LABORATION 5
KÖ

Rapport av
Henrik Bäck, 850611-6253
Mathias Andersson, 850424-6292

Handledare
Kerstin Anderson
Nils Dåverhög

KARLSTADS UNIVERSITET
2004-11-30

Innehållsförteckning

1. Antaganden

Under projektet antogs att köns displayfunktion skulle ”fylla på” en befintlig array med värden, denna array skulle sedan användas för att skriva ut den totala kön. Det antogs även att efter avslutad programkörning skall kön raderas, med detta alltså inte sparas till sekundärmedia.

2. Användarbeskrivning

Programmet är utvecklat för att kunna lagra element i en kö och användargränssnittet är utformat i en CLI[1]-miljö. Genom en meny kan användaren välja olika funktioner för att hantera kön. Det finns möjlighet att visa den totala kön, lägga till element i kön, titta på det första elementet i kön, ta bort det första elementet i kön, tömma kön och att avsluta programmet.

Lägga element i kö

För att lägga till ett element i kön ska man vid huvudmenyn ange menyalternativ 2.

Fig 1. *Programmets huvudmeny*

Ange därefter det HELTAL (i denna version) som skall läggas i kö. Tryck därefter RETUR. Ett meddelande visas på skärmen om köläggning lyckats, annars visas ett felmeddelande.

Efter att köläggning lyckats kommer programmet att återvända till huvudmenyn. Se fig 1.

[1] Command Line Interface

Fig 2. *Lägg element i kö*

Visa kö

Från huvudmenyn välj menyalternativ 1 för att visa aktuell kö. Den aktuella kön skrivs ut på skärmen och kan läsas av. Det första elementet, elemetet på tur, står högst upp i kön. Ifall kön är tom kommer ett meddelande om detta att skrivas ut på skärmen. Ingen kö kan visas!

Fig 3. *Exempel på en kö, 1 står på tur i kön.*

Visa första elemetet i kön

Genom menyalternativ 3 i huvudmenyn (Se Fig 1.) Om inte kön är tom så kommer det första elementet i kön att skrivas ut annars kommer ett meddelande om att kön är tom att visas på skärmen.

Ta bort första elementet

För att ta bort första positionen i kön skall menyalternativ 4 vara i huvudmenyn användas. Genom att ange detta alternativ kommer det första elementet i kön att tas bort, dock om inget element finns i kön kommer ett meddelande att skrivas till skärmen med information om att kön är tom.

Köns status

Menyalternativ 5 i huvudmenyn ger användaren möjlighet att se hur många element som står i kö. Finns inga element i kön kommer en text att skrivas till skärmen med information om att kön är tom.

Töm kön

Genom att ange menyalternativ 6 i huvudmenyn kommer kön att tömmas på sitt innehåll. Detta innebär att hela kön kommer att raderas och de imatade värdarna kommer att bli oåtkomliga!

Avsluta

Detta menyalternativ, 0, gör det möjligt att avsluta programmet. OBSERVERA! Vid avslutning kommer hela kön att raderas och inte bli återställbar.

3. Systemöversikt

Programvaran består av tre stycken klasser. Vid programmering behövs endast klassen Queue användas. Klassen kö sköter alla kommunikation med den ”fysiska” kön som finns lagrad med hjälp av noder och klassen List.

För övrigt består programmet utav en stor mängd drivrutiner som sköter all kommunikation mellan användaren och systemnivå. Drivrutinerna läser in användarens val och utför, om möjligt, detta hos kön.

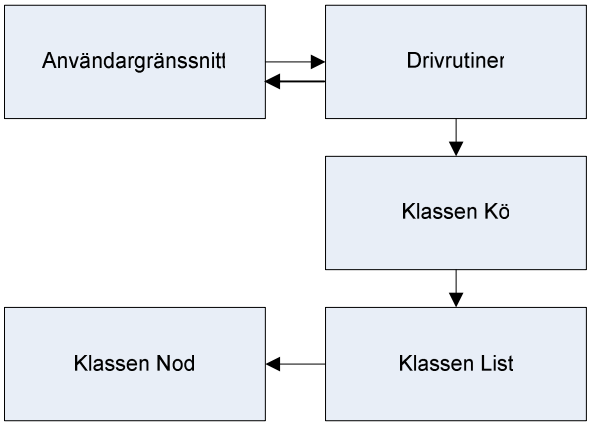

Fig 4. *Systemöversikt*

Filrelationer

Filen *lab5.cpp* har hand om själva main filen för programmet. Härifrån startas och anropas programmets funktioner. Filen inkluderar *drivers.h* och *queue.h*

drivers.cpp innehåller köns drivfunktioner. Även denna fil inkluderar *drivers.h* eftersom *lab5.cpp* och *drivers.cpp* separatkompileras.
I och med att *drivers.h* inkluderas

