INF1060: avsluttende eksamen

Kandidatnummer: _____

1 Oppgave 1 - operativsystemer — *max 30 poeng*

Løsningsforslagene er mer eller mindre hentet fra foilene. Merk at andre forklaringer/bruk av andre ord også kan være fornuftige og bør også gi full uttelling!!

- 1a **(5 poeng)** – Et OS kan defineres på mange måter, så vær litt fleksible, men noen ting er viktige å nevne: mellomlag mellom HW og brukeren, tilbyr tjenester for å håndtere ressursene, sett nedenfra er OSet en ressurshåndterer (som bør være rettferdig, sikker og effektiv), sett ovenfra gir det et grensesnitt for brukeren til maskinen (virtuell maskin hvor lavnivå detaljer skjules).

- 1b **(4 poeng)** – I denne forbindelsen er en stack et kontinuerlig minneområde som typisk brukes til å lagre variable, parametere og register verdier (hvor hver funksjon ofte får en egen stack-frame). Den er FIFO med sine push og pop operasjoner. For en prosess er den som regel plassert sist i addresserommet og den Vokser mot lavere addresser.

- 1c **(4 poeng)** – Et systemkall er vanlig funksjonskall som kaller tjenester i OS kjernen, og som dermed danner kjernens grensesnittet mot brukeren. Systemkallene identifiseres ved et nummer som brukes som en indeks til en systemkall-funksjonstabell og parameterene puttes i et buffer.
  Kort sagt, det som skjer er (oppsummert fra foilene): 1) push parameters on stack; 2) call library code; 3) put system call number in register; 4) call kernel (TRAP, examine system call number, find handler, execute requested operation; 5) return to library and clean up (increase instruction pointer, remove parameters from stack); 6) resume process

- 1d **(5 poeng)** – En sidefeil en hendelse (interrupt sendt av hardware) som oppstår når et program prøver å aksessere en side i minnet som er mappet inn i adresserommet, men som ikke er lokalisert i det fysiske minnen (men da på disken).
  Kort sagt, det som skjer er (oppsummert fra foilene): 1) Hardware traps to the kernel saving program counter and process state information; 2) Save general registers and other volatile information; 3) OS discovers the page fault and determines which virtual page is requested; 4) OS checks if the virtual page is valid and if protection is consistent with access; 5) Select a page to be replaced; 6) Check if selected page frame is “dirty”, i.e., updated. If so, write back to disk, otherwise, just overwrite; 7) OS finds the disk address where the needed data is located and schedules a disk operation; 8) A disk interrupt is executed when disk I/O operation is finished, page tables are updated, and the page frame is marked normal state; 9) Faulting instruction is backed up and the program counter is reset; 10) Faulting process is scheduled, and OS returns to the routine that made the trap to the kernel; 11) The registers and other volatile information are restored, and control is returned to user space to continue execution as no page fault had occured. — man trenger ikke forklare dette så detaljert, men ha fått med seg hovedpoengene.

- 1e **(4 poeng)** – En konteks-switch er operasjonen med å bytte fra en kjørende prosess (A) til en annen (B), og det som skjer er at 1) prosess A stoppes; 2) tilstanden (registere, etc) til A lagres (på stakken, evt. PCB); 3) tilstanden til prosess B gjenno-prettes (legge verdiene tilbake i registerene); 4) prosess B gjenstartes fra punktet den stoppet sist ved at programtelleren er lest inn i EIP registeret.

- 1f **(4 poeng)** – Extents er ment å adressere problemet med mange pekere i inoder (metadatastrukturen for å hlde orden på blokkene til EN fil), og man har da lagt til en teller til hver peker som sier hvor mange _kontinuerlige blokker_ som starter med blokken pekeren peker på. Den store fordelen er at man ikke trenger så mange pekere og mye plass i inoden, og ofte trenger man ikke mer en det som direkte kan brukes i inoden selv. Ekstentene er som sagt også kontinuerlige som gir rask lesing uten inter-blokk seeks mellom blokkene innenfor extenten. Mao, filfragmentering er redusert.

- ❑ 1g **(4 poeng)** – Problemet er at forskjellige maskiner har forskjellige måter å representere multi-byte verdier. Det vil si at forskjellige maskiner tolker rekkefølgen på bytene forskjellig. Dette går fint så lenge man jobber innen samme maskin, men når man må sende data mellom maskiner, gir dette et problem ettersom for eksempel en 32-bit int tolkes forskjellig. Løsningen er ofte å ha oversettelsesfunksjoner (htons etc...) som oversetter til en felles bestemt rekkefølge network order (= big endian).

2 Oppgave 2 - flervalgsoppgaver: OS — *max 20 poeng*

Hvert riktige svar gir 2 poeng, og de riktige er:
2a-3, 2b-2, 2c-1, 2d-3, 2e-3, 2f-2, 2g-2, 2h-3, 2i-1, 2j-5

- ➽ **Poeng totalt: _____** (*max 100 poeng*)