Projekt iMalloc
Imperativ och objektorienterad programmeringsmetodik, 1DL221

Niclas Edenvin
Åke Lagercrantz
Andreas Lelli
Daniel Lindgren
Elias Lundeqvist
Jakob Sennerby

2012-11-09

Sammanfattning

Vi fick i uppgift att, i programmeringsspråket c, konstruera ett eget bibliotek för minneshantering.

Genom att dela upp uppgiften i mindre block och följa projektspecifikationen skapade vi en minneshanterare med flera alternativ för hur man vill behandla minnet.

Problemet löstes enligt specifikationerna och resultatet blev som väntat en effektiv minnesallokerare som med fördel kan användas av program som vill kunna bestämma själv på vilket sätt minnet ska användas.

Innehåll

1 Dagbok

1.1 Fredag 19 Oktober

Vi träffades hemma hos Niclas för att gemensamt gå igenom instruktionerna och förbereda oss för första projektveckan. Vi bestämde även att vi skulle använda oss av olika applikationer, tjänster och versionshanteringssystem som gör det enklare för oss att arbeta tillsammans samt hålla koll på hur vårt arbetsflöde fungerar. Vi valde att använda oss av

Git och github versionshanteringssystem
Trello strukturering av uppgifter och vem som gör vad
Tickspot tidsrapportering
Hipchat kommunikation och forum
Google Drive utkast av projektdagbok och övriga dokument

Vi började även skissa på en övergripande design för systemet för att hitta en lämplig uppdelning.

1.2 Måndag 22 Oktober

Vi gjorde klart den övergripande designen. Se Figur 1. Sedan bestämde vi oss för en gemensam kodstandard enligt följande:

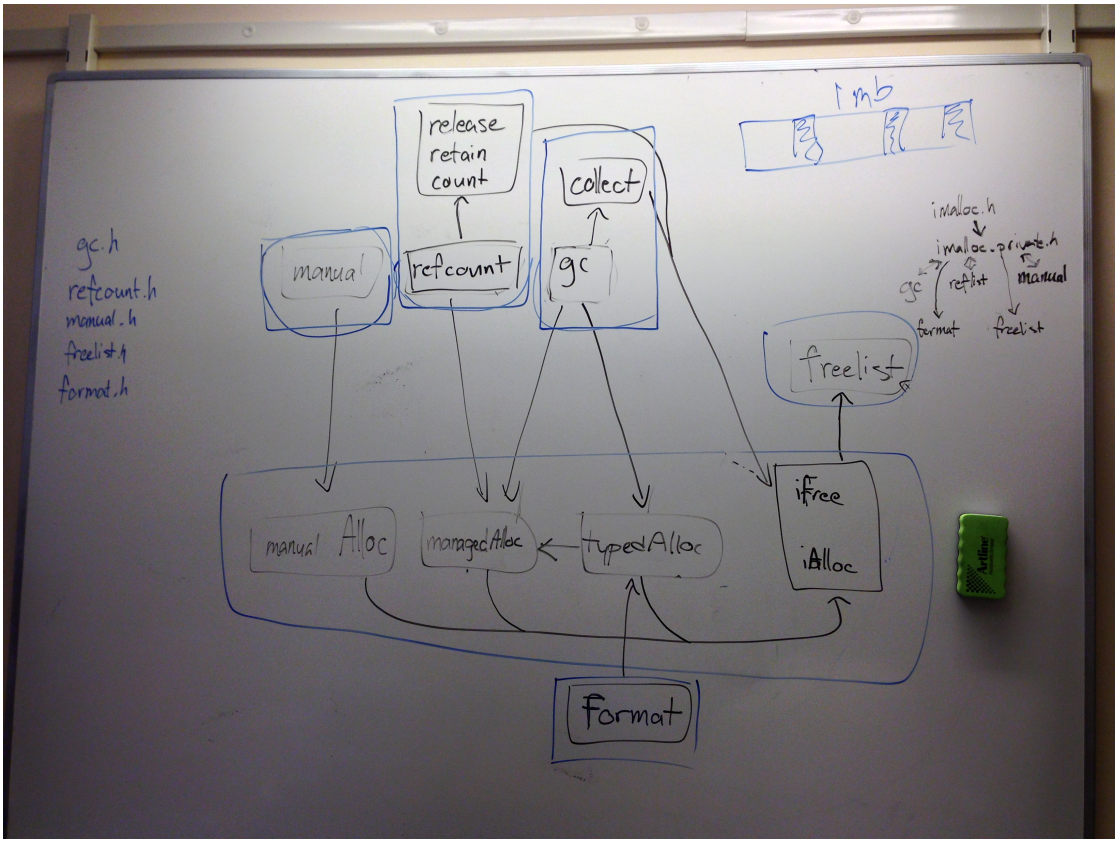
Tab size 2 med soft tabs
Exempel funktionsnamn king_in_danger
Exempel variabelnamn king_color

