně není vůbec těžké.

Deskriptivní geometrie *("Jak splácnout tři rozměry do dvou")* **DG**
Jirka Setnička
Jemný úvod do Mongeova promítání a axonometrie. Jak nakreslit krychli aby vypadala „jako živá", jak narýsovat na list papíru dvě navzájem kolmé roviny a poznat, kde se protínají a spousta dalších zajímavých konstrukcí.
Předpoklady: Prostorová představivost výhodou, pravítko a kružítko rovněž, koulítko a rovinítko netřeba.

Kombinatorické struktury * *("BIBy, BIBDy a jiná chlupatá zvířátka")* **KST**
Lucie Mohelníková
Kombinatorika v sobě skrývá různa (ne)tradiční schémata, o jejichž použitelnosti bychom se mohli přít hodiny a hodiny. Ukážeme si, co jsou to latinské čtverce, projektivní roviny a jak souvisí s BIBDy (bloková schémata). Dokážeme si větu, kterou budeme mít problém si vůbec zapamatovat, a povíme si něco o Steinerových systémech trojic. Přednáška bude hodně matematická a absolutně nepředstavitelná.

Ostatní přednášky

Lingvištika *("Přísudek je v této větě podmět.")* **LING**
Martin 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. Jak se jazyky vyvíjejí a jak se navzájem ovlivňují. Kde jsme přišli k pravidlům a jaký je jejich smysl. Existují synonyma? Proč je jazyk nejednoznačný a proč je to dobře. Jak se na jazyk dívá matematik a jak se na matematiku dívají lingvisté. Jak vzniklo písmo? A jak otazník? Jak zapsat zachrochtání a jak třeba mlasknutí &c.

MFF UK aneb co obnáší matfyzákem býti *("Maminko, ptá se tatínka, kdy už budu matfyzákem?")* **MFF**
Nezávazné povídání o Matfyzu a základním matfyzáckém folkloru. Určitě si přečteme matfyzáky sepsaný Úvod do matfyzáka a zazpíváme pár matfyzáckých písní. Zbytek už bude záležet na tom, co budete chtít slyšet.

Matfyzácké vtipy **MFV**
Karel Tesař
Vše o matfyzáckých vtipech a jejich rozdělení do kategorií. Které z nich jsou ty pravé matfyzácké? U kterých zas stačí jen dosadit jméno své školy a pořád zůstanou vtipné? A které vtipy pochází od normálních lidí a snaží se akorát matfyzáky pomlouvat?

Sbírka brouků *("No keyboard found. Press any key to continue.")* **BUG**
Michal Vaner
Neuspořádaná přehlídka, jaké chyby kdo kdy udělal, kdy se z nich stal stal standard a jak se to komu povedlo využít. Jak některá taková chyba stála miliony dolarů, či dlouhé hodiny hledání. Očekává se přispění posluchačů.

Nečisté obchodní praktiky *("Pokud odejdeš, pošleme ti z tvých dat napřed malíček, potom prsteníček. . . ")* **SCUM**
Michal Vaner
Některé programy a služby na internetu jsou udělané tak, aby z nich měl uživatel užitek. Některé také tak, aby z nich měl užitek autor. A některé tak, aby z nich měl užitek jen autor a uživatele pokud možno odrbal. Mezi ně zajisté patří různé viry a trojské koně, ale také mnoho legálních věcí. Jak zavírá uživatele do ohrádky takový Microsoft či Apple, jak takový Google a Facebook?

Orientace **ORI**
Martin 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.

OpenStreetMap *("Mapa, ve které je i má oblíbená lavička")* **OSM**
Michal Vaner, Martin Mareš
Dříve se ke kreslení map používaly pastelky, dnes k tomu lze použít také počítač. A v době internetu to i sdílet. Co se stane, když pustíme desetitisíce lidí, kteří začnou kreslit zároveň? Co použít k zaznamenání vlastního domu, kde sebrat řeky a jak z toho nakonec udělat turistickou mapu či autonavigaci? Aneb hračka technologického nadšence do terénu.

Čaj *("Jak vypadá odvar z nezralých pražců?")* **TEA**
Martin 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.

Programování v týmu *("Přiznejte se, kdo z vás ten bug neopravil!?")* **TEAM**
Michal Vaner, Martin Mareš
Když je program na jednoho člověka moc velký, začne na něm pracovat lidí více. Tato přednáška bude pojednávat o tom, jak takový tým lidí uřídit, od dobrovolnických neřízených projektů, po hierarchie. Které nástroje se k tomu hodí a co dělat, když je tým rozložen do několika časových pásem. A co plyne z toho, že devět žen neporodí dítě za jeden měsíc?

Email Best Practices *("Proč si výhru nevyzvedl, když na něj blikala, svítila a pípala?")* **EBP**
Michal Vaner
Jak správně komunikovat emailem? Stejně jako papírové dopisy, i ty elektronické mají nějakou kulturu a měly by dodržovat základní štábní kulturu. Jaká pravidla dodržovat a co naopak nikdy nedělat, aby se emaily příjemci dobře četly? Jak se vyznat v stovkách emailů denně a nezbláznit se z toho? Jak psát známým, jak obchodním partnerům, a jak na konferenci? A co dělat, aby příjemce email okamžitě zahodil? Na tyto otázky si odpovíme a předvedeme i několik ukázek.

