OFFLINE NR.1 • 2013

ÅRESPILLET

Hvem blir ÅREts mester?

DUNGEONS AND DOUCHEBAGS

Vi har tatt en titt på rollespill

Offline er et tidsskrift for linjeforeningen Online.

REDAKTØR:
Rikard B. Eide
rikardbe@online.ntnu.no

MARKED:
Even Lislebø
e.lislebo@gmail.com

ØKONOMI:
Rikard B. Eide

FORSIDEBILDE:
Redaksjonen

GRAFISK PROFIL:
Beate Hay Sandmo

LAYOUT:
Rikard Eide
Kathrine Steffensen
Sverre Johann Bjørke
Øyvind Hellenes
Hallvard Jore Christensen
Lorents Gravås
Kari F. Skjold
Ingrid W. Myrann
Aleksander Skraastad

TRYKK:
Øien & Indergaard
Opplag: 512

KONTAKT:
Redaksjonen, proKom
redaksjonen@online.ntnu.no
Sem Sælands vei 7-9
7034 TRONDHEIM
http://online.ntnu.no/offline

REDAKTØRENS

Return.
Et enkelt linjeskift er alt som skal til. En ny mulighet, en ny start. En ny sjanse til å rette opp de feil du begikk i forrige avsnitt. Alt ettersom, kan et linjeskift ha både store og små konsekvenser. Kanskje befinner du deg på siste linje, og kan starte med et blankt ark. Eller du befinner deg på åttende linje i syvende setning, og et linjeskift vil ikke bety spesielt mye.

Når det er sagt ønsker jeg deg velkommen til Offlines første utgave i 2013. Denne velkomsten retter jeg både mot våre (tro)faste lesere, og alle dere som plukker opp Offline for første gang. Jeg har grunn til å tro at sistnevnte er en delmengde i full vekst. For hvis du om en fire linjers tid ser til venstre, vil et forhåpentligvis kjent tall fange oppmerksomheten din. Nei, ikke stopp opp for å hverken telle eller se. Vis tålmodighet, og tiden vil sørge for at dine drømmer går i **oppfyllelse**.

Mot formodning, var ikke dette et sitat av en kjent, gammel mann, men et hint om at de faktisk gjorde det. Så følg med i timen, noe annet har vi ikke tid til.

Tidligere har `README` vært en ensom skuespiller. De har stått alene på denne (noe smale) scenen for linjeforeningsaviser med stort nok opplag til å bli plukket opp av utenforstående på Stripa. Men fortvil ikke, vår kjære Aksel Hennie; Nicolai Cleve Broch har inntatt scenen. Sammen ser jeg for meg at vi skal løse mysterier og komme til bunns i store skandaler. Om dette blir realiteten, eller om vi utvikler oss i samme retning som landets største aviser, vil kun fremtiden vise.

Snakker om fremtiden, så har det seg slik at jeg er i en posisjon som gjør meg utrolig god til å forutse nettopp denne. Allerede nå vet jeg hva du kommer til å lese på neste setning, eller neste ord, for den saks skylles før du klipper. - Ikke prøv å leke smart med meg, det er nemlig jeg som styrer denne opplevelsen, enten du liker det eller ei.

Tilbake til saken, jeg nevnte noe med å forutse fremtiden, og hintet til at det landets største aviser skriver om, stort sett er søppel. Med dette friskt i minnet ønsker jeg å avslutte med et løfte, et løfte om verdighet. Selv om vi, mest sannsynelig, kommer til å bli like store som VG, Dagbladet eller Her&Nå skal vi aldri havne i posisjonen hvor vi blir forvekslet med dem. Dette gjelder ikke Se og Hør selvfølgelig. Tvert imot er det bare hyggelig om vi blir forvekslet med dem. God lesning.

Rikard Eide
Redaktør, Offline
rikardbe@stud.ntnu.no

//STYRE-ORD

Denne utgavens styreord har jeg lyst til å dedikere til mine flotte og ressurssterke kvinnelige medstudenter på informatikk. I skrivende stund er det omtrent akkurat to år siden jeg skrev min første artikkel sammen med ei venninne, om det å være jente på informatikk, i den aller første utgaven av Offline. Denne gangen skal jeg ikke skrive om mine erfaringer, men om hva jeg ønsker for både meg selv og for alle dere.

Først og fremst ønsker jeg at vi skal ha tro på oss selv. Glem janteloven, glem å tvile på oss selv og tro på at vi kan klare hva som helst, for det kan vi. Jeg ser alt for ofte jenter som ikke føler de mestrer, som ikke tror på sine egne evner og som tilslutt ikke tør å gjennomføre. Tro meg, jeg vet hvordan det er. Allikevel er det alltid noe man mestrer, noe man gjennomfører og noe man kan være stolt av, og da vil jeg at vi skal være det! Aldri la noen fortelle oss at vi ikke kan gjøre noe, og om det skjer, motbeviser vi det.

Jeg ønsker at vi skal bli dyktige. Jeg ønsker at vi skal utforske informatikkverden, finne det vi brenner for og øve til vi blir best. Om det er programmering, prosjektledelse, drifting av servere eller en av de andre utallige mulighetene vi har med en informatikkbakgrunn, ønsker jeg at vi blir dyktige på vårt felt. Jeg ønsker at vi tar opp konkurransen med gutta, og ikke bare måler oss mot hverandre. Jeg håper vi aldri finner oss i å ha en "kvotefyll"-stilling, og at vi setter stoltheten vår inn i å bli ansatt på grunn av det vi er dyktige til. Jeg ønsker vi blir flere. Jeg håper at når vi møtes igjen på Onlines femtiårsjubileum er det like lang kø på jentedoen som på guttedoen. Jeg ønsker at jenteprosjektet Ada klarer å oppfylle ønske sitt om å bli overflødige, og at valget om å studere informatikk har blitt like vanlig for jenter som for gutter.

Jeg ønsker at vi skal være modige. Å være én av så få er vanskelig. Det er lett å havne i søkelyset, men det er vanskelig å havne i søkelyset for de riktige tingene. Jeg ønsker at vi sier meningene våre, selv om det er skummelt. Jeg ønsker at vi tør å satse, selv om vi har noe å tape. Mest av alt ønsker jeg at vi tør å være oss selv, for det kan være det vanskeligste.

Sist, men så visst ikke minst, ønsker jeg at vi støtter hverandre. Det er så få av oss jentene på informatikk, at støtten vi kan tilby hverandre er uvurderlig. Vi har ikke råd til baksnakking og misunnelse. Aksepter at vi alle er forskjellige, oppmuntre hverandre til å gjøre det bra og bygg hverandre opp. Det vil ingen tape på.

Til slutt vil jeg minne alle på hvor heldige vi er. Vi får holde på med et fagfelt som er i så kontinuerlig utvikling at det aldri blir kjedelig, vi har utallige jobbmuligheter, både underveis i studiet og når vi er ferdige, og vi får studere med noen av kuleste menneskene i Trondheim. Studiet kan føles tøft og det kan være vanskelig å se det store bildet, men det er der vi i Online ønsker å stille opp. Hvis noen har spørsmål eller hvis jeg kan hjelpe dere med noe, ikke nøl med å ta kontakt. Husk at dere kan nå så langt dere vil, og at det blir lettere hvis vi står sammen.

Hanne Gunby
Leder, Linjeforeningen Online
leder@online.ntnu.no

EN TRUSSEL MOT ANARKIET?

Regjeringen la nylig frem forslaget til ny åndsverkslov. Kritikken har ikke latt vente på seg. Lovforslaget åpner nemlig for at internettilbydere kan bli rettslig pålagt å blokkere hele domener.

TEKST: ALEKSANDER SKRAASTAD

Endelig ferdig

I flere år nå har et høringsutkast til den nye loven vært på rundgang hos en rekke departementer, institusjoner og interesseorganisasjoner. Spesielt interesseorganisasjonene har hatt de sterkeste meningene i debatten. På den ene siden kan enkelte av forslagene ses på som radikale forsøk på å etterkomme det kommersielle og politiske presset fra de store selskapene i musikk- og filmindustrien. På den andre siden forsøker motstandere å argumentere for at de samme forslagene er direkte brudd på ytringsfriheten. Det er nettopp denne diskusjonen som har blitt veldig interessant. Nettopp hvordan man tolker loven vil være avgjørende for om en nettside kan bli blokkert med rettslig fullmakt eller ei. Andre argumenter fra motstanderne er blant annet knyttet til personvernhensyn. Det nye lovforslaget inneholder nemlig tre svært interessante momenter.

(Nesten) fritt frem!

Den første av de tre er at organisasjoner, ja faktisk også enkeltpersoner om de måtte ønske det, vil få tilgang til å loggføre IP-adresser i jakten på nettpirater. Konsesjonen man tidligere var pålagt å ha for å drive slik virksomhet erstattes med en enkel meldeplikt til datatilsynet, samtidig som de generelle kravene til slik kartlegging fra personopplysningsloven forblir uendret. Det andre forslaget er mer en tydeliggjøring av rettighetshaveres mulighet til å kontakte domstolene, for å få tillatelse til å hente ut personopplysninger tilknyttet loggførte IP-adresser. Denne prosessen er i dagens lovverk nokså vag, og dekkes av tvistemålsloven. Den tredje og også den mest radikale endringen er å blokkere domener til nettsteder som åpenbart bryter opphavsretten.

Halvveis effektive tiltak

Det er nettopp dette punktet som har satt sinnene i kok hos motstandere av lovforslaget. Det forsikres allikevel fra politisk hold om at denne klausulen i lovverket vil ha en svært høy terskel, og begge parters syn skal veies mot hverandre, og ytringsfrihet og personvern vil veie tungt. Det har vært diskutert ulike måter å praktisk gjennomføre en slik blokkering på. Den aller enkleste, dog minst effektive av tiltakene vil være å pålegge nettilbyderne å fjerne tilgangen til et domene for sine brukere. Det har også vært diskutert aktiv pakkeinspeksjon og andre tekniske løsninger, men omfanget av en slik løsning vil trolig langt overskride den faktiske effekten. Hvor mye internettleverandørene indirekte skal sniffe i sine kunders trafikk er i hvert fall en diskusjon i en personvernmessig gråsone. Det er naturligvis umulig å fullstendig blokkere tilgangen til et domene, da det alltids vil være mulig å benytte proxy- og VPN-løsninger for å sikre seg tilgang. Dette er dog noe interesseorganisasjonene for musikk- og filmindustrien ikke bryr seg nevneverdig om. Hensikten er å begrense omfanget.

