Oppgave 4 Dokumentasjon

Først så tok jeg å skrev koden i en .c fil, og kjørte den for å se om det kom noen feil. Det var noen få ting som måtte til for at koden skulle kompiles. sånn som noen includes stdio stlib math osv. koden ville fortsatt ikke kjøre fordi den ikke fant pow(), så jeg man-paget pow i shell og fant ut at det trengtes -lm når man kompilet filene så jeg la det inn i makefilen min. Jeg tviler på at dette var en av "feilene" i koden, men er verdt å nevne.
Etter dette kompilet koden helt fint, og det kom ingen feilmeldinger og når man kjørte executablen så skjedde ingenting :)))))

Så det jeg startet med var jo å prøve å skjønne hva som skjedde, så jeg laget printf() på nesten alle ting som jeg ikke skjønte hva gjorde, og fant etterhvert ut av noen forskjellige ting. Jeg regnet jo med at det hadde noe med å gjøre med det siste tallet i andre kodeblokk av kredittkortet (som nevnt i oppgaven) så jeg prøvde å teste ut litt forskjellige funksjoner for å se om jeg kunne spotte noen feil. (For en kjip kode å lese..) Men når jeg printet ut den rare for-loopen for å "Calculate the cardnumber as a 64 bit integer" så fikk man et ganske stort minus tall, og så at for-loopen kjørte 5 ganger istedenfor 4 som den skulle. Husker ikke rekkefølgen jeg løste de forskjellige feilene på, men nevner 2 her og nå fordi det er de jeg har funnet så langt.

Den ene feilen var jo at når andre segment startet på 123x så skulle noe skje, og hvis tallet 'x' var 9 så kunne det bli feil. Det var fordi (cc->p->digit[cc->p->digit[3]/((cc->p->digit[3]-'0')%9)]) blander characters og tall, så man ville fått ascii-verdien av hvert tall, og noen operasjoner ville blitt utført, ihvertfall så er det en (% 9) på slutten som gjorde at (cc->p->digit[3]-'0') ville blitt '9' - '0' = tallet (9) også ville man gjort 9 % 9 som blir 0, også ville man tatt første del av den koden (cc->p->digit[cc->p->digit[3]) som er '9', så hvis man tar 9 / 0 så blir det feil, dermed var dette feilen, det med 9 % 9 = 0 også dele et tall med 0 går ikke i c. Dette var selvfølgelig hvis char *c = "42421239xxxxxxxx"
Jeg lar også mine printf() og kommentarer bli igjen i filen, så du kan se litt hvordan jeg har tenkt. (kanskje kjipt for deg å lese, men da ser du litt hva jeg har gjort/tenkt) måten jeg løste det på var å sette (j = cc->p->digit[3];) fordi alt man ville var jo å få tak i det 4.tallet i segmentet. Denne if-setningen triggres jo bare hvis "node" nr2 = "123x", Hvis man ikke hadde fikset det problemet her og prøvd å printe ut hva J er, så hadde man fått "Floating point exception(core dumped)".

Feil nr 2, fant jeg ut når jeg printet ut for-loopen og så at den ble gjort 5 ganger, FORDI man mallocer minne til pc->p altså neste "blokk" før man gjør noe med den, i while-loopen tidligere i koden. har også kommentert litt rundt det. Det gjorde at kreditkortnummere så ganske riktig ut, men den siste for-loopen la til en 0.0001 value til kortnummeret, noe som blir oversett pga liten verdi, (fjern den if-setningen jeg la til i while-loopen for å se hvordan det ser ut), det vil også si at jeg løste det problemet ved å lage en if-setning rett etter at i-pointern har inkrementert. Sånn at den ikke mallocer neste del. Det er jo ikke en optimal løsning, men en midlertidig fiks på problemet :D

Til slutt så kom jeg på at jeg kunne bruke valgrind, for å sjekke etter minnelekasjer, så jeg skrev i shell "valgrind ./main" og fikk vite at "total heap usage: 5 allocs, 2 frees, 1,120 bytes allocated", noe som fikk meg til å tenke på at man allokerer minne til pc->p uten å free() de, så hvis man husker å frigjøre alle de 3 "p'ene" på slutten så er problemet fikset.
free(cc->p->p->p);
free(cc->p->p);
free(cc->p);
også i denne rekkefølgen sånn at man klarer å få tak i de siste "blokkene" i linked listen

Til slutt så lagde jeg den utløpsdato sjekk-delen av oppgaven.
Hvor jeg da henter inn lokal tid, med include <time.h> og time_t variabel og struct tm, også fant jeg ut hvordan jeg fant måneder og år ved hjelp av den

innebygde structen.
Så valgte jeg å gå for strtok(), men det funket ikke på char *e derfor la jeg over hver bokstav fra e over i en char temp[]-array og fikk så gjort en strtok() på den, hvor jeg da hadde "/" som delimeter, så gjorde at første del av stringen ble måned, så lagret jeg den verdien i en int variabel, også tok jeg en strtok() til for å få neste del av stringen, altså år, og lagde en variabel for år.
Etter det var gjort så var det bare å lage noen if-setninger om utløpsår > inputår, osv osv. sjekk selve koden..

Det er selvfølgelig mange ting her som er helt unødvendig, sånn som å lage en char *i = (char*) e, int a og int n blir jo ikke brukt osv, men disse småfeilene har jeg ikke gjort noe med. Jeg legger ved 2 filer, hovedfilen som vil bli kompilet når du kjører make (som inneholder mange av de printf() jeg brukte og litt kommentarer, for å vise hvordan jeg tenkte, var litt usikker hvordan jeg ellers skulle vise deg det) og en fil som heter Oppgave4Clean.c, den inneholder for det meste det samme, bare at jeg har fjernet mye rot, med fremprovoserte feil også. for å kompile denne kan du da skrive noe som "gcc -o main2 Oppgave4Clean.c -lm", husk å få med -lm for pow :)

Måten jeg tenkte å fremprovosere feilene på var å printe j, i den for-loopen som sammenligner node nr 2 og "123x", og printe ut litt hva som skjer i den for-loopen som legger til tall for llCreditCard
fordi den kjørte jo 5 ganger (nå bare 4). Minnelekasjen var jeg ikke helt sikker på om jeg fikk fremprovosert, på denne måten.