Programų kūrimo metodologija.

Šis kursas galėtų būti prerekvizitas tiems kursams, kuriuose praktikos darbų metu studentai turi parašyti savo programas ar skriptus ir jas pritaikyti. Kursas dėstomas naudojant Linux kaip demonstracinę terpę (sistemą), ir studentai tuo pačiu įgys elementarios patirties dirbdami su šia sistema, bet išdėstyti principai bus pritaikomi bet kokioms operacinėms sistemoms, kalboms ar aplinkoms.

Prerekvizitas – turėti patirtį rašant paprastas programas ir dirbant su kompiuteriais kokioje nors OS; žinoti kompiuterių architektūros pagrindus; žinoti bent vieną procedūrinę programavimo kalbą.

Kurso turinys

1. Unix(R) istorija, dabartis ir paplitimas, architektūros principai (viena programa == viena funkcija, programų kompozicija, minimalus funkcionalumas branduolyje), šiuolaikinės Unix tipo sistemos: Linux, *BSD, Solaris, ... Laisvos/Atviro kodo ir nuosavybinės sistemos. Unix įtaka paplitusioms šiuolaikinėms sistemoms (MacOS X, DOS/Windows).
2. Darbas Unix sistemoje: terminalas, prisistatymas (login), darbo pabaiga (logout), komandų/ komandinė eilutė (command line), pakvietimas (prompt), komandų interpretatorius (shell). Komanda ≡ programa. Komanda `which`. Komanda `locate`. Komanda `whoami`. Specialūs failų vardai: "/", ".", "..". Specialūs komandų interpretatoriaus `sh` (shell) simboliai ("*", "?") ir jų išplėtimas. "Keisti" failų vardai ("-", " ", "...", vardai su tarpais ir kabutėmis, vardai su specialiaisiais interpretatoriaus `sh` simboliais) ir darbas su jais.
3. 0x10 Unix komandų (įsakymų ;) – `ls`, `cd`, `cp`, `mv`, `rm`, `echo`, ... Teksto redaktoriai: `ed`, `vi`, `nano`, `nedit` (Nirvana editor), `emacs`. Komandų argumentai – failų vardai ir neprivalomi parametrai (options). Parametrų sintaksė, istoriniai variantai. Komandų aprašymai: --help parametras ir man komanda. GNU info. Patogus darbas komandinėje eilutėje: Tabuliacijų išplėtimas, sutrumpinimai (`~`, `~name`), istorija, sinonimai (alias). Gerai žinomi sinonimai (well-known aliases). Atsargumas su `rm`, `cp` ir `mv` komandomis.
4. Standartiniai įvedimo/išvedimo (įvesties/išvesties) kanalai – `stdin`, `stdout`, `stderr`. Kanalų nukreipimas. Programų kompozicija, konvejeriai (Unix pipes). Filtrai. Komandos `grep`, `tr`, `cut`, `paste`. "Miniatiūrinės" programavimo kalbos filtrams: `sed`, `awk`, `perl`.
5. Komandų interpretatorius `sh` (shell) – eilinė Unix programa. Interpretatoriaus skriptai (scenarijai, angl. scripts). Intrepretatoriaus `sh` programavimo kalba: kintamieji, parametrai, valdymo operatoriai (`if`, `while`, `for`, `case`, `test`, `goto`). Interpretatorius – eilučių pakeitimo kalba (string substitution language) ir jos ypatybės. Savo skriptų ir mažų programų kūrimas. Specialūs Shell simboliai ir jų išplėtimas. Kaip komandų interpretatorius ieško komandų-programų, kintamasis PATH. Aplinkos kintamieji (environment variables).
6. Unix failų sistema. Medis, šaknis, keliai (paths), absoliutūs ir santykiniai keliai. Failų vardai. Simbolinės nuorodos (symbolic links), komandos `ln`, `cp -a`, `cp -r`. Komandos `find`, `xargs`. Rekomendacijos patogiam darbų ir projektų organizavimui. Naudotojų teisės. Priėjimo prie failų teisės (file access rights). Vykdymo teisės. Vartotojų grupės. Komanda `chmod`.