Kan få uante konsekvenser

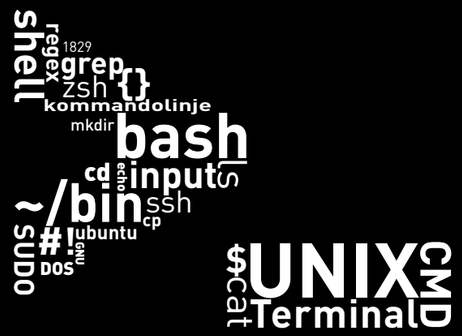
Her i Norge har håndheving av opphavsrett på internett vært en svært liten sak. I motsetning til de forente stater over atlanterhavsdammen blir det hele en dråpe i nettopp dét havet. SOPA/PIPA og tilsvarende forslag dukker jevnlig opp. Det er helt naturlig at eierne har et ønske om å beskytte sine interesser, og det bør de få lov til. Men la oss nå sette oss litt tilbake og kaste lys over det store bildet. Det er rimelig å anta at selv med de endringene som foreslås, så vil hensyn til ytringsfrihet og personvern veie tilstrekkelig høyt. Selve diskusjonen kommer i hvilken grad man skal veie hensyn til ytringsfrihet opp mot hvilken grad av brudd som utøves mot opphavsretten. Hvilke måleenheter er det snakk om? Hvor mye er en ærlig stemme verdt i forhold til antallet tilgjengelige Hollywood-filmer? Det er disse spørsmålene som er vanskelig å besvare. Det er nettopp disse spørsmålene som må besvares, og deretter dømmes etter. Torgeir Waterhouse, direktør for internett og nye medier i IKT-Norge er kritisk. Til NRK uttalte han på starten av året at den nye loven er en "form for sensur i offentlig regi" og at loven kan få "konsekvenser vi enda ikke vet omfanget av". -Problemet med ulovlig fildeling kan kun løses ved å gi folk tilgang til gode lovlige tjenester. Folk forventer å få tilgang til alle typer informasjon og medieprodukter digitalt. Når det ikke er tilgjengelig vil alltid noen dele det ulovlig, uttalte Waterhouse til NRK. Han mener det vil være svært vanskelig å praktisk gjennomføre blokkering av nettsteder og risikoen for at forslaget vil kunne ramme nettsteder utover det forslaget er tiltenkt.

Bedre tjenester er kanskje løsningen?

Når hvermannsen med Datatilsynet i ryggen kan hamstre IP-adresser fra nettpirater og selge informasjonen til interesseorganisasjoner åpnes et nytt marked for handel med brukerdata. Det som da er en smule uheldig er at når man skal måle effekten av dette forslaget kommer det nettopp i kjølvannet av en rekke nye tjenester for musikk- og filmstreaming som virkelig har fått rotfeste i den teknologiske befolkningen. Det registreres allerede en enorm økning trafikk på de nye tjenestene som NETFLIX, HBO Nordic, ViaPlay et cetera. Den enkle tilgangen på on-demand tjenester i stuen vil åpenbart være med på å påvirke hvor mye hverdagspiratene laster ned. Det er nettopp dette som trolig vil ha den største effekten på ulovlig opp- og nedlastning av underholdning. I mye større grad enn en enkel domeneblokkering av ett eller flere nettsteder. Det virker som filmbransjen nå gradvis begynner å følge i musikkbransjens fotspor, og ser etter nye forretningsmodeller tilpasset den digitale tidsalder. Dette er dog bransjer som er kjent for å være trege, og langt i fra være i stand til å snu seg på en femøring. Stort sett samtlige teknologier har vært saksøkt utallige ganger i frykt for å miste inntekter. LP-Platen, kassettspilleren, VHS, og så videre. En ting er i hvert fall sikkert. Gigantsøksmål i milliardklassen har uansett svært lite for seg

Går selv til sak for brudd på opphavsrett

Med forsikringer fra politikerne om at ytringsfrihet og personvern skal veie tungt virker det som forslaget til den nye loven ikke vil være en nevneverdig trussel for et åpent internett. Men det er allikevel ikke uten risiko for at mulighetene det åpnes for kan medføre konsekvenser som på det nåværende tidspunkt vil være vanskelig å forutse. Historien har nok av eksempler på uforutsette hendelser kun de færreste var i stand til å spå. Nylig ble dokumentaren om grunnleggerne av The Pirate Bay, TBK AFK sluppet, fritt tilgjengelig for nedlastning fra deres eget hjertebarn. Selv om argumentene tidvis er gode er det ikke til å legge skjul på at nettsidens navnevalg er nokså avslørende for hva hensikten med nettsiden er. Uavhengig av om de ikke legger seg borti hva brukerne laster opp eller ei. Da er det et lite paradoks at The Pirate Bay vil saksøke en antipiratgruppe for å stjålet logoen deres. Sirkelen av kaos som kalles opphavsrett på internett er komplett.



Bash sa du?

Kommandolinjen - fryktet og hatet av massene, forgudet av de opplyste. I denne artikkelen oppsummerer vi hvorfor kommandolinjen ikke er så skummel som mange tror, og hvordan du kan bruke den til å spare tid og automatisere kjedelige oppgaver.

TEKST: DAG OLAV PRESTEGARDEN

Er du som folk flest har du trolig fått med deg at grafiske brukergrensesnitt, datamus og dårlige kontorrekvisita-metaforer er "det nye kule" i dataverdenen. Som informatikkstudent kan det nok likevel tenkes at du er hakket mer teknisk begavet enn mannen i gata, og av og til har behov for å logge inn på en server over SSH eller føler kallet til å leke litt med Linux. Min påstand er at alle på et såpass avansert nivå som informatikk tross alt holder, har godt av å ha et forhold til kommandolinja. Ikke bare for å kunne gjøre de sjeldne oppgavene som må gjøres en gang i blant (./configure, make, make install) (sudo apt-get upgrade), men også fordi den ofte eksponerer avansert funksjonalitet man sjelden finner i grafiske brukergrensesnitt.

Ikke bare kommandoer

Det første møtet med kommandolinjen er for mange en rekke kryptiske kommandoer man følger i en tutorial for å få installert et eller annet program man trenger, og "magiske" kommandoer som ./configure, make, make install og sudo apt-get install er noe mange har møtt på.

For de som har fått ennå litt mer tid på baken med kommandolinjen begynner gjerne en del kommandoer for navigasjon i filsystemet å bli en del av rutinene, og kommandoer som ls, cd, rm, cp og mv sitter godt i minnet. Men her stopper det for mange, noe som egentlig er veldig synd, for det er ikke mye man trenger å kunne for å forstå hvordan kommandolinjen fungerer, og hvordan man dra nytte av den for å effektivisere sin hverdag som informatiker.

Input/Output Redirection

Programmer som lever i et typisk kommandolinjemiljø kommer med tre forhåndsdefinerte veier ut og inn av programmet: stdin, stdout og stderr. Når vi kjører et program uten å eksplisitt definere disse er stdin automatisk koblet til det som kommer inn i terminalen, typisk fra tastaturet ditt, mens stdout og stderr - vanlig output og feilmeldinger - automatisk sendes tilbake til terminalen som viser dette på skjermen.

Siden alt i Unix er filer kan vi koble disse tre kanalene, eller fildeskriptorene, til andre ting som er filer, og dermed få lagret output av programmet i filer eller sendt det til andre systemer som kan representeres ved fildeskriptorer (f.eks. kan vi sende output rett i søpla ved å redirecte til /dev/null). Et par eksempler på bruk kan vi da se i følgende eksempel.

```
offline@proKom:~$ ls
hello.txt
offline@proKom:~$ cat hello.txt
Hello Offline!
offline@proKom:~$ cat hello.txt > new.txt
offline@proKom:~$ ls
hello.txt    new.txt
offline@proKom:~$ cat new.txt
Hello Offline!
```

Hadde vi hatt et mer avansert program som kan produserte feilmeldinger også, kunne vi redirected stderr til en egen fil for å logge eventuelle problemer som måtte oppstå underveis.

Et eksempel kunne da blitt noe ala dette:

```
offline@proKom:~$ ./some_program > output.txt 2> errors.log
```

Vi ser her at 2> redirecter stderr, om vi absolutt ville kunne vi brukt 1> for å redirecte stdout, men dette er ikke nødvendig. Om vi ønsker å redirecte både stderr og stdout til samme resultat kan vi bruke snarveien &>.

Pipes

Pipes lar oss koble stdout fra et program rett inn i stdin i et annet program. Dette lar oss koble sammen programmer i en kjede, nesten som et samlebånd, eller en rekke funksjonskall i et funksjonelt programmeringsmønster. Denne sammenkoblingen av programmer i en kjede glir rett inn i den typiske Linux-filosofien om å ha mange små programmer som alle gjør sine enkle, avgrensede oppgaver og ingenting annet. I motsetning til et grafisk grensesnitt der programmer ofte må gjøre alt mellom himmel og jord før hele brukermassen er fornøyd, fungerer dette strålende i kommandolinjemiljøet. De kjente, kjære programmene for vanlige oppgaver, som søking, sortering og manipulering kan gjenbrukes, og nye programmer kan fokusere på å bli kjempegode på sin spesialnisje.

Et eksempel på en slik spesialisering, og ikke minst selve bruken av pipes, kan vi se i eksempelet under. Vi har her en fil med navn i vilkårlig rekkefølge. Programmet cat er spesialisert på å lese en eller flere filer og skrive dem ut til stdout mens sort tar input på stdin, sorterer linjene som kommer inn, og sender dem utigjen på stdout. Ved hjelp av en pipe kan vi koble sammen disse to spesialiserte programmene og få det resultatet vi ønsker. (For flisespikkerne: Sort har funksjonalitet for å lese filer, og hele kommandoen kan dermed omskrives til "sort navn.txt", men da hadde jo ikke poenget kommet så godt frem).

```
offline@proKom:~$ cat navn.txt
Flette-Mette
Blodgunn
Hvalhild
Finnbjørg
offline@proKom:~$ cat navn.txt | sort
Blodgunn
Finnbjørg
Flette-Mette
Hvalhild
```

GNU coreutils

Det de fleste av oss, inkludert forfatteren, til daglig bare kaller Linux, beskrives egentlig best som GNU/Linux, dette fordi vi bruker operativsystemet GNU på toppen av en Linux-kjerne. I disse systemene finner vi alltid samlingen av programmer kalt GNU coreutils. Dette er en pakke som inneholder de vanlige kommandoene for filmanipulasjon, mappevisning, søking, sortering og en rekke andre funksjoner.

Ved å ha en grunnleggende kunnskap om hva de forskjellige programmene i GNU coreutils gjør, og hvilke muligheter de byr på kommer man langt i kommandolinjekarrieren. Alle de magiske argumentene og kommandoene for å faktisk få dem til å gjøre de mest avanserte funksjonene slår man lett opp i det man trenger dem. Som studenter er de fleste av leserne godt kjent med Google, og skulle man ønske dokumentasjon på "gamlemåten", eller sitter i en situasjon der internett ikke er aktuelt, finner man en god oversikt over argumenter og funksjoner hos de forskjellige verktøyene ved å kalle dem med "--help" parameteret. Trenger man mer utfyllende informasjon kan også systemets manualsider brukes ved hjelp av kommandoen "man <kommandonavn>".

Use cases

Av erfaring er det ofte vanskelig å overbevise "hedningene" om hvor bra kommandolinjen er, og jeg regner ikke med at selvstendig tenkende lesere av dette magasinet er noe lettere å lure inn i det svarte skallet. I et nobelt forsøk på å overbevise deg som leser skal vi derfor ta for oss et mer eller mindre realistiske bruksområde.

Case: Telle kodelinjer i et Java-prosjekt

Du har en tåpelig faglærer som er sykelig opptatt av at alle øvinger skal bestå av minst 800 kodelinjer. Læreren har laget en enkel definisjon på hva "en kodelinje" betyr: Linjer som inneholder ';' eller '{'. På denne måten håper han å få med de fleste kontrollstrukturer, vanlige statements osv, men slipper å telle med kommentarer og tomme linjer.

Et av problemet med tellingen er at prosjektet ditt består av veldig mange filer, og det er kjedelig å lese gjennom alle filene i alle mappene for alle øvingene. Vi har derfor en gyllen anledning til å automatisere denne oppgaven.

De som er flinke å plukke ut essensen av oppgaver, vil da kanskje se at vi må gjøre noe à la dette:

1: Finne alle filer, i alle mapper (rekursivt) som inneholder java-kode.
2: Ignorere alle linjer som ikke inneholder '{' eller ';'
3: Summere anntallet linjer vi sitter igjen med.