Vi delade slumpvis in oss i par och tilldelade paren varsin del av programmet och började sedan jobba i paren med headerfilerna.

Andreas och Daniel Den större delen av dagen gick åt till att planera hur projektet skulle vara utformat. Vi började arbeta på skräpsamlingen och test för dessa.

Elias och Jakob Gick igenom hur minneshanteringen kommer fungera på representationsnivå. Diskuterade en del med Åke och Niclas också och kommer inte fram till någon vettig lösning. Vi pratade med våran coach Niklas också och han sa att han inte har fått tillräckligt med info för att hjälpa oss med detta. Vi mailar Tobias för att se vad han har att säga.

Niclas och Åke Gjorde makefile och skelett till unittesting. Försökte lösa designproblemet med hur refcount och de privata managed- och manualobjekten ska hanteras.



Figur 1: Våran ursprungliga design och indelning.

1.3 Tisdag 23 Oktober

Andreas och Daniel Vi fortsatte arbeta på gc och diskuterat på vilket sätt den borde fungera. Koden för att traversera stacken har inte delats ut ännu, av denna anledning är det svårt att få våra testfall att fungera som de bör (Då vi ej kan traversera stacken på ett vettigt sätt utan att se vilka pekare som leder dit).

Elias och Jakob Vi satt fast och kunde inte riktigt koda någonting på grund av att vi inte visste hur vi skulle representera våran freelist, alloclist och refcount. Vi väntade på svar från Tobias men började koda på formatstring så länge.

Vi blev klara med formatstring och alla tester till denna innan samtliga i gruppen gick och pratade med Justin Pearsson. Justin berättade att vi skulle "gömma" freelist och alloclist innan stylen och även refcount skulle "gömmas" innan objektet. Vi började arbeta med refcount igen och gjorde klart det mesta förutom release-delen. Vi kunde inte riktigt arbeta mer då vi behöver färdig kod för "ifree". Därför har nu release och testkoden en del placeholders.

Niclas och Åke Fortsatte på funktioner i memory. Fixade unittests så att alla kan skriva tester. Tillämpade TDD och par-switching (highfive!). Vi gick även och pratade med Justin som tyckte att det var helt ok att göra som vi tänkt att spara refcount direkt innan varje objekt på heapen,

trots att programmeraren då förlorar en del minne.

1.4 Onsdag 24 Oktober

Andreas och Daniel Idag arbetade vi med gc.c och främst funktioner för traversering och markering av element på heapen. Vi fastnade en del på hur vi skulle göra med de element på heapen som var en pekare till andra element inom adressrymden, i början valde vi att spara allt på en lista och sedan markera dem en och en men ändrade oss sedan till att använda rekursion. Vi är snart även klara med Sweep-delen av algoritmen, dvs den delen då vi friar upp de objekt som inte har blivit markerade som använda.

Jakob och Elias Eftersom vi inte kan slutföra refcount på grund av att release inte har funktioner tillräckligt för att free:a korrekt så har vi nästan gjort färdigt priv_imalloc idag, det saknas lite tester. Efter två timmars diskuterade om hur exemplen som finns i projektspecen var tänkt att fungera så gick vi och Åke till Wrigstad och fick konstaterat för oss att det var som vi trodde, dvs fel i specen.