   Unix ir tinklas: `ssh`, `telnet`, `scp`, `rcp`, `rsync`, `wget/wput`, ...
7. Unix kaip programavimo aplinka: kompiliatoriai (`cc`, `gcc`, `g++`, `tcc`, `javac`, `g77`, `gfortran`, `ada`, `chill`, `fpc`, ...), programų kompiliavimas ir surinkimas (compiling and linking), surinkimo valdymas: `make`, `ant`, Perl ir Python `Makefiles`. Programų automatinis konfigūravimas su `automake` ir `autoconfigure`. GNU Make kintamieji ir

funkcijos. Make kalba. Aplinkos sukūrimas naudojant GNU Make. Integruotos terpės (Eclipse, Anjuta). IDE palyginimas su komandinės eilutės įrankiais. Elementarūs make-failai, taisyklės (rules), komandos, tikslai (targets), gerai žinomi tikslai (well-known targets). Savo programų kūrimas, diegimas, naudojimas.

8. Unix administravimo pradmenys: sistemos administratorius `root`. Programų diegimas/pašalinimas (`apt-get`, `emerge`, `rpm`, `yum`, BSD ports). Naudotojų sukūrimas, pašalinimas, administravimas. Geri slaptažodžiai. Komandos `useradd`, `userdel`, `usermod`, `chsh`, `chown`, `chgrp`. Direktorija `/etc`. Saugumo abėcėlė.
9. Programų rašymo apžvalga (priminimui): mašinos kodai, autokodai, asembleriai, makroprocesoriai, aukšto lygio programavimo kalbos. Bendras programos kūrimo darbo ciklas: edit-compile-link-(test)-run. Šiuolaikinių interpretuojamų kalbų darbo ciklas: edit-compile+test-run.
10. Versijų kontrolė – kas tai yra, kam ji reikalinga. Versijų kontrolės sistemų ir programinės įrangos (PĮ) kūrimo procesų variantai: paskirstytas ir centralizuotas PĮ kūrimas ir versionavimas. Centralizuotos versijų kontrolės sistemos: RCS, Subversion, CVS. Paskirstytos (distributed) versijų kontrolės sistemos: BazaarNG (`bzr`), `monotone`, `mercurial`, `opencm`, `git`, `arch`. Komandos darbas su versijų kontrolės sistema Subversion: revizijos ir versijos, atsikėlimas (checkout), pakeitimų įkėlimas (checkin), suliejimas (merging), užraktai (locking), konfliktai (conflicts) ir jų sprendimas. Žurnaliniai įrašai (log records). Senų versijų atsikėlimas ir analizė, skirtumai (`diff`, `annotate`), senų ir mišrių revizijų surinkimas, paleidimas ir testavimas.
11. Programų kūrimo organizavimo būdai, naudojant versijų kontrolės sistemas. Kamienas (trunk), šakos (branches). Darbas kamiene, darbas individualiose šakose. Šakų suliejimas su kamienu (branch merging). Suliejimo ir surinkimo organizavimo būdai: surinkimo eilė (merge queue), surinkimo serveris (merge server), programos surinkėjas (merge lieutenant/officer), pakeitimų išrankiojimas (cherry-picking). Suliejimų sekimas (merge tracking).
12. Programos laidų ciklo (release cycle) organizavimas, naudojant versijų kontrolės sistemas: versijos (versions), žymės (tags).
13. Programų testavimas – kam jis reikalingas, ką jis duoda ir ko ne. Apskritai, kodėl testavimas veikia. Automatinis testų valdymas Make programa. Regresijos testai. Modulių testai, arba vienetiniai testai (unit tests), testavimo serveriai. Ekstremalus programavimas (XP, extreme programming); kūrimas, paremtas testais (test driven development). Testų padengtis (test coverage). Testavimo privalumai, trūkumai, galimybės ir galimybių ribos.
14. Programų formalus specifikavimas ir teisingumo įrodymas. Pradinės sąlygos (preconditions), pabaigos sąlygos (postconditions), invariantai (invariants). Formalūs (automatizuoti) įrodymai. Projektavimas, paremtas kontraktais, Eifelio programavimo kalboje (design by contract, Eiffel programming language). Formalių metodų privalumai, trūkumai, galimybės ir ribos. Susipažinimas su Z-notacija, Vienos mokykla.
15. Programų optimizavimas ir profiliavimas. 20/80 našumo taisyklė. Profiliavimo įrankiai: `gprof`/`gmon`.
16. Moksliniai (bioinformatiniai) skaičiavimai ir galimybė juos valdyti make-tipo sistemomis. Pradiniai failai, algoritmai, parametrai, rezultatų failai. Apibendrintas skaičiavimo procesas ir jo valdymas make pagalba. Integravimas su vizualiomis (visual, windows based) sistemomis.