For disse tre deloppgavene, skal vi bruke 3 verktøy fra coreutils:

1: find - en kommando som kan finne alle filer som matcher et søkemønster. I vårt tilfelle alle som har .java i filnavnet.
2: grep - gir oss mulighet til å kun plukke ut linjer som inneholder søkeord.
3: wc - ordtellingsprogram med argumenter for å telle antall bokstaver, ord, linjer osv. I vårt tilfelle, linjer. Vi starter med første ledd:

```
find . -name '*.java'
```

Denne kommandoen gir oss filsti til alle java-filer vi finner i et rekursivt søk. For å få ut innholdet på en måte som lett lar seg pipe videre legger vi på -exec cat {} \;

ILLUSTRASJON: Hentet fra xkcd.com/208/

for å kjøre cat og printe ut hver enkelt fil. Vi får da første ledd:

```
find . -name '*.java' -exec cat {} \;
```

Outputen fra denne kommandoen kan vi lett mate gjennom grep. For å finne filer som inneholder enten ";" eller "{" kaller vi grep med -E argumentet som betyr at vi kan bruke regex[1] i søkestrengen, og gir den søkestrengen ";|{". Andre ledd blir da grep -E "{|;".
Den totale kommandoen så langt:

```
find . -name '*.java' -exec cat {} \; | grep -E "{|;"
```

vil nå printe alle linjer som er i en java-fil og som inneholder ";" eller "{". Det eneste som gjenstår da er å telle antall linjer. Vi legger på en siste pipe, og sender samtlige linjer gjennom ordtellingsverktøyet wc. For å telle linjer istedenfor ord må vi også sørge for å bruke parameteret -l for å si at vi ønsker antall linjer.

Totalt vil nå få en kommandolinjekjøring som ser noenlunde slik ut:

```
offline@proKom:~$ find . -name '*.java' -exec cat {}
                        \; | grep -E "{|;" | wc -l
125
```

Og vi ser at vi mangler mye før vi kan levere inn en god øving til vår kjære faglærer.

På tampen

Vi har i dag sett at Bash er mer enn bare et svart vindu der man skriver navnet på programmer. Bruksmønsteret med kall til forskjellige småprogrammer, gjenbruk av resultater og oppdeling av oppgaver ligner på mange måter på programmering, og burde ikke være uoverkommelig for den jevne informatiker. Noe av moroa med slike kommandoer er selvsagt også at de kan skrives på et utall forskjellige måter, og bare på de enkle eksemplene som er vist over finnes det trolig mye bedre måter å gjennomføre dem på. Jeg har likevel prøvd å holde meg til eksempler som er lettleste og lettforståelige. Om du har fått smaken for kommandolinjen anbefaler jeg absolutt å ta en tur i dokumentasjonen til coreutils for å se etter alternative og bedre løsninger på problemene som er beskrevet her.

OPPSETT AV KOMMANDOLINJEN

For de som kjører en Linux-distribusjon bør alle disse verktøyene allerede være på plass på systemet. Brukere med MacOS har trolig installert både bash og en rekke verktøy som fungerer noenlunde likt de som ligger i GNU coreutils. Ønsker man "the real deal" eller opplever at noen av de innebygde programmene ikke er tilstrekkelige kan man installere coreutils gjennom macports e.l.

De fleste sitter nok likevel på software fra gode gamle Microsoft, og selv om det ikke følger med noe som ligner på et brukbart kommandolinjemiljø i den gården finnes det flere måter man kan få installert et fungerende bash-miljø. GitHub leverer en kjempebra Git-klient for Windows som inkluderer et kommandolinjemiljø, og som i tillegg er utrolig lett å sette opp. For de som ønsker kjapp tilgang til bash på Windows anbefales denne fremgangsmåten på det varmeste.

```
offline@proKom1829:~$ ls
hello.txt   begreper.txt   navn.txt  script.py

offline@proKom1829:~$ cat begreper.txt

ls:     Viser innholdet i mappe.
cat:    Printer innholdet i filen til standard
        output.
grep:   søker etter bestemt tekst i en fil.
regex:  (Regular expressions) Beskriver et sett
        med strenger -et mønster- som følger
        gitte syntaksregler.
```

Dungeons and Douchebags

For meg har Dungeons and Dragons alltid vært et veldig mystisk spill. Hele livet har jeg latt meg forhekse av fortellinger om drager og magi, men det har alltid vært vanskelig å forestille meg hvordan det egentlig spilles. Hvordan er det egentlig å styre en karakter uten et ordentlig spillbrett, der du og dine medspillere skaper historien?

TEKST OG FOTO: THOR HÅKON BREDESEN

Da rollespillet Dungeons and Dragons ble lansert i 1974, var det begynnelsen på noe stort: Det ble det første i en lang rekke vi i dag kjenner som moderne rollespill. Spillet baserer seg i stor grad på tidligere strategikrigsspill, men til forskjell fra disse får du her én karakter å spille med, i stedet for en hel hær. Siden den gang har det kommet flere versjoner av spillet med en rekke tilleggslitteratur.

Før du kan starte å spille, må du lage deg en egen karakter. Det finnes karakterer som er forhåndslagd, men mye av moroa er likevel å skape seg noe eget. For det utrente øye kan det virke som en omfattende prossess. Det skal velges hva som skal være med i ryggsekken, hva slags våpen du skal bruke, valg av klær og ikke minst; ferdigheter. Det finnes i utgangspunktet syv ulike raser å velge blant, deriblant alver, mennesker, gnomer og dverger. Disse kan kombineres fritt med elleve ulike klasser, som jeger, trollmann, tyv eller barbar. Dette kan gi opphavet til mange interessante personligheter. Med det er Kamuflasje-Kai født, en halv-alvisk jeger med pil og bue, etter et ønske om å være som Aragon, bare litt mer alvisk.

En annerledes spillopplevelse

I Dungeons and Dragons går du sammen med andre venner og medeventyrere for å danne et reisefølge, som sammen skal ut på eventyr og overkomme hindringer på veien. I følget mitt var den lite karismatiske dvergen Gladriel, den halvt alviske trollmannen Mithrandor og William Dearhorne, menneske og kriger.

Det finnes utallige kampanjer og historier å spille, og det er fullt mulig å lage sine egne. Den som forteller historien og styrer hvordan omgivelsene reagerer på reiseføgets avgjørelser kalles Dungeon Master(DM). Normalt er det samme Dungeon Master gjennom en hel kampanje, som kan vare i alt fra noen uker, til flere år.

For å avansere, gjøre karakteren din bedre og oppgradere utstyret, må du spille en god stund. Dette gjøres gjennom hele kampanjen. Eventyret vi spilte denne gangen varte kun i fire timer, og var bare en introduksjon til spillet. Det ga oss nok tid til å bli kjent med spillestilen og litt av mulighetene i spillet, men karakterene våre kom ikke over første nivå. Det høyeste du kan utvikle en karakter i Dungeons and Dragons er nivå 20.

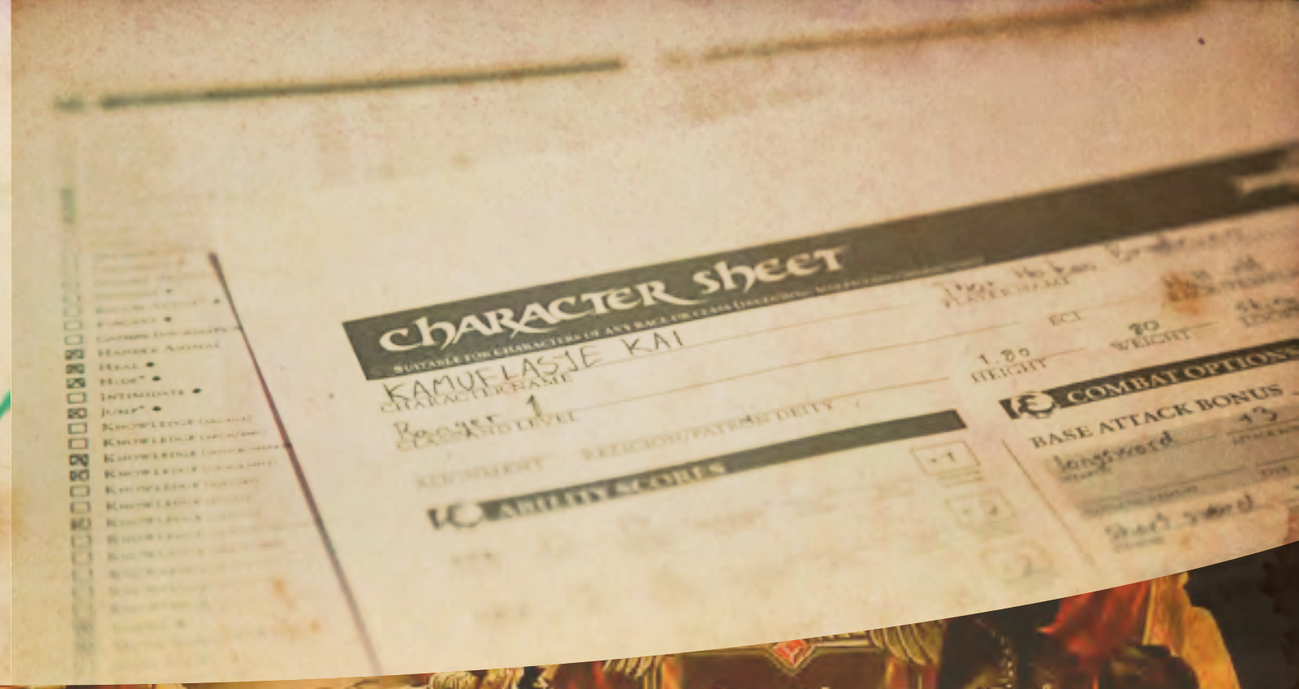
For en som aldri har spilt et rollespill av denne typen før, er det en veldig uvant spillestil. Dungeon Masteren starter med å introdusere omgivelsene, og etter en dypere introduksjon blir det opp til oss å fortelle hva vi vil gjøre.

It begins

Vår historie startet i den lille byen Montville. Uten helt å vite hva vi skulle ta oss til, fikk vi høre at innbyggerne kom ned fra steinbruddet der de jobbet, selv om det var midt i arbeidsdagen. Vår første tanke var selvsagt å undersøke hva som hadde skjedd. Det viste seg at en stor dør var kommet til syne, og den lot seg verken åpne eller hugge i gjennom. Siden man ikke kan se omgivelsene dine, blir man nødt til å nøye seg med beretninger og spørsmål til Dungeon Masteren. Det fungerer veldig bra når du blir vant til det.

Det er dette som er mye av kjernen i Dungeons and Dragons: Om du vil gå et sted må du fortelle at du skal gå dit, og hva du gjør der. For å finne ut mer om den ugjennomtrengelige døren valgte vi å snakke med borgemesteren, bartenderen på vertshuset vi bodde og en gammel, vis mann. Ut i fra informasjonen vi fikk tok vi valget om å reise til en større by, Beehaven, for å søke svar der.

Reisen blir for lang og detaljert til å kunne få med alt i en artikkel med respekt for seg selv, men i løpet av de første par dagene kan det nevnes at Mithrandor drepte en hund, kjøpte seg en lang pipe, og vi fant svar på hvordan vi skulle komme oss gjennom den tidligere ugjennomtrengelige døren.



Bak den fant vi en hule, og alle selverklærte nerder vet at huler alltid inneholder monstre i forskjellige størrelser og fasonger. Denne var selvsagt ikke noe unntak. Noen dramatiske kvarter senere hadde vi med nød og neppe unnsluppet den sikre død og fortsatte innover i mørket.