Niclas och Åke Vi fortsatte på funktioner och tester av memory. Vi försökte också komma på hur vi ska göra med cross-referencing som vi har på vissa ställen.

1.5 Torsdag 25 Oktober

Andreas och Daniel Idag har vi arbetat en hel del på funktionerna för att traversera heapen, Swipe algoritmen och markeringsalgoritmen. Vi har fått det mesta att fungera och börjat felsöka. Vi behöver dock funktionen för att traversera Stacken innan vi kan säga att GC är helt klar men det borde inte vara mycket arbete kvar här nu!

Jakob och Elias Har jobbat med priv_imalloc för det mesta idag. Det har gått bra. Vi är nästan färdig med implementationen av själva imalloc-funktionen. Har mest felsökning kvar. Sedan kan vi förhoppningsvis börja med priv_free (a.k.a ifree) så vi kan slutföra refcount.

Niclas och Åke Arbetat med tester och funktioner i memory. Löst problemet med cross-referencing av headerfiler. Designändringar i vissa structs gällande memory. Implementerat en privat headerfil för memory, så att vi kan testa våra privata hjälpfunktioner med enhetstestning.

1.6 Fredag 26 Oktober

Andreas och Daniel Vi har först och främst gjort klart GC, vi har implementerat funktionerna för att traversera stacken och allting kompilerar som det sig borde. Vi måste dock fixa våra testfall då dessa inte längre fungerar på grund av att vi har ändrat hur minnet beter sig. Dessutom kräver testerna (för att kunna testa vettiga saker) att iMalloc är mer eller mindre klar, detta eftersom att gc ska "städa undan" data genererad av användaren. Vi har kollat på olika program för koddokumentation och de bäst lämpade tycks vara Doxygen som vi började arbeta med då det kan generera latexkod. Vi har även hunnit göra två stycken "flow charts" för hur de olika modulerna kommunicerar med varandra.

Jakob och Elias Precis som igår arbetade vi idag med priv_imalloc, vi skrev klart alla funktioner samt skrev tester som vi tidigare inte kunnat skriva pga. de okompletta memory-funktionerna. Nu återstår bara en del småfix och felsökning.

Niclas och Åke Implementerat memory_claim och diverse underfunktioner. Tester för desamma. Memory känns nu mer eller mindre klart, finns en del småfix och felsökning att genomföra.

1.7 Måndag 29 Oktober

Samtliga Idag träffade vi vår coach Niklas för ett avstämningsmöte. Vi roterade även parkombinationerna och under den här veckan kommer följande par att arbeta:

Andreas och Niclas Dokumentation
Jakob och Åke priv_imalloc
Daniel och Elias garbage collection

Arbetsfördelningen kommer att variera en del under veckan då de flesta av våra funktioner är mer eller mindre färdiga.

Andreas och Niclas Vi började under dagen bygga upp all dokumentation som ska in. Vi har helt enkelt börjat sammanställa all tidsrapportering, alla övergripande designdokument etc och sammanställt detta till en inlämningsbar rapport.

Elias och Daniel Idag har vi jobbat med gc-delen, det blev vi som fick fortsätta med den efter rotationen. Elias fick lite tid att sätta sig in i koden och efter det släppte Jakob och Åke ut ny kod till priv_imalloc. Detta resulterade i att vi fick modifiera vår kod då priv_imalloc nu hade olika funktioner för managed och manual alloc, något den inte hade tidigare. Detta gjorde att vi fick en del problem att kompilera vår kod. Därefter löste Åke och Jakob lite problem, varpå vi kunde ändra tillbaka vår kod till den ursprungliga. Vi fick koden att fungera som det var tänkt men det saknas fortfarande tester, något vi inte har kunnat göra då vi behöver ha iMalloc helt färdigskriven.

