UNIVERSITÀ DI PISA

SISTEMI OPERATIVI E LABORATORIO

MAURIZIO ANGELO BONUCCELLI

Anno accademico	2018/19
CdS	INFORMATICA
Codice	277AA
CFU	12

Moduli	Settore/i	Tipo	Ore	Docente/i
LABORATORIO	INF/01	LABORATORI	48	LUCA ONETO GIUSEPPE PRENCIPE
SISTEMI OPERATIVI	INF/01	LEZIONI	48	MAURIZIO ANGELO BONUCCELLI

Obiettivi di apprendimento

Conoscenze

Lo studente acquisirà conoscenze relative ai concetti principali dei sistemi operativi e della loro implementazione, e alla programmazione concorrente.

Il corso intende fornire le conoscenze di base relative alla programmazione C con chiamate di sistema Unix/POSIX. In particolare, vengono prese in considerazione le chiamate standard per il trattamento di file e directory, processi, thread, segnali, pipe, socket e vari meccanismi di sincronizzazione.

Modalità di verifica delle conoscenze

La valutazione avverrà tramite una prova scritta e la discussione di un progetto.
Nella prova scritta verrà valutata la capacità dello studente di comprendere e di risolvere alcuni esercizi che presentano situazioni ipotetiche in un sistema operativo e verrà valutata la capacità di presetare contenuti del corso usando una terminologia appropriata.
Con il progetto verrà valutata la capacità dello studente di mettere in pratica le conoscenze acquisite, tramite la realizzazione di un progetto funzionante.

Prova scritta, e progetto in C e sua discussione.
Metodi:

- Prova scritta finale
- Prova di laboratorio (progetto)

Capacità

Lo studente acquisirà la capacità di scrivere programmi concorrenti che utilizzano chiamate di sistema per l'uso dei dispositivi e per l'interazione con il sistema operativo

Programmazione POSIX. Conoscenza degli aspetti fondamentali del Sistema Operativo UNIX.

Modalità di verifica delle capacità

Progetto in C e sua discussione.
Non sono previste prove per la verifica dei comportamenti.

Comportamenti

Il corso non si pone obiettivi di apprendimento in termini di comportamenti attesi.

Prerequisiti (conoscenze iniziali)

Una buona conoscenza di C. sono utili basi di architetture degli elaboratori e di programmazione

Indicazioni metodologiche

Le lezioni avvengono in aula o in laboratorio alla presenza del docente. Le attività di apprendimenti comprendono:

- seguire le lezioni
- attività pratica di laboratorio
- studio individuale

Non c'è obbligo di presenza alle lezioni

Lezioni frontali.
Attivita' di apprendimento:

- frequenza lezioni
- studio individuale
- lavoro in laboratorio

Frequenza non obbligatoria.
Metodi di insegnamento:

- Lezioni fronatli
- Lavoro di progetto

Programma (contenuti dell'insegnamento)

Il corso è strutturato in due moduli: il primo modulo presenta i fondamenti dei sistemi operativi e il secondo modulo di laboratorio mostra un'applicazione concreta dei concetti teorici presentati nel primo modulo.
Il primo modulo introduce i principi ed i concetti su cui si basano i sistemi operativi e analizza la loro realizzazione nei sistemi reali. In particolare, vengono presentate le tecniche che consentono di coordinare e gestire le risorse di un sistema di elaborazione e che permettono di trasformare la macchina fisica in una macchina astratta, dotata di funzionalità più convenienti per l'utente. Il corso lascia ampio spazio alla descrizione e all'esemplificazione di come i vari concetti presentati sono realizzati nei sistemi Unix/Linux e in quelli della famiglia Microsoft Windows. Il secondo modulo fornisce le conoscenze di base relative alla programmazione C con chiamate di sistema Unix/POSIX. In particolare, vengono prese in considerazione le chiamate standard per il trattamento di file e directory, processi, thread, segnali, pipe, socket e vari meccanismi di sincronizzazione.