På dette tidspunktet hadde ingen av oss noe særlig med penger. Jeg tenkte jeg skulle være smart og lete rundt i rommet vi befant oss i for å se hva jeg fant av verdi, men dette skulle vise seg å bli et skjebnesvangert valg. Etter å ha kastet en terning for å se hvor godt jeg lette, fant jeg en krukke med 500 sølvmynter, noe som var mye penger i våre øyne.

Kampmodus

Vi gikk videre inn i et nytt rom. Etterhvert som vi beveget oss lengre innover skildret Dungeon Masteren hvordan det var blod og lik på bakken, og at det foran oss var noen gryntende lyder. Dette viste seg å være zombier. Vi forberedte oss på et nytt oppgjør, og krysset både fingre og tær for at det ikke skulle være det siste.

Når man skal engasjere en fiende bruker man en litt annen spillestil: For å lettere kunne visualisere situasjonen blir kampområdet tegnet opp på en egen duk med ruter som som viser hvor vi er, hvor vi kan bevege oss, og hvor fiendene er. Spillet blir også turbasert, der hver spiller kan gjøre noen små avgjørelser, som for eksempel å flytte seg og angripe. I kamp blir vi nødt til å ta taktiske valg, og helst samarbeide om det. Hvilken fiende bør bli tatt ut først, og hvem kan vente. Det er her vi for alvor merker konsekvensene av hvordan vi har valgt å forme karakterene våre, både når det gjelder hvor mye skade vi gjør og hvilke evner vi har. De ulike rasene og klassene har også sine fordeler og ulemper i ulike situasjoner.

Spesialiteten til jegeren, slik som karakteren min, er å skyte med bil og bue og bevege seg raskt. Dum som jeg var valgte likevel jeg å angripe zombiene med sverdet mitt, etter at de tre andre hadde gjort det samme. Dette resulterte i at jeg endte oppe med minus to liv. Når du kommer til null i Dungeons and Dragons blir du bevisstløs, men du kan fortsatt overleve med hjelp fra de andre. For hver runde som går mister du et og et liv, om du ikke klarer å stabilisere deg.

Et kaldblodig svik

De tre andre klarte endelig å eliminere fienden mens jeg var bevisstløs. Plutselig, som lyn fra klar himmel, hentet fortiden meg igjen: Mithrandor viste sitt sanne jeg og erklærte at han ville drepe meg. Selv om jeg var halvdød ble jeg skikkelig fornærmet. Hva hadde jeg noensinne gjort mot ham?

Det viste seg at Mithrandor var misfornøyd med at jeg var mye rikere enn alle andre, og siden jeg var bevisstløs ble det kaldblodige mordet utført før noen kunne hindre ham.

William kunne ikke tro hva som hadde skjedd og truet Mithrandor på livet, men Gladriel prøde å utnytte situasjonen ved å tilby trollmannen asyl i bytte mot halvparten av pengene. Hennes egoistiske handlinger straffet seg da hun mistet kontakten med guden sin og ble fratatt sine magiske evner.

William trådde frem i rettferdighetens navn og annonserte død over Mithrandor og hans kyniske handlinger. Mithrandor hadde ingen sjanse i en kamp mot krigeren, og resultatet ble et halvert reisefølge og 500 sølvmynter å dele. Man kan si at stemningen var noe amper. Det gikk ikke lenge før de innså at det hadde vært bedre med et fullt reisefølge. Et nytt monster, større enn noe de har møtt så langt, sto i veien.

Det så ikke så verst ut en stund, men så dør William, og Gladriel sto alene mot monsteret. Uten at hun vet hvor mye liv motstanderen har igjen, prøvde hun forgjeves å slå mot monsteret, men bommet hver gang. Etter flere forsøk måtte hun se seg slått av monsteret, og med det endte eventyret vårt brått og tragisk. Det hadde nok gått bedre om Mithrandor ikke hadde tatt mitt liv, eller jeg hadde delt pengene mine med de andre. Det er vel en lekse å lære her et sted.

Mersmak?

Dungeons and Dragons lar deg være noen andre enn den du egentlig er. Å spille en karakters personlighet fullt ut er mye av underholdningen, samtidig som at det gjør spillet mye mer nyansert. For min del føler jeg at det ville vært interessant å være mindre lik min faktiske personlighet neste gang jeg spiller, slik at reisefølget blir mer variert og har mer interessante interaksjoner. Alle mine medspillere er veldig greie folk i virkeligheten, og det at ikke alle hadde samme type motivasjoner gjorde spillet annerledes. Det er langt mer omfattende enn andre tradisjonelle brettspill, men samtidig er det utrolig spennende opplevelse hvor man knytter sterke bånd til både sin egen og andres karakterer. Jeg vil helt klart spille dette igjen.

ÅRESPILLET! BETA

Regler:

Beveg deg rundt på brettet med selvvalgte brikker og en terning. Retning velger du selv, men reglene i hver rute må følges. Første-mann til mål!

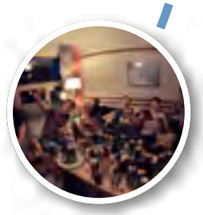
START: Velkommen til ÅRE!

KOM DEG UT: Du blir kastet ut av hytta. Tilbring tre runder på fanget til en motspiller

SKÅL: *Alle skåler.*

HACKER: Du fant ut wifipassordet. Lag din egen regel.

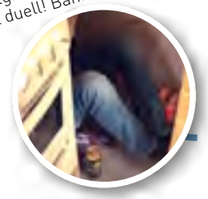
INVASJON: Du invaderer HS-hytta ett ekstra kast.



STRIP TEASE: Det strippes i leilighe-ten! Spilleren til venstre bestem-mer hvilket plagg du skal ta av.

FYSJ!: Romkameraten din kastet opp i stua. Bruk en runde på å vaske opp

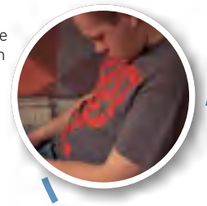
UNDER BORDET: Du må utfordre en valgfri motspiller til duell! Bånnski!



PASS OPP!: Du får klager på hytta, og får Jim på nakken. Løp frem to plasser!

SCORE!: Du sjekker opp en russejente. Drikk opp ølen før du setter deg på bussen

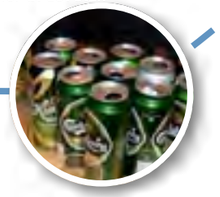
SOVNA: Du er for trøtt for dette spillet... Syng en valgfri sang fra start til slutt



STENGT: System-bolaget har ikke åpnet enda! Vent en runde, og start neste med en seiers-shot

MAXBURGER!: Du spanderer max-burger på alle rundt deg. Skål med hver spiller rundt bordet.

TOMT: Her var det tomt gitt, lag et høyt tårn, og sørg for at det står.



SVENSKE **:** Du slo ned en svenske, tilbring neste runde på glattcelle.

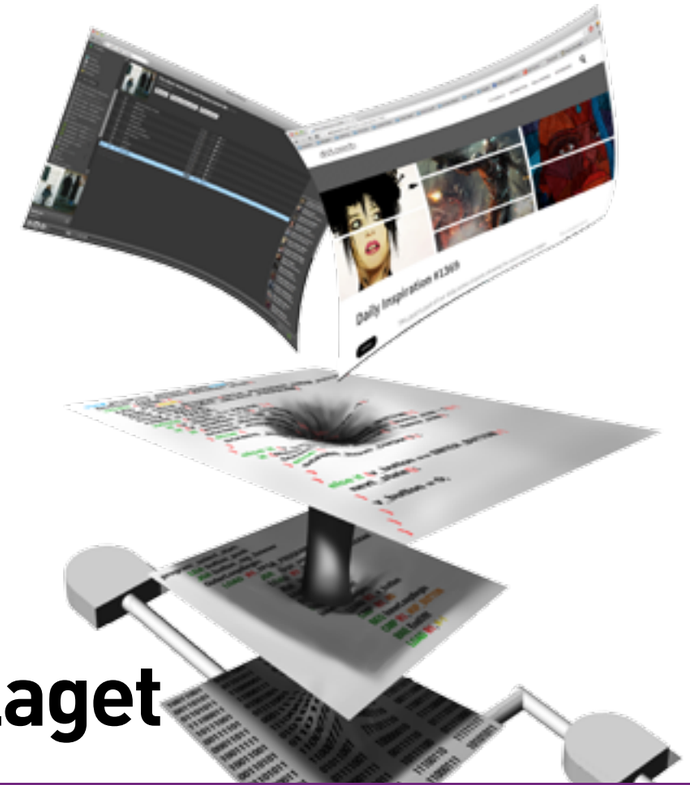
SKI?: Du har da ikke tid til det?! Bussen går om en halvtime! Slå en sekser for å rekke bussen

MÅL **GRATULERER:** Du fikk plass på bus-sen hjem. Takk for i år!

ifl[terning.getValue == 6]{continue}

NOPE!: Du slipper ikke inn og må ta taxi hjem. Rykk tilbake til start

7 kilometer unna...



Gjennom abstraksjonslaget

Programmering med språk som Java og C kan noen ganger virke litt magisk. Vi vet masse om kontrollflyt, beregninger og variabler, men likevel, hvordan kan datamaskinen gjøre mening av dette? Hvordan gjør datamaskinen *egentlig* alt det vi ber den om?

TEKST: ERIK LOTHE

ILLUSTRASJON: ERIK LOTHE, MAGNUS LINE

Mange opplever sine første møter med lavnivå-programmering som noe ubehagelig. Å forlate den trygge, intuitive og abstrakte verdenen av objekter, metoder og kontrollstrukturer virker for mange mer som et tiltak enn å være til nytte, og som regel er det faktisk det. Men når vi forstår utførelsen på lavt nivå kan vi også forstå hvordan det hele henger sammen, og en helt ny verden av forståelse åpner seg. For å få til dette skal vi gjøre et dykk gjennom abstraksjonslaget. Vi starter med et enkelt C-program som vi lar kompilatoren bryte ned til assembly-kode, og deretter til maskinkode, og på bunnen skal vi se på hvordan maskinvaren til slutt vil utføre disse instruksjonene. Eksempelprogrammet vårt er såpass enkelt at det ville sett nesten identisk ut om det var skrevet i Java, så ingen C-kunnskaper er nødvendig. Kjøring av Java-kode inneholder noen litt uvanlige ekstra steg som vi ikke skal gå innpå

Steg 1. På toppen

For at vi skal klare å holde oss til poenget vil vi ikke blande inn noe operativsystem. Det er faktisk helt unødvendig for å kjøre eksempelprogrammet vårt. Programmet vi skal se på skal kun gjøre et såkalt 'busy wait' over 10 iterasjoner, med andre ord bare telle til 10.

```
void main() {
  int counter = 0;
  int steps = 1;
  while (counter < 10)
  counter += steps;
}
```

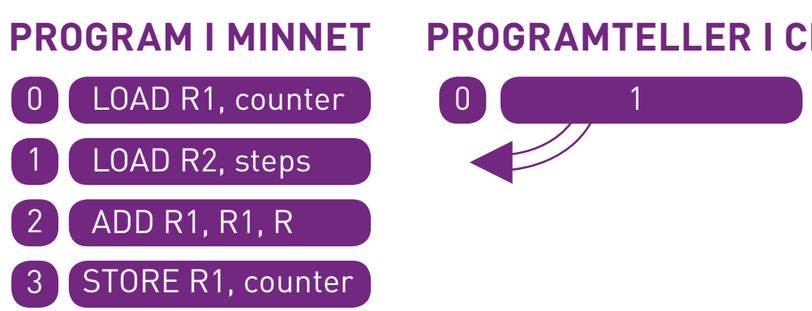
Vi skal nå kjøre kildekoden til dette lille C-programmet gjennom en samling programmer som steg for steg oversetter den til maskinkode. Det første steget innebærer å kompilere fila til assemblykode. Hvordan kompilatoren gjør dette er en vitenskap i seg selv, men ved å se på én av flere mulige måter forstår vi det overordnede.