Základy první pomoci *("Jak někomu zachránit život a jak málo k tomu stačí")* **ZDRAV**
Jirka Setnička, Karolína Burešová
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.

Svět hazardu *("Mám šanci obrat kasíno?")* **HAZARD**
Karel Tesař
To, že kasína ještě nezkrachovala, není jen tak samo sebou. Ukážeme si, jaké hry se v kasínech hrají, jak jsou které (ne)výhodné a u kterých svítá naděje na malou pravidelnou výhru. Také si povíme, kdo a jak v minulosti kasínu rozbil bank a jak se jim to povedlo a jak by měli závislí hazardéři hrát, aby neprohrávali moc rychle. Také se trochu zasmějeme nad tím, jakou zvláštní mythologii používají majitelé a manažeři kasín při své práci.

Předmětové olympiády od A do Z **SOUT**
Karolína Burešová
České předmětové olympiády z pohledu soutěžícího i nezávislého pozorovatele. Jak se dostat do celostátního kola, jak (možná) dojít až do mezinárodní olympiády a která cesta vede zaručeně do pekel. Příspěvek ze strany korespondenčních seminářů, aneb zapomeňte školní znalosti, ty vám nepomůžou. Nečekejte univerzální rady, neb žádné takové neexistují, spíše vyprávění o cestě obyčejného smrtelníka olympiádním molochem.

Auto z pohledu technika *("Co mi to vrčí pod kapotou a proč bliká ta kontrolka?")* **CAR**
Jirka Setnička
Nahlédneme do tajů starších i novějších aut. Podle zájmu se můžeme pobavit o tom, jaký je rozdíl mezi benzínovým a naftovým motorem. Na praktické ukázce probereme auto z pohledu běžné údržby a jednoduchých oprav. Na co si mohu troufnout sám a co bych měl raději svěřit servisu. Praktické typy, jak se chovat ke svému autu a mnoho dalšího podle zájmu.

Mapování mysli a jiné techniky *("Někdo mapuje terén, my mapujeme mysl")* **MIND**
Pavel Veselý
Mapy mysli (neboli myšlenkové mapy) jsou speciální diagramy, které mají v člověku povzbudit kreativitu. Kromě vymýšlení všemožných věcí od přednášky po výpravu na Mars je lze využít pro organizaci myšlenek a při učení. Na příkladě si nějakou mapu mysli vytvoříme a představíme si program na jejich tvorbu. Dále je v nabídce: brainstorming a jak ho dělat efektivně, i když jste sami, metoda GTD (Getting Things Done) pro zvýšení produktivity práce, duševní rozcvičky, nasazování klobouků a další kreativní techniky.

Běžecké praktikum *("Pojďme si zaběhat!")* **BEH**
Lukáš Folwarczný, Pavel Veselý
Běh je spolu s chůzí nejpřirozenějším sportem. Pojďme se si ho prakticky vyzkoušet v okolí objektu dle aktuálních časových a silových možností, ale určitě poklidným tempem. Můžeme se také pobavit, kde běháme, jakých jsme se zúčastnili závodů nebo o nových trendech v běhání (třeba o minimalismu pod vlivem knížky Born to run).

Technika ve sci-fi *("Scotty, beam me up")* **SCIFI**
Jirka Setnička
Technické objevy v různých sci-fi (Star Trek/Gate/Wars i dalších) a pohled na ně z perspektivy dnešních fyzikálních znalostí. Proč se v Hollywoodských filmech ozývá ve vesmíru zvuk laserů, i když je tam vakuum, a je možné cestovat rychleji než světlo? Také možná zabrousíme do některých filozofických otázek – primární směrnice o nevměšování ve Star Treku a jiné.

Fyzikální přednášky

Podzimní obloha **SKY**
Martin Mareš
Pozorování podzimní hvězdné oblohy spojené s astronomickým minikursem. Od antických a ještě starších bájí k modernímu příběhu o Velkém Třesku a naopak od celkem seriózní vědy k rozmarnému filosofování o světě a našem místě v něm. Hvězdáři a hvězdopravci, „Už staří Řekové…“, měření a vážení na dálku, vývoj hvězd a kosmologie, antropický princip, kdo schvaluje fyzikální zákony? Jak se podle hvězd orientovat a jak fungují sluneční a třeba i měsíční hodiny.
Předpoklady: Počasí dovolí. Měsíc nejlépe v novu.

Keplerovy zákony trochu jinak * *("aneb jak dostat kosmonauta na měsíc")* **KEPP**
Jan Kubálek
Jsou Keplerovy zákony přesné, nebo se jedná o aproximaci? Ukážeme si, jak odvodit Keplerovy zákony z čistě matematického základu, jak vypočítat dráhu rakety letící k měsíci nebo proč planety obíhají po elipsách.
Předpoklady: Znát pojem derivace (KA) a vektoru (LA).

Elektřina a magnetismus * **ELMAG**
Jan Kubálek
Tak jak se učí na matfyzu. Co to jsou Maxwellovy rovnice, jak vznikly a co znamenají všechna ta písmenka kolem.
Předpoklady: Znát pojem derivace (KA) a vektoru (LA).

Abecední seznam přednášek