- Strumenti per la programmazione C in ambiente GNU (e dintorni) (2h lez + 2h ese)
  - compilazione e linking
  - suddivisione di un programma in moduli: gestione(make) e archiviazione (ar)
  - debugging con gdb
  - cenni ad altri tool (valgrind)
- Unix e la shell (6h lez + 6h ese)
  - Unix: concetti di base, comandi/utility
  - La shell di Unix
    - introduzione alla shell
    - cenni agli script
    - personalizzazione con alias ed opzioni e variabili
    - espansione (alias, storia, tilde, percorso, variabili, sostituzione di comando, espressioni aritmetiche)
    - quoting
    - Controllo dell'I/O: ridirezione
    - Exit status
    - Combinare comandi
      - pipe
      - sequenza non condizionale
      - sequenze condizionali &&, ||
      - comandi composti (uso di {…} e (…))
    - Operatori su stringhe (pattern matching)
    - Costrutti di controllo
      - condizionale (if) e test
      - iterazione limitata (for)
      - scelta multipla (case)

- iterazione illimitata (while e do/while)
- menu' con select

- C e la programmazione di sistema (1h lez)
  - cos'e' una System Call
  - gestione sistematica degli errori
  - consigli pratici di programmazione
  - convenzioni e commenti
- I file (2h lez + 2h ese)
  - concetti base dei FS Unix
  - operazioni di base (open, read, write, close, lseek)
  - differenza tra chiamate di libreria e system call (write vs fwrite)
  - link: concetti e system call per la gestione
  - manipolare e accedere alle informazioni sui file (stat, fcntl, ioctl)
  - gestione delle directory (opendir,readdir,…,closedir)
  - duplicazione di descrittori (dup e dup2)
- I processi (2h lez + 2h ese)
  - il modello fork/exec
  - gestione dei figli (wait), exit status
  - atexit(), _atexit()
- I thread (3h lez + 4h ese)
  - i modello a thread
  - creazione e distruzione
  - mutex e variabili di condizione
  - soluzione ti tipici problemi di concorrenza
- Maccanismi di IPC : Pipe (2h lez + 2h ese)
  - Pipe anonime
  - Pipe con nome (FIFO)
- Maccanismi di IPC : Socket (2h lez + 2h ese)
  - Modello client-server
  - Socket per comunicazione locale (AF_UNIX)
  - Socket per comunicazione in rete (AF_INET)
- Segnali (2h lez + 2h ese)
  - Concetti generali: eventi asincroni, gestori di eventi
  - Ignorare, gestire, mascherare ed attendere i segnali
  - Gestione dei segnali con i processi e con i thread

Bibliografia e materiale didattico

T. Anderson, M. Dahlin : "Operating Systems: principles and practice", Recursive Books Ltd, 2013 Marc J. Rochkind. Advanced UNIX Programming 2nd Edition, Addison-Wesley Professional Computing Series, 2004.

Indicazioni per non frequentanti

gli studenti non frequentanti hanno accesso a tutto il materiale didattico (inclusa la registrazione delle lezioni) al sito web del corso

Modalità d'esame

Completamento progetto in C e sua discussione.

La valutazione avverrà tramite la discussione di un progetto assegnato durante il corso e realizzato dagli studenti, e tramite una prova scritta che verterà sugli argomenti di entrambi i moduli di laboratorio e di teoria. In ogni appello d'esame, l'ammissione alla prova scritta è vincolata alla presentazione (almeno 5 giorni prima della data dell'appello) di un progetto che deve essere valutato ammissibile per poter proseguire l'esame. Nel caso di un numero limitato di iscritti all'appello i docenti si riservano di sostituire la prova scritta con una prova orale equivalente, avvisando preventivamente gli iscritti tramite l'e-mail indicata dagli stessi all'iscrizione.
Nella prova scritta verrà valutata la capacità dello studente di comprendere e di risolvere alcuni esercizi che presentano situazioni ipotetiche in un sistema operativo e verrà valutata la capacità di presentare contenuti del corso usando una terminologia appropriata.
Con il progetto verrà valutata la capacità dello studente di mettere in pratica le conoscenze acquisite, tramite la realizzazione di un progetto funzionante.
Durante il corso si svolgeranno inoltre due prove (scritte) di verifica intermedia (orientativamente a metà del corso e alla fine del corso). Lo studente che ottiene un voto sufficiente alle prove di verifica intermedia è esentato dalla prova scritta, e dovrà solo discutere il progetto realizzato in uno degli appelli dello stesso anno accademico.
La validità del voto conseguito con le prove di verifica intermedia cessa al termine dell'anno accademico (quindi dopo l'ultimo appello straordinario, che si svolge normalmente ad Aprile dell'anno successivo). Ogni prova di verifica

UNIVERSITÀ DI PISA

intermedia consiste nello svolgimento di alcuni esercizi riguardanti entrambi i moduli (di teoria e di laboratorio) e può essere sostenuta prima della presentazione del progetto.
Si fa presente che per l'ammissione a tutte le prove d'esame, incluse le prove di verifica intermedia, è obbligatoria l'iscrizione secondo le modalità stabilite dal corso di laurea.

Pagina web del corso
http://didawiki.di.unipi.it/doku.php/informatica/sol/laboratorio/programma

Ultimo aggiornamento 01/02/2019 12:35