Steg 2. Assembly

Assemblyspråk er ikke standardiserte slik som programmeringsspråk pleier å være. Siden assemblyspråk «jobber» så tett på prosessoren varierer disse språkene, også kalt instruksjonssettene, avhengig av hva slags prosessor du har. I eksemplene vi skal se på her skal vi bruke et fiktivt assemblyspråk som er mest mulig lesbart samtidig som det ligner på assemblyspråk flest.

Konvensjoner fra 40-tallet

De aller første programmeringsspråkene som ble oppfunnet for digitale datamaskiner var assemblyspråk, og konvensjonene er arvet fra en helt annet tid. Konvensjonen i assemblyprogrammering er å gjøre koden så kort som mulig med voldsomme forkortelser som ikke alltid er så intuitive. På denne måten tok kildekoden svært lite plass og dermed kompilerte raskere, samt at man kunne programmere svært raskt i fravær av autofullfør. Men for å gjøre eksemplene her mer lesbare har jeg derfor brutt med konvensjonen og valgt betydelig lengre nøkkelord enn det som er normalt.




```
LOAD  R1, counter   // 1. Hent counter, legg det i R1
LOAD  R2, steps     // 2. Hent steps, legg det i R2
ADD R1, R1, R2      // 3. Addér R1 og R2, legg resultatet i R1
STORE R1, counter   // 4. Lagre resultatet (R1) tilbake til minnet
```

1

Trikset med assembly er å bryte en «komplisert» operasjon som dette ned til flere enkle steg satt sammen. Stegene vi trenger her er som følger: Steg 1 og 2, hent variablene *counter* og *steps* fra minnet. Steg 3, pluss dem sammen, og 4, lagre resultatet tilbake i minnet på plassen vi hentet *counter* fra. Hvis vi skal skrive nettopp dette i assembly, kan det gjøres som i kodeboks 1.

Entydige nøkkelord

OP-kodene og registernavnene er hardkodet inn i språket, og vil alltid ha en entydig oversettelse til maskinkoden. Vi kan også skrive konstanter og minneadresser inn i stedet for registernavn. Adresser, tall og nøkkelord er alltid representert av et entydig tall og assembleren kan oversette dem direkte til maskinkode. Alt annet enn slike nøkkelord, som for eksempel *counter* og *steps*, ville ikke hatt noen direkte betydning for prosessoren. Disse variabelnavnene ville man vanligvis kalle for *labels*. Et label er ikke noe mer enn en plasshолder for et tall, og i dette tilfellet er tallet en minneadresse for hvor innholdet i variabelen befinner seg. Assembleren (programmet som oversetter assemblykode til maskinkode) vil i første steget av assemblingen erstatte alle labels med deres respektive minneadresse. Programmereren (eventuelt C-kompilatoren) vil da på et eller annet sted i assemblykoden være nødt til å skrive noe som

Syntaks

En assemblyinstruksjon består av en OP-kode, etterfulgt av ingen, ett eller flere operander. OP-kode står for operasjonskode og kan sees på som en enkel funksjon som CPUen har hardkodet støtte for å utføre. Eksempler på operasjoner og deres OP-koder er aritmetiske operasjoner som ADD, boolske operasjoner som COMPARE, kontrollflyt-operasjoner som BRANCH_IF_NOT_EQUAL og minneoperasjoner som LOAD og STORE. Operandene kan angi inputen til funksjonen, og eventuelt hvor outputen skal plasseres. Konvensjonen er OP-*kode, output, input* 1, *input* 2. For eksempel vil instruksjonen ADD R1, 1, 2 lagre 1+2 i registeret R1. Registrene (prosessorens interne variabler) som vi skal bruke er merket med R1 og R2.

Aritmetikk (counter += steps)

Når vi skal kompilere C-programmet vårt er det naturlig å dele det opp i to forskjellige eksempler: Adderingen og while-løkka. Det første vi skal oversette til assembly er den enkleste delen av programmet vårt, adderingen som skjer inni while-løkka, nemlig

```
counter += steps;
```

Abstraksjon: Strukturert sammenslåing av mindre og enklere komponenter for å produsere større og mer kompliserte systemer der implementasjonsdetaljene kan glemmes. Et høyere abstraksjonnivå kjennetegnes av mer menneskelige og intuitive konsepter. Et lavere abstraksjonsnivå kjennetegnes av en større mengde elementære og konkrete detaljer.

Kompilator: Sett med programmer som stegvis oversetter kildekode til maskinkode.

Programteller: CPU-register som peker til minneadressen til instruksjonen som skal utføres. Ofte forkortet PC (Program Counter).

Kontrollflyt: Kontroll over rekkefølgen for utføring av instruksjoner. Eksempler fra Java er metodekall og strukturer som if, while, for, osv.

Buss: Et sett signaler som brukes til å overføre data. En buss er ofte delt mellom flere enheter. Består som regel av flere signaler/kabler for parallell overføring.

CPU: Central Processing Unit. Prosessoren.

ALU: Arithmetic Logic Unit. Delen av prosessoren som utfører beregninger.

```
  LOAD        R1, counter             // 1. Hent counter, legg den i R1
  LOAD        R2, steps               // 2. Hent steps, legg den i R2
Count_again:
  ADD         R1, R2, R1              // 3. Addér R2 og R1, legg resultatet i R1
  //STORE     R1, counter           // 4. Lagre resultatet (R1) tilbake til minnet
  COMPARE     R1, #10               // 5. Sjekk om R1 og tallet 10 er like
  BRANCH_IF_NOT_EQUAL   Count_again   // 6. Branch til ADD-instruksjonen, hvis den
                                      // forrige COMPARE-instruksjonen gav NotEqual
```

2

forteller assembleren hvilken minneadresse hvert label skal erstattes med.

Kontrollflyt

De neste eksemplene har med kontrollflyt å gjøre. Kontrollflyt implementeres i prosessoren ved å manipulere programtelleren, men i assembly er vi gitt et par OP-koder som gjør dette for oss. Et program er som kjent en samling instruksjoner, og instruksjonene vil ligge utover i minnet i den samme rekkefølgen som assemblyinstruksjonene har. Hver instruksjon vil ha en egen minneadresse på samme måte som variabler har. Når programmet starter å kjøre settes programtelleren til minneadressen hvor den første instruksjonen ligger. Hver gang en instruksjon er ferdig utført vil programtelleren øke til den neste minneadressen og den neste instruksjonen vil bli utført. Ved å manipulere programtelleren kan vi dermed hoppe rundt i koden som vi vil. Dette kan vi utnytte til å for eksempel hoppe over en blokk med kode, hoppe bakover til kode vi allerede har utført, eller hoppe til et helt annet sted, for å senere hoppe tilbake. Den oppmerksomme leser vil se at dette kan brukes til å implementere henholdsvis if-løkker, while-løkker og funksjonskall. Vi skal nå se litt nærmere på hvordan dette kan gjøres.

While-løkken

Før vi går videre må vi få et begrep på plass. Å hoppe til et annet sted i koden kalles vanligvis branching. Branching kan skje *ukondisjonelt* (vil alltid skje), eller *kondisjonelt* (vil skje dersom en betingelse er tilfredsstilt). En if-løkke er et eksempel på en såkalt *kondisjonell* branch. Dersom betingelsen blir tilfredsstilt fortsetter vi videre inn i if-løkkens innhold. Dersom kondisjonen *ikke* tilfredsstilles må vi hoppe over innholdet av if-løkken og vi brancher til den første linjen med kode *etter* innholdet i if-løkken. Altså et 'branch if not equal'-tilfelle. Et litt mindre forvirrende tilfelle av kondisjonell branching ser vi i en while-løkke:

```
while (counter < 10)
  counter += steps;
```

For å implementere while-løkken fra programmet vårt i assembly starter vi, som i forrige eksempel, med å hente variablene våre fra minnet og deretter plusser vi dem sammen. Forskjellen er at nå skal vi legge på følgende steg. Steg 5, sjekk om counter er lik 10, og steg 6, hvis sammenligningen i den forrige instruksjonen var ulik, eller «not equal», skal vi hoppe tilbake og gjenta de to siste instruksjonene. Se kodeboks 2.

Labels

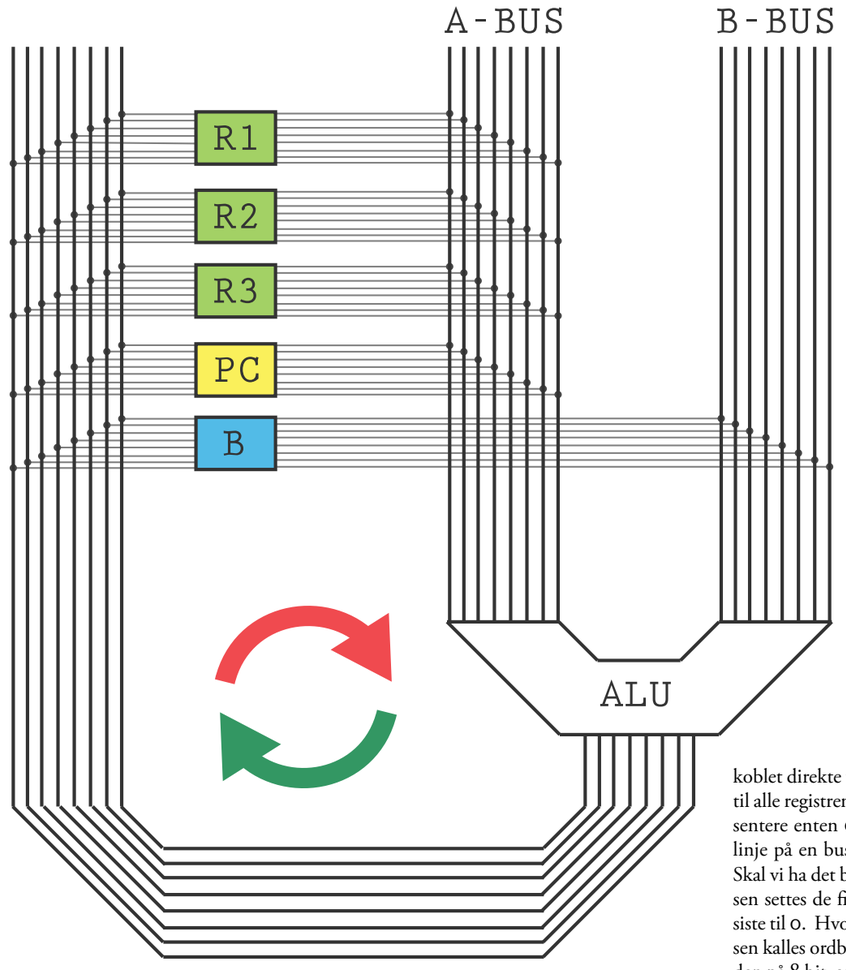
Count_again: er ikke en kommentar, og heller ikke en instruksjon, men et label. Når labels brukes på denne måten, altså utenfor instruksjonene og etterfulgt av et kolon, vil assembleren automatisk knytte det til minneadressen til den neste instruksjonen i programmet. I dette tilfellet kommer ADD-instruksjonen rett etter labelet og det er denne instruksjonens minneadresse som vil være adressen til labelet *Count_again*. I den siste linjen vil vi, dersom kondisjonen er tilfredsstilt, sette programtelleren til adressen som *Count_again* representerer (ADD-instruksjonen), og programmet vil fortsette å utføre derfra.