Jakob och Åke Skrev fler tester för priv_imalloc samt spenderade mycket tid på att debugga kod här och var. Alla började bli typ klara med sina delar och vi försökte sätta ihop och testa dem tillsammans.

1.8 Tisdag 30 Oktober

Andreas och Niclas Under dagen arbetade vi en del med koddokumentation men gick sedan över till att bygga upp de dokument vi skall lämna in i Latex. Vi har lagt in dagboken, arbetat på diverse stapeldiagram för att visa tidsfördelningen och under processen även lärt oss hur man arbetar i latex!

Daniel och Elias I väntan på att Åke och Jakob ska bli färdig med priv_imalloc så har vi börjat skriva på den övergripande designdokumentationen. Vi har haft problem med att förstå vad man ska få med och hur, men vi har skrivit det så bra vi kan utifrån vår tolkning. Efter det kom vi igång med rc:s test igen och satt med det resten av dagen.

Jakob och Åke Skrev klart sista delarna av priv_imalloc inkl tester. Fortfarande massor av buggar. 32-bitars minnesadresser i traverse_heap på 64-bitarsmaskiner, varifrån kommer dem?

1.9 Onsdag 31 Oktober

Andreas och Niclas Under dagen arbetade vi med de dokument vi skall lämna in, vi har

fortsatt formatera allt i Latex, ordnat med graferna och gjort klart den övergripande designen. Vi felsökte också GC tillsammans med de andra grupperna.

Daniel och Elias Idag har vi ägnat all tid åt felsökning av GC. Vi har problem när vårt test för GC:n har skräp att samla.

Jakob och Åke Debuggade gc-delen. Skrev om stora delar av traverse_heap. Olika problem på olika platformar. BUS error på SPARC, segfaults på x86.

1.10 Torsdag 1 November

1.11 Fredag 2 November

2 Parrotationer

Den första dagen jobbade vi allihopa tillsammans. Vi gick igenom instruktionerna och diskuterade uppgiften. Vi valde att inte dela upp oss i par innan vi var säkra på hur vi ville dela upp uppgiften och hade koll på strukturen.

2.1 Vecka 1

Dag två delade in oss i par baserat på den övergripande designen vi hade gjort. Vi använde oss av en slumpgenerator för att ta fram paren samt även vilket par som skulle jobba med vad. Under första veckan jobbade vi så här:

Niclas och Åke memory
Elias och Jakob refcount
Andreas och Daniel garbage collection

Detaljerade beskrivningar av koddelarna finns under A.

2.2 Vecka 2

Vi räknade med att arbetsfördelningen skulle variera under andra veckan då vi var klara med de flesta stora delar. Däremot roterade vi paren slumpmässigt, precis som första veckan. Vi slumpade även vilka som skulle sitta kvar med samma uppgift som tidigare. Gruppindelningen vecka två såg ut så här:

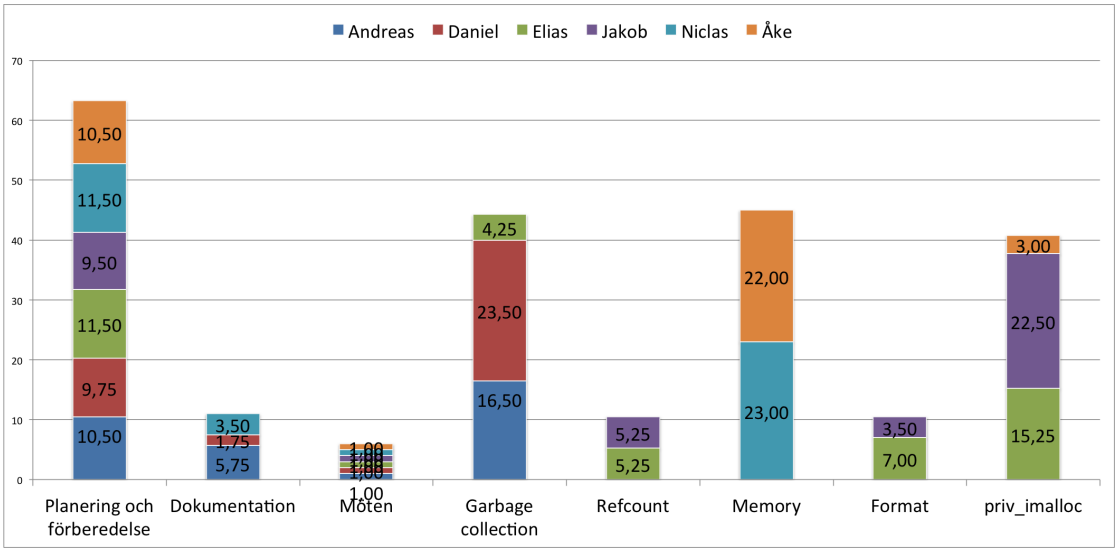
Andreas och Niclas dokumentation
Jakob och Åke priv_imalloc
Daniel och Elias garbage collection