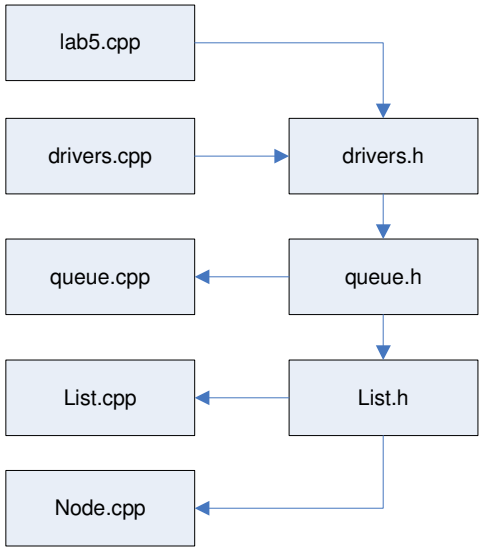

Fig 5. *Filrelationer*

kommer även resterande .cpp och .h-filer att inkluderas. Se figur till höger.

queue.h innehåller klassen Queue. Denna fil inkluderar *queue.cpp* som innehåller kö-klassens funktioner. *queue.h* inkluderar även *List.h* som innehåller klassen List.

List.h inkluderar *List.cpp* som innehåller List-klassens funktioner. *List.h* inkluderar även *Node.cpp* eftersom klassen List använder sig av klassen Node.

4. Detaljerade modulbeskrivningar

Klassen Queue

Klassen Queue ger tillgång till att skapa ett objekt vars egenskaper är som hos en kö. Klassen Queue har följande funktioner/operationer:

- enqueue
- dequeue
- isEmpty
- getSize
- display

klassen har också en konstructor och en destructor som körs när man skapar en instans av ett objekt respektive när objektet går ur scope[2]. Dock har inte klassen Queue någon specifik destructor då klassen endast är ett skal för att hantera en sorts lista med specifika egenskaper.

Villkor och beskrivningar för klassen Queue finns i Bilaga A sida 11.

Klassen List

Klassen List ger används av klassen Queue.
Klassen List har följande funktioner/operationer:

- insert
- remove
- isEmpty
- isElement
- getPos
- size
- getElement

klassen har också en konstructor och en destructor som sköter minnesallokering.

[2] Objektet raderas från minnet och är inte längre användbart.

Villkor och beskrivningar för klassen List finns i Bilaga A sida 5 och 6.

Klassen Node

Klassen Node är den klass som innehåller den data som önskas lagras. Klassen Node har inga operationer.

Drivrutiner

Programmet har följande drivrutiner för att ta hand om programmets kö.

- mataIn
- laggTill
- visaKoe
- visaFirst
- tabortFirst
- toem
- status

All information om och hur dessa drivrutiner fungerar och kan användas finns i Bilaga A sida 1.

5. Sammanfattning

Projektet har förståelse i hur en kö fungerar och hur denna kan implementeras med hjälp av en lista, i detta fall en länkad lista. Tidsåtgången för projektet har varit god och allt som skall ha blivit gjort har blivit gjort inom avsatt tid. Projektet utföll väl med uppsatta förväntningar och i planeringen låg att få projektet klart så fort som möjligt på grund av andra kurser och projekt som skall slutföras inom samma tidsrymd.

6. Problem

De största problem som uppstått har under projektets fortlöpande har varit relaterade till slarv eller trötthet. Några större implementationsproblem har inte förekommit under projekttiden.

Slarvet eller trötttheten har resulterat i konstiga kompileringsfel, dock har alltid dessa kunnat lösas inom en kort stund då hjärnan har fått återhämta sig en stund.

7. Referenslista

Inga övriga källor vid sidan av kurslitteratur har använts.

Bilagor

Bilaga A – ADT Queue
Bilaga B – Programkod