Du har kanskje lagt merke til at vi har kommentert ut steg 4, og ikke lenger lagrer *counter* tilbake til minnet. Siden vi ser at counter-variabelen aldri vil bli brukt mer kan vi hoppe over dette tidkrevende steget.

Steg 3.Under abstraksjonslaget

Variabler, slik vi kjenner dem fra Java og C, representerer minneadresser. Når vi leser verdien til en variabel, aksesserer vi adressen og henter verdien som ligger der. I variablenes tilfelle peker denne adressen helst til et sted i minnet, men prinsippet gjelder også for adresser som peker til for eksempel en harddisk eller et skjermkort. Minnet er ganske raskt etter våre tidsoppfatninger, men fra CPU-ens perspektiv tar en minneaksess en evighet. For å unngå så mye dødtid jobber prosessoren derfor helst ikke direkte med minnet, men henter heller på forhånd ut variablene den trenger og legger dem i et lite sett med registre inni CPU-en som den har rask tilgang til. De fleste prosessorer pleier å ha i størrelsesorden 10 «general purpose» registre som den kan bruke til hva den vil, i tillegg til et fåtall spesifikke registre slik som programteller, stakkpeker, statusord og andre ting vi ikke skal gå innom nå.

Alle disse registrene er koblet til enten én eller to busser internt i CPU-en som går direkte til ALU-en, der selve databehandlingen skjer. Den andre siden av ALU-en er koblet til en annen buss som går tilbake til registrene. ALU-en kan derfor jobbe direkte med registrene og det er denne flyten av data fra registrene til ALU-en og tilbake til registrene igjen som er selve hjertet i CPU-en.



Klokkesignal

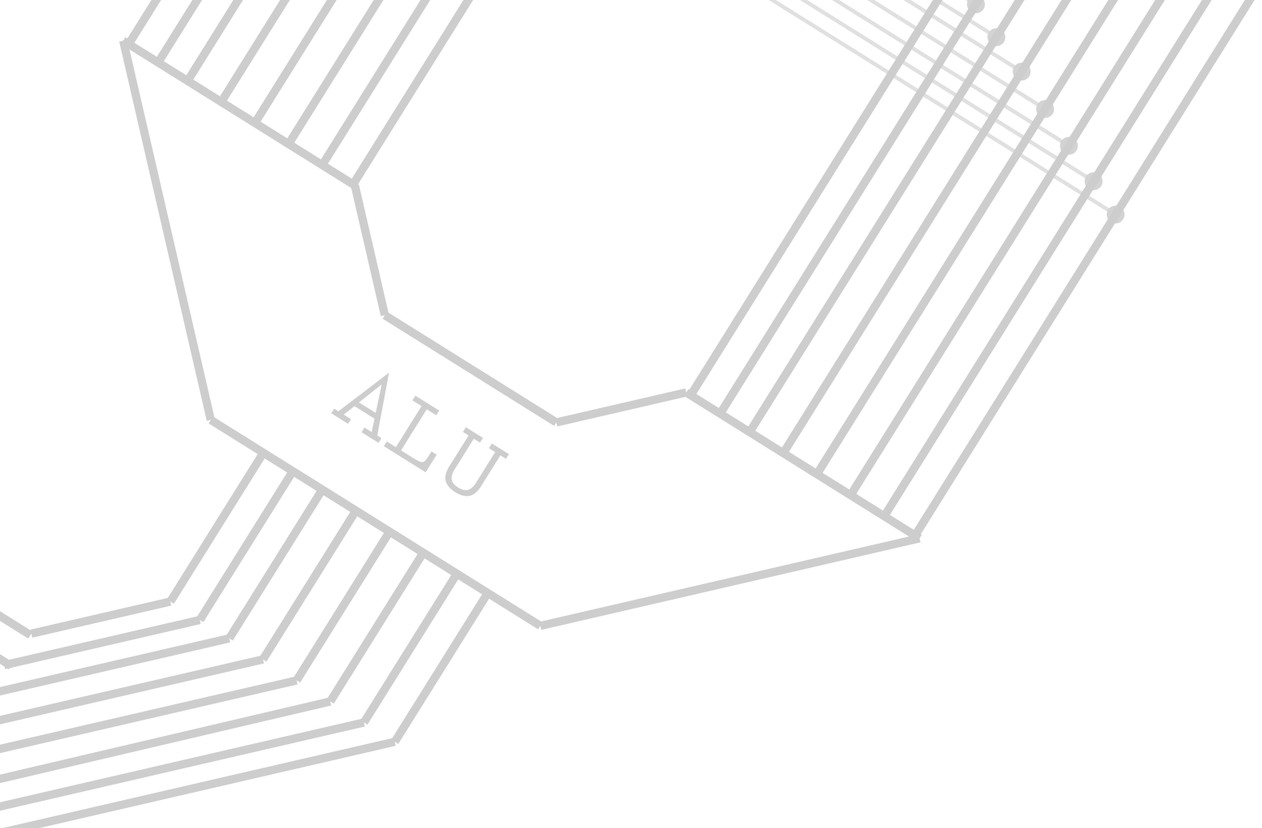
Alt dette skjer i enorm hastighet og er til syvende og sist kun begrenset av lysets hastighet og antall kretser et signal må kjøre gjennom for å bli behandlet. Men for å holde orden på signalene times alt som skjer av et klokkesignal, et signal som går av og på i en fast frekvens. En assemblylinje kan i beste fall utføres i løpet av det vi betegner som én klokkesykel, tiden det tar for signalet å skru seg på og deretter av. Med en 2 GHz prosessor snakker vi 2 milliarder klokkesykler i sekundet, altså opptil (dog usannsynlig) to milliard beregninger på et lite sekund.

Flyten av data

Vi kan tenke oss at det er to busser som går inn til ALU-en: A- og B-bussen. A-bussen er koblet på bl.a. R1 og R2-registrene våre, B-bussen er koblet på et annet register som vi kaller X. Hver buss går til en av ALU-ens to innganger. Disse busslinjene er koblet på et sett med logiske kretser inni ALU-en, der forskjellige subset med kretser kan utføre forskjellige oppgaver, for eksempel addisjon. ALU-en har to innganger fordi de logiske kretsene som for eksempel adderer to tall, er avhengig av at begge tallene kommer inn samtidig. Den andre siden av disse kretsene er koblet direkte til utbussen, som igjen er koblet til alle registrene. Siden en linje kun kan representere enten 0 eller 1, kan hver individuelle linje på en buss kun overføre én bit av tallet. Skal vi ha det binære tallet 11110000 på bussen settes de fire første linjene til 1 og de fire siste til 0. Hvor stort tall det er plass til på bussen kalles ordbredden, og i dette eksempelet er den på 8 bit, som var svært vanlig for noen tiår siden. Moderne datamaskiner pleier å ha 32 eller 64 biters ordbredde.

ADD R3, R2, R1

For at ALU-en skal ha mulighet til å addere verdiene i R1 og R2, trenger den som nevnt tilgang til begge registrene samtidig, et tall i hver av de to inngangene. Siden det er kun registeret X som er koblet på høyresiden av ALU-en (gjennom B-bussen) må vi bruke den første klokkesykelen til å flytte R2 over til X. Deretter legger vi R1 på A-bussen og X på B-bussen. Klokkesignalet går opp, og signalene strømmer inn i ALU-inngangene gjennom en rekke logiske kretser som resulterer i en binær addisjon. Når klokkesignalet har gått ned strømmer summen av tallene ut på baksiden, og registeret R3 kan lese og lagre resultatet som nå ligger på utbussen.



Signalene flyter altså ut fra registrene R1 og X, inn i ALU-en, gjennom noen av ALU-ens mange logiske kretser og til slutt tilbake til R3. Hverken registrene eller ALU-en har egentlig gjort noe som helst. De er bare blitt satt opp på en måte som gjorde at det elektriske signalet fløt gjennom en kombinasjon av koblinger, som i dette tilfellet resulterte i en addisjon av register R1 og X.

Kontrollenhet

Alle disse koblingene settes opp av en kontrollenhet som er styrt av mikroinstruksjonene. Mikroinstruksjonene er maskinkoden som ble oversatt fra assembly og er det laveste nivået av kode. Oversettingen er veldig rett fram, og hver assemblylinje pleier å ende opp som en eller to mikroinstruksjoner. Softwaremessig er vi nå helt på bunnen av abstraksjonsspekteret. Nå som vi har sett hvordan CPU-en utfører operasjoner er det kanskje litt enklere å forstå hvorfor assembly består av så mange små steg. Det er nettopp fordi CPU-en er så enkel at den trenger å jobbe i veldig mange små steg. Men å faktisk kode på denne måten er tidkrevende og gjør det svært vanskelig å holde oversikten når programmets kompleksitet begynner å vise seg. Nøkkelen til håndtering av kompleksitet er abstraksjon, og det er her kompilatoren kommer inn i bildet.

Små og store byggeklosser

Med kun sjeldne unntak som ekstreme sikkerhets- eller ytelsessensitive krav, har vi sjeldent bruk for å spesifisere hver eneste CPU-instruksjon. Disse byggeklossene er rett og slett for små og vi kan like gjerne sette sammen noen av dem en gang for alle og la dem forme et sett med litt større og svært mye penere byggeklosser. Man kan sette sammen et sett med flere små instruksjoner til intuitive kontrollstrukturer som for eksempel if-, while- og for-løkker, og med god samvittighet glemme detaljene bak. I tillegg kan man forenkle vanlige prosedyrer som variabelaksess og dermed skape kraftige abstraksjoner som datatyper, arrays og objekter. Dette er kort sagt det en kompilator gjør. En kompilator leser inn kildekode der instruksjonene er skrevet med et sett med byggeklosser, syntaksen i programmeringsspråket, og bryter dem ned til enda mindre byggeklosser, assemblyinstruksjoner for akkurat din CPU.

Abstraksjonen godtas

I programmeringsmiljøet på 50-tallet var holdningen at kompilatorer aldri ville kunne produsere like optimaliserte programmer som en assemblyprogrammerer. Men kompilatorene har utviklet seg voldsomt, og har med tiden fått en enorm evne til å produsere høyst optimalisert kode. I dag har stort sett alle godtatt bruken av dette abstraksjonslaget som den beste måten å produsere kode på i stort sett alle tilfeller. Med ulempene mer eller mindre borte er det nesten bare fordeler som står igjen: Kraftige abstraksjoner som tar konsentrasjonen din vekk fra trivielle detaljer rundt for eksempel datatyper, kontrollstrukturer, kodeoppdeling, osv. og som tillater deg å fokusere på hva du faktisk vil at datamaskinen skal gjøre.

Denne sjangeren møtte mye motstand da den så lyset på begynnelsen av 80-tallet og er fortsatt ikke særlig sosialt akseptert. På lik linje med sult i Afrika mener vi at dette er et problem som bør belyses og presenterer herved stolt, topp ti erotiske thrillere:

OFFLINES TOPP 10

01 **Eyes Wide Shut** (Stanley Kubrick, 1999)
02 **Body Double** (Brian De Palma, 1984)
03 **Femme Fatale** (Brian De Palma, 2002)
04 **Basic Instinct** (Paul Verhoeven, 1992)
05 **Crash** (David Cronenberg, 1996)
06 **Dressed to Kill** (Brian De Palma, 1980)
07 **Cat People** (Paul Schrader, 1982)
08 **Wild Things** (John McNaughton, 1998)
09 **Fatal Attraction** (Andrian Lyne, 1987)
10 **De Vierde Man** (Paul Verhoeven, 1983)

Etter 20 år i kalde nord var jeg lei av lusekofter og gløgg. Det var på tide å oppleve noe nytt. Jeg kastet skiene i peisen og heiv meg på første fly til Sør-Afrika. Etter tjuetre kjappe timer var jeg fremme i Durban. Her er mitt reisebrev:

ET REISEBREV FRA DET TRADISJONELLE SØR-AFRIKA

TEKST OG FOTO: KARI F. SKJOLD

Når du kommer utenfor byene i Sør-Afrika er avstandene fra bygd til bygd stor, allikevel spaserer det alltid noen langs veikanten – alt fra gamle koner som balanserer krukker på hodet til unge mennesker av den mer moderne generasjonen. De fleste av dem haiker, noen prøver å selge ting og andre tigger. Uansett hvor langt ut i ødemarken du kjører, er det alltid mennesker der, selv om neste hus er flere mil unna. I dagene jeg tilbrakte på "landet" kom jeg i kontakt med en lokal jente ved navn Kagiso. Hun lever i en verden der døtre og kuer er de eldres livsforsikring, der varebytte uten penger praktiseres og der flerkoneri slettes ikke er unormalt. Hvor lykkelig du er dreier seg i stor grad rundt hvorvidt du har hell eller uhell i livet.

Kagisos landsby består av inngjerdede beiteområder, samt små, runde jordhus som befolkningen bor i. Runde hus tyder på en veldig tradisjonell arkitektur, dersom du har et firkantet hus vil du vise at du fører en mer moderne livsstil. I landsbyer som denne blir mennene alltid servert først – noe annet er uhørt – og i stedet for å handle med penger, bytter de varer seg imellom. Melk fra kuer eller geiter kan byttes mot hjemmedyrkede grønnsaker og så videre. De bytter alltid det de har til overs.

Kriminalitet

Siden det virket som om de fleste husene pent sagt hadde "åpne løsninger" og siden alle spaserte overalt, spurte jeg Kagiso om kriminialitet er et problem. Da ristet hun raskt på hodet og fortalte at selv den minste kriminelle handling vil føre til forvisning på livstid. Her får man "no second chances". Det virker som om befolkningen på landsbygda lever i sin egen lille boble, der det er deres egne regler som gjelder. Forvisning og den slags straffer blir bestemt av den eldste mannen i landsbyen – han er den overordnede og tar alle slike beslutninger.

Språk

Zulubefolkningen snakker et språk jeg aldri har hørt maken til. Når de snakker, hører du at de lager et slags "klikk" etter flere av ordene. Disse klikkene har stor betydning; dersom du glemmer et, kan setninger/ord få en helt annen betydning enn det som var ment. Kagiso fortalte en historie for å illustrere hvor fatale følger slike "klikk"-feil kan føre til: I en av de lokale restaurantene hadde kokken sagt "hent alle eggene til hønene" til en av sine medarbeidere. Ordet "eggene" hadde manglet et klikk i uttalelsen og hørtes dermed ut som "hodene" i stedet. Medarbeideren, lydig som han var, hadde hugget av alle hønenes hoder og levert de på kjøkkenet. Jeg tror det var slutten både på guttens servicekarriere og på restauranten.

Krigsdans

Helt fra de kan å gå, lærer zulubefolkningen en spesiell dans som preger resten av deres liv. Dansen er en slags krigsdans som har som mål å skremme fiendene. Hovedtrinnene er å trampe opp og ned med foten, der det er om igjen å få beinet høyest mulig opp i luften og deretter hardest mulig ned i bakken. Det er en dans med ekstremt mye energi i, og tro det eller ei, den virker faktisk skremmende. Guttene/mennene danser denne dansen hele livet, mens jentene slutter når de gifter seg. Jentene danser toppløse, men pupper er ikke forbundet med noe seksuelt når jentene er ugifte. Begge kjønn har et skjørt laget av kuskinn som antrekk. Leggene deres er også dekket til; av saueull, i den hensikt at ullet skal beskytte mot slanger osv. De går derimot barbent: Huden under føttene er tykk og trenger ikke beskyttelse. Guttene og jentene danser stort sett separert. Mens jentene opptrer, klapper guttene eller slår på trommer for å skape rytme.

En kveld var jeg på et danseshow laget av en gruppe ungdommer. Dansen deres inneholdt disse tradisjonelle trinnene, samt noen sanginnslag og ekstra dansetrinn de hadde kommet opp med selv. Gruppen viser dette showet ca fire ganger i uken for turister som bor på hotellområdet, og i gjengjeld blir de sponset med bøker og annet utstyr de trenger til skolen. Selve skolen er betalt av staten, men tilbehøret er det stor mangel på.

Zulukongen

Zulukongen trenger visst litt mer "spenning" enn det kongen vår gjør. Han velger seg gjerne en ny kone hvert år – på tross av at han har rundet 70. Utvalget av kone skjer under en dans der de jentene som ønsker å bli valgt stiller opp og viser seg fram. Også i denne dansen brukes de tradisjonelle "krigstrinnene". Når dansen er ferdig, velger kongen om han ønsker en av dem som sin fremtidige kone. I dag har zulukongen åtte koner, og velger sannsynligvis sin niende nå i 2013. Hans yngste kone er bare litt over 20 år gammel. Det høres kanskje sykt ut med en aldersforskjell på 50 år og generelt det å ha åtte koner, men Kagiso sa at flerkoneri er mest vanlig blant den eldre generasjonen. "De fleste på min alder syns det hadde blitt for mye stress", sa hun og lo. Selv ønsker hun absolutt ikke å få seg en mann som må deles på.

Hell og uhell

Zulubefolkningen har en veldig spesiell tankegang når det kommer til hva som er greit og ikke greit å gjøre. For dem handler alt om hell og uhell. Dersom du har blitt president, har du hell i livet, og å prøve å "ta fra" noen hellet deres bringer uhell til en selv. Av denne grunn vil befolkningen neppe gjøre opprør mot de overordnede på noen som helst måte. For å se på et mer hverdagslig eksempel kan vi tenke oss at du og din bestevenn går langs veikanten. Vennen din går foran, og du legger merke til at han mister et par pengesedler ut av lomma. Er du en passe moralsk nordmann ville du ha plukket opp pengene og gitt de tilbake til vennen din. Er du zulu derimot, tar du gjerne pengene selv nettopp fordi du da har hellet med deg som "fant" de. Vennen din derimot, har uhell og det var nærmest hans skjebne å bli et par sedler fattigere den dagen. Dersom du kan ta noe uten å bli sett, er det helt greit. Ifølge zuluene er det forfedrene dine som gir deg hell eller uhell. Forfedrene har veldig stor betydning, de skal tilbes som om de var guder, og det er helt vanlig å legge fram mat eller andre gaver til dem.

Giftemål

Som zulu er det ikke bare-bare å gifte seg. Retningslinjene for hvordan dette skal foregå er ganske strenge. Som nevnt, har veldig mange av landsbybeboerne kuer som sin største innvestering og verdi. Når man skal gifte seg, er mannen – med støtte fra sin familie - påbudt å gi damens familie en brudegave på omtrent elleve kuer. Hver ku er verdt ca 8000 norske kroner så det er snakk om ganske mye penger med tanke på hvor dårlig zulubefolkningen tjener. Dersom den gifteklare jenta har utdannelse, må brudegaven være større – jenta blir "verdt" mer. Dersom hun har barn med en annen mann fra før av derimot, blir hun "verdt" mindre. For hvert barn hun har fra før av, skal brudegaven være 2 kuer mindre. Kagiso er barnløs og har en treårig utdannelse, så vi regnet ut at en mann og hans familie må betale litt over 100 000 norske kroner for å få gifte seg med henne.

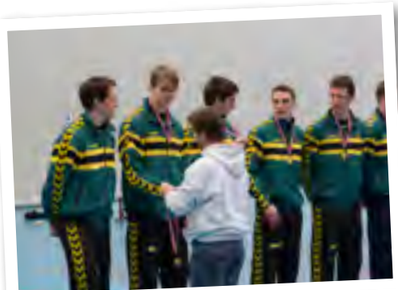
Bryllupsfest

Også selve bryllupsfesten krever store pengesummer. Du kan ikke begrense antall gjester ved å invitere kun den nærmeste vennekretsen – skal du ha et tradisjonelt zulubryllup må du regne med at hele landsbyen kommer! Ryktet går fort, og det er frekt å ikke ha nok mat til alle.

Siden kravene er så store fører det til at veldig mange par må vente flere år med å gifte seg. Nå er det faktisk akseptert å være samboere og å få barn før man er gift. Gjennomsnittsalderen for å gifte seg er 27 år.

Da det gikk opp for meg at jeg ikke var stort mer verdt enn gjennomsnittet på elleve kuer, bestemte jeg meg for at det var best å komme meg hjem og fullføre utdannelsen min i Norge før jeg finner min Zulumann. Dersom jeg øver inn noen dansetrinn, kan jeg kanskje sette meg som mål å kapre kongen om et par-tre år. Vi sees igjen!

HARD KAMP: Årets studentlekerker hadde deltakerrekord, med over 2000 påmeldte utøvere.



STUDENT OM DAGEN
ATLET OM KVELDEN

Den stereotypiske informatikkstudenten er nok ikke videre kjent for å drive aktivt med idrett. Under Studentlekene 2013 i Trondheim møtte jeg to som gjør det, Morten Lillehagen og Halvor Strand.

TEKST OG FOTO: THOR HÅKON BREDESEN

Morten Lillehagen går første året på bachelor i informatikk, og har spilt innendørs volleyball i fem år. Med sine 1.97 meter spiller han på førstelaget til NTNUI, som er i eliteserien.

– NTNUI er det eneste laget i Norge på elitenivå som er et rent studentlag. Vi er nyopprykkede og ligger allerede midt på tabellen. Dette er over all forventning; under Studentlekene 2013 er vi det eneste laget fra eliteserien. Det er unikt å ha et så sterkt lag i studentsammenheng, sier Morten.

Selvutnevnt favoritt

I forkant av studentlekene så Morten frem til å spille kamper som ikke er like alvorspreget som de i eliteserien.

– Siden det ikke er en like seriøs turnering, vil flere få prøve seg på banen og få viktig kamperfaring, spesielt de som vanligvis ikke starter på banen. Selv håper jeg å få prøve noen andre posisjoner på banen. Vanligvis spiller jeg midt, som er en vanskelig posisjon med mye fart og eksplosiv styrke, forteller Morten.

Vinneren av Studentlekene 2013 kommer videre til student-EM i Kypros. Morten liker ikke å være arrogant, men legger ikke skjul på at han tror at de er favoritter.

– Siden vi er det eneste laget i eliteserien tror jeg vi har gode sjanser for å vinne, sier Morten.

Vanskelig balansegang

For å spille i eliteserien, kreves det mye innsats. Morten trener volleyball fem dager i uken, spiller en til to kamper, i tillegg til noen styrkeøkter.

– Det er vanskelig å finne riktig balansegang mellom studier og volleyball. Første semester tok jeg meg mye fri fordi jeg var sliten. Å spille på elitenivå krever mye av hver spiller. Det er mye som skal tenkes på og presset er stort, men på den positive siden består laget kun av studenter og dermed vet alle hva det innebærer av arbeid. Det er betryggende.