3 Tidsåtgång

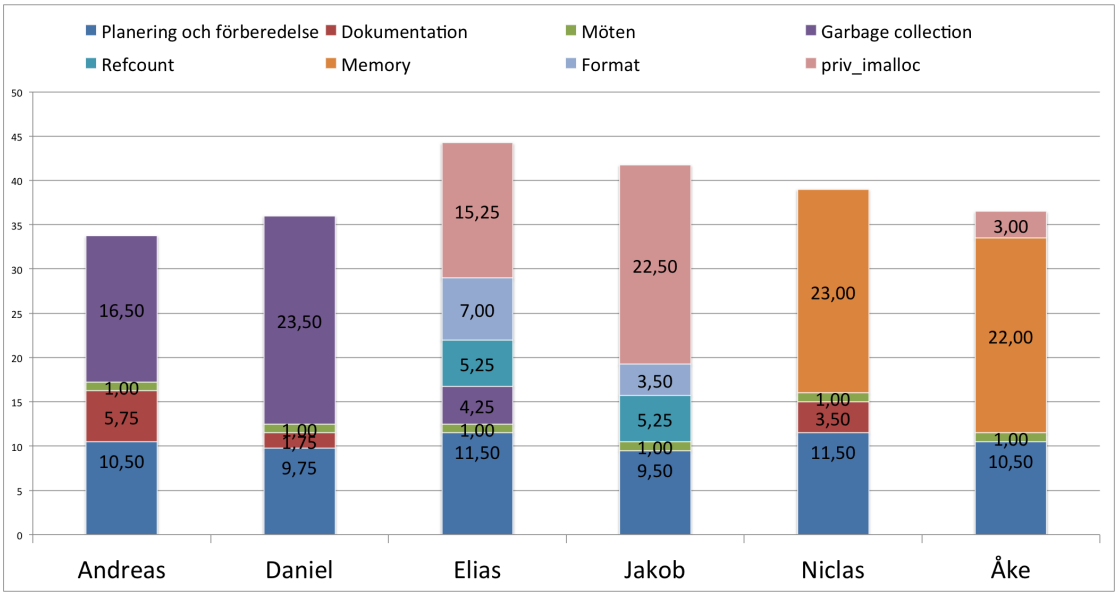
Under projektets gång har vi arbetat med ett program för "Time Tracking" som heter Tick. Efter varje arbetsdag rapporterade alla in hur många timmar dem hade arbetat samt hur många timmar dem har lagt på diverse delar av projektet.

Vi har totalt arbetat 230 timmar med projektet varav 6 timmar på möten, x timmar på dokumentation, x timmar på planering samt x timmar på implementation och testning.

Då vi har försökt arbeta enligt test driven development är det svårt att räkna ut hur mycket tid som gick specifikt till testerna utan vi ser det som en del av implementationen. Nedan följer två diagram där arbets- och tidsfördelningen är tydlig.



Figur 2: Tidsfördelningen över projektets olika delar samt vem som arbetat på vad.



Figur 3: Personlig tidsfördelning över de olika delarna av projektet.