Labbrapport: Labb 4

Författare: Dennis Löv och Fredrik Jansson

Datum: 2025-12-20

Kursnamn: GIK299 – Objektorienterad Programmering

Examinator: Elin Ekman och Ulrika Artursson Wissa

Innehåll

1. Introduktion

I den här laborationen har vår uppgift varit att skapa ett program som hanterar data om personer enligt SCRUM-metoden, vilket innebär att arbetet delas upp i flera “sprintar”. Syftet med denna övning är dels för att träna vår kunskap och förståelse av C# och koncept som klasser samt olika datatyper, men syftet är även att lära oss om agila arbetsmetoder och hur de kan fungera.

2. Metod

2.1. Verktyg

Visual Studio 2022, version 17.14.19 (July 2025) med ramverket .NET 8.0 har använts för att bygga och testa vår applikation.

Discord användes för planering av vårt arbete samt kommunikation oss emellan, och skärmdelningen användes för parprogrammeringen när vi inte kunde vara fysiskt närvarande.

2.2. Stegvis beskrivning av tillvägagångssätt

Vi kände inte att någon flödesschema behövdes då vi båda programmerat förut och det är en ganska enkel uppgift med tydliga instruktioner.

Dennis skötte programmeringen på sin skärm och skärmdelade åt Fredrik över discord. Vi följde kravlistan punkt för punkt när vi programmerade.

När det var dags för Fredrik att genomföra sprint 2 uppstod problem då hans dator plötsligt vägrade att installera .NET. Han kunde då inte köra koden, och en lång felsökning gjordes där han installerade om både visual studio och försökte installera om .NET sdk men fick bara felmeddelandet att installationen misslyckades. När vi kollade i loggarna efter en felkod och sökte på nätet verkar ingen ha samma problem som Fredrik, så vi testade att installera JetBrains Rider som alternativ till visual studio men det fungerade fortfarande inte då det var .Net som var problemet. Till slut kunde Fredrik använda en annan dator, och då fungerade allting arbetet flöt på.

2.3. Förutsättningar för att göra labben

Vi behövde ju såklart använda C# och .NET 8.0 samt att vi använde Visual studio 2022. Vi var båda hemma när arbetet gjordes så vi använde discord för att kunna prata med varandra men även skärmdela för parprogrammeringen.

2.4. Testning av koden

Vi testade koden genom att försöka ge vanliga felaktiga inputs och se hur programmet svarade. t.ex. bara klicka enter i menyn, att ange siffror när programmet frågade efter en färg och siffror istället för ett kön osv.

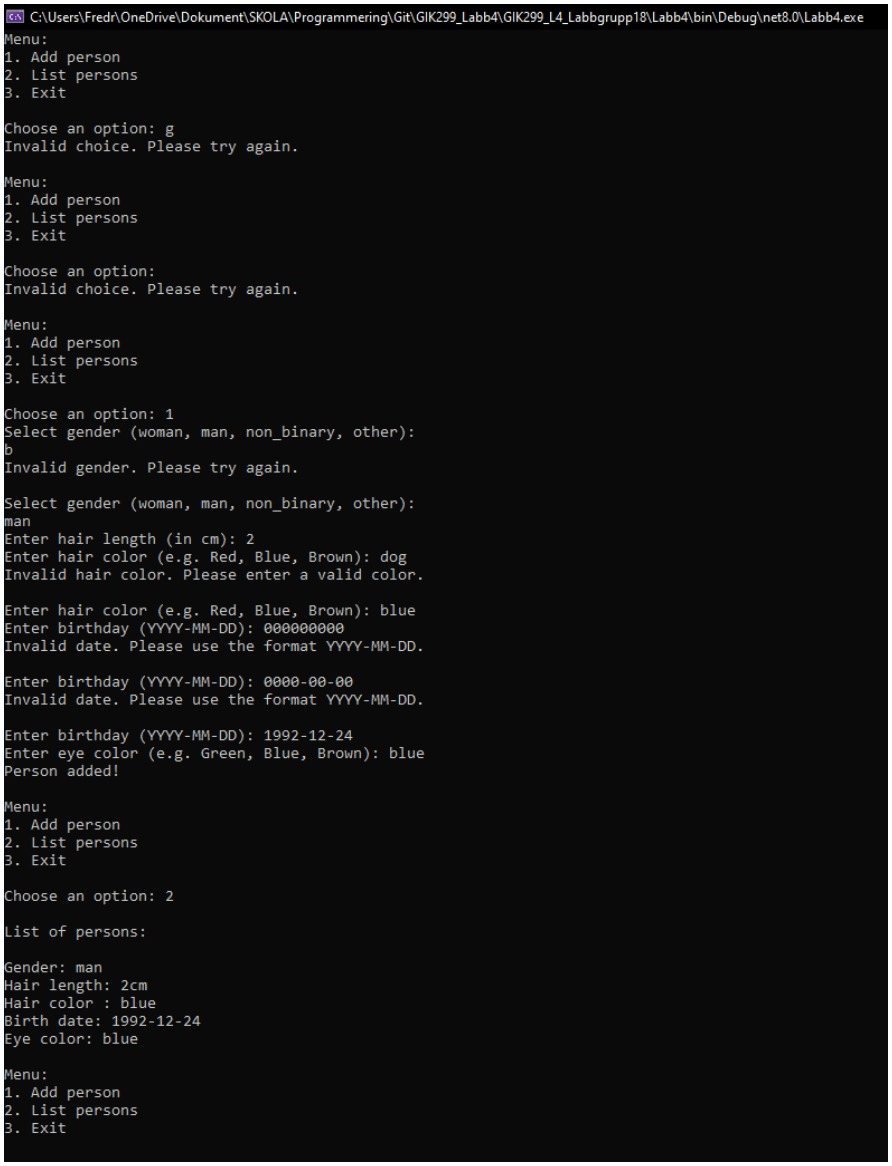
2.5. Etiska överväganden

Då vi inte behandlat känslig eller riktig information så finns inga särskilda etiska överväganden att ta hänsyn till. Det enda övervägandet är väl möjligtvis användningen av AI för att lösa uppgiften, men i syfte av att lära oss så mycket som möjligt så har vi valt att använda AI så lite som möjligt, samt att inte låta AI sköta tänkandet åt oss utan att det faktiskt är vi som kommer på lösningarna. AI kan dock användas som en rådgivare samt kontrollant när det behövs.

3. Resultat

Att implementera funktionaliteten enligt kraven var inte alltför svårt då vi båda har tidigare erfarenhet inom C#, även om vi dock behövde fräscha upp kunskaperna kring vissa delar som t.ex. exakt hur syntaxen ska se ut. Att få ner funktionaliteten enligt kraven gick