Morten ser på seg selv som en sjelden art innen informatikk-verdenen. Han har møtt få som driver like aktivt med idrett.

– Idrett har alltid vært viktig for meg. Jeg klarer ikke å sitte stille for lenge, og må bevege meg i blant. Jeg har alltid vært høy, og syns det var greit å bruke høyden til noe fornuftig, derfor startet jeg med volleyball, sier Morten.

En annerledes idrett

Halvor Strand er på sitt siste semester på master i informatikk, og spiller squash under studentlekene. Han startet å spille sammen med en venn, fordi han ville prøve noe nytt og annerledes innen idrett.

– Det var lett å komme i gang og siden vi begge var jevngode var det gøy å spille. Squash er mest gøy, og kjennes egentlig ikke ut som trening. Jeg tenker mest på å spille gode kamper. At det er bra for kroppen er bare en bonus, sier Halvor.

Etter å ha spilt en stund meldte han seg inn i NTNUI Squash, og jobbet seg gradvis oppover for å bli bedre. Han er også involvert i styret til squash-gruppa.

– Det åpnet seg en stilling som dataansvarlig, så jeg søkte og fikk den. Jeg ønsket å engasjere meg i noe som betyr noe for meg, forteller han.

Under årets studentleker ser Halvor frem til å konkurrere mot andre gode studenter fra hele landet.

– Jeg liker turneringer, så denne måtte jeg nesten være med på. Studentlekene er jo en stor begivenhet i studentidretten. I tillegg til konkurransen er det sosialt og moro med banketten.

Under turneringen håpet Halvor å ende opp blant topp tre, men etter å ha kjempet seg til bronsefinalen, måtte han dessverre se seg slått, og endte på en fjerdeplass.

– Jeg hadde en god start på sluttspillet med to seire, før jeg tapte semifinalen mot han som, svært fortjent, vant turneringen senere. Så tapte jeg 2 - 3 i en svært jevn bronsefinale. Det var en litt sur slutt på turneringen. Jeg følte jeg presterte bra og var lettet over å ikke skulle spille mer squash den dagen, siden jeg var helt tom for krefter, sier Halvor.

Seier

NTNUIs elitelag i volleyball kom seg til finalen og vant 2 - 0. Morten er fornøyd med innsatsen til laget og syns det var gøy å være med, til tross for at motstanden ikke var like stor som i eliteserien.

– Vi fikk muligheten til å vise oss frem, det var gøy. Jeg er litt usikker på om jeg skal være med til Kypros for student-EM, siden det er midt i beach volley-sesongen, men vi får se, det hadde vært moro, avslutter han.

HØR OG SE

HJEMME HOS DAVID

Vi har tatt turen hjem til kontorsjef David Storjord. Mannen med planen. Planen for et bedre kontor, et bedre Online.

TEKST: MAGNUS LINE, ALEKSANDER SKRAASTAD
FOTO: MAGNUS LINE

En nydelig søndagsmorgen er Offline på besøk hjemme hos en lett avslappet kontorsjef iført morgenkåpe. Tente stearinlys og en atmosfære preget av dempet jazzmusikk og trivsel gjør at vi føler oss virkelig velkomne hjemme hos David, på Onlines nyoppussede kontor.

Et bedre liv

Det begynner snart å nærme seg ett år som beboer på kontoret for Trivselskomiteens sjef.
- Jeg flyttet inn i slutten av april ifjor. Før dette bodde en ved navn Håkon Giraff her, sympatisk type, forteller David. Det var en forholdsvis grei overtakelse av boet fra tidligere kontorsjef, men noen grep måtte han ta.
- Det var greit nok det, men var nødt til å vaske og lufte ut en del etter han, sier David med glimt i øyet.

David virker veldig avslappet, vi har inntrykket av at han trives godt her. Etter litt trivelig tomgangsprat ser vi oss nødt til å spørre om David faktisk er klar over at tidligere kontorsjefer aldri fysisk har bodd på kontoret, og undres over hva som forårsaket dette.
- Heh, du vet... David virker lettere forvirret.
- Først og fremst var det vel av praktiske årsaker. Det er mye lettere å komme seg på skolen, og jeg føler meg skjeldent ensom. Jeg er en ganske sosial person, og her har jeg besøk hele dagen, hver dag.

Vil ha forbud

Selv om David stortrives i sitt nye kontor er det ikke til å legge skjul på at det er noen kritiske mangler som han har bitt seg merke i.
- Jeg er glad i å se film, og har enda ikke et hjemmekinoanlegg her. Dessuten er det ikke til å legge skjul på at det har gått utover privatlivet mitt. Det er ikke så mange damer som har lyst til å bli med på skolen etter å ha vært på byen, uttaler han med en noe nervøs latter.

Med hjemmel i prinsippet om "fashionably late" bestemmer vi oss først nå for å gratulere David med oppussingen av kontoret.
- Å, jada! Det trengtes! Jeg følte at kontoret trengte et nytt image, sier han entusiastisk.
- Det har blitt mye mer åpent. Vi har gjenoppfunnet kjøkkenkroken og det har virkelig åpnet opp mer plass til stua og møtebordet. Svake minner om hvordan kontoret så ut før tegner en sterk kontrast til hvordan det er i dag, og vi kan med sikkerhet fastslå den ekstreme forvandlingen.
- Vi har også fått nye, låsbare komitéskap, og vi har personlige bokser for alle som vil ha. David stanser et lite øyeblikk for å ta en ny slurk av kaffen sin, og fortsetter:
- Vi har også fått nytt, fint møtebord, veldig corporate. Vi har heldigvis fått enda et kjøleskap også , vi trengte mer plass til øl.

Slår tilbake mot ryktene

En entusiastisk David vifter med armene da vi spør om hva det beste tillegget til kontoret er.
- Helt klart sofaen! Da vi pusset opp fikk vi en ny stor hjørnesofa, der nyter jeg ofte en flaske champagne og ser på den flotte utsikten.

David viser oss utsikten fra vinduene, noe vi må si var meget imponerende!

En ting vi har lagt merke til er at det mangler en mikrobølgeovn på kontoret.
- Jeg vil helst å ikke snakke om det, sier han vedmodig. Vi velger å la emnet ligge inntil videre.

I det vi forsøker å igjen gratulere med det nye kontoret og hjemmet til David blir vi skrekkelig klar over at David har mer å fortelle;
- Jeg ønsker at Online sine medlemmer skal ha flere matmuligheter på kontoret. Vi har begynt å selge lefser, nudler og annet i tillegg til det vi alltid har solgt. Sånn jeg ser det, er det viktig at studentene får i seg fem-om-dagen. En Billy's, en Gorby's, en pakke nudler, en øl og en kaffe så er du så å si i mål.

Kjærligheten varer lengst

Et dårlig kamuflert forsøk på å takke for seg opp blir avslørt av kontorsjefen, som fortsetter:
- Dessverre fantes det ikke penger i budsjettet til et nytt anlegg akkurat nå, men jeg kan love dere at det er planer om det i fremtiden.

Omsider må vi bryte igjennom og forlange å ta turen videre. Vi etterlater David henslengt på sofaen akkurat tidsnok til at morgenkåpen glir helt opp. Det er tydelig at det er hjemmekoselig på Onlines nye kontor.

kjøkken ~ HACKS

Eriks pizzapai

Skapkokk? Glad i mat?

Send oss ditt bidrag på:

redaksjonen online.ntnu.no

```
/**
* Meksikansk pizzapai.
* Ca 4 porsoner med glory.
* Paiform kan byttes ut med en vanlig stenform e.l.
* @author Erik Lothe
*/

import kitchen.*;
public class Pizzapai {
   Ingrediens ferdigPizzadeig;
   Ingrediens kjøttdeig = new Kjøttdeig(400, Enhet.GRAM);
   Ingrediens mais = new Mais(1, Enhet.BOKS, Størrelse.LITEN);
   Ingrediens tomat = new Tomat(3);
   Ingrediens løk = new Løk(1);
   Ingrediens hvitløk = new Hvitløksfedd(2);
   Ingrediens egg = new Egg(3);
   Ingrediens crèmeFraîche = new CrèmeFraîche(0.3, Enhet.LITER);
   Ingrediens ost = new Gulost(200, Enhet.GRAM);
   Ingrediens tacokrydder = new Tacokrydder();
   Ingrediens sennep = new GrovSennep(1, Enhet.SPISESKJE);
   Ingrediens salt = getSalt();
   Ingrediens pepper = getPepper();
   Form form = getPaiForm();
   Ovn ovn = getStekeovn();
   Stekepanne stekepanne = get Stekepanne();

   public Pizzapai() {
      pizzadeig = new pizzadeig();
      ovn.setTemperatur(200, Enhet.CELSIUS);
      form.add(pizzadeig);
      ovn.join();
      ovn.add(form);
      ovn.stek(36000000); // 10 min
      stekepanne.add(kjøttdeig, løk, hvitløk, tacokrydder,
      mais, tomat);
      stekepanne.stek();
      bolle.add(egg, crèmeFraîche, salt, pepper);
      bolle.rør();
      ovn.join();
      form = ovn.getForm();
      form.add(pizzadeig);
      form.pizzadeig.spreUtover(sennep);
      while (stekepanne.hasInnhold()) form.add(stekepanne.pop());
      form.add(ost);
      ovn.add(form);
      stekepanne.stek(90000000); // 25 min
      }
   }
}
```

Refresh

I denne spalten oppdaterer vi deg kort om hva som har skjedd på IDI, hos Online og ellers i studiebyen siden forrige Offline.

Ledervalg på Samfundet

Samfundet holdt nylig ledervalg, men ingen ny leder ble valgt, da det manglet noen usle prosent for å nå 50%-grensen. Nytt valg skal avholdes i en ekstraordinær valgrunde.

Jubileum!

Både linjeforeningen Hybrida og Nabla feirer sine jubileer i disse dager.. Offline gratulerer så meget!

Ny leder for IDI

Det ryktes at det skal velges ny leder for Institutt for Datateknikk og Informasjonsvitenskap. Hvem dette blir er ennå uklart.

Gløsmestere!

Vel, nesten. Vi gratulerer Henning, samt redaksjonens egne representanter, Sverre og Aleksander med andreplassen i Gløsmesterskapet i Ingeniørspillet!

Ny nettside - stay tuned

Drifts- og utviklingskomiteen arbeider på spreng for å få ferdig neste versjon av Onlines nettside, som forhåpentligvis vil se dagens lys nå før sommeren. Vi gleder oss!

Abakus starter revy

Linjeforeningen for datateknikk, Abakus, skal for første gang starte sin egen revy. Dette kan bli spennende!

Linjeforeningsband!

Det arbeides med å opprette et linjeforeningsband! De har enda ikke noe navn, men dersom du er interessert, ta kontakt med Thor Håkon Bredesen for mer informasjon.

VelKom

Velkomstkomiteen er nå opprettet! Planleggingen av årets fadderuker er i gang. Husk å melde deg på som fadder for å skape en fantastisk fadderuke for de nye førsteklassingene. (Man trenger ikke være VelKom-medlem for å være fadder).

06 MARS — KURS **Testdreven utvikling med Steria**

07 MARS — BEDPRES **Simula**

13 MARS — KURS **BEKK-kurs del 2/2**

14 MARS — BEDPRES **Knowit**

19 MARS — BEDPRES **FINN.no**

21 MARS — BEDPRES **Visma Consulting**

17 APRIL — KURS **Kurs med Knowit**

25 APRIL — BEDPRES **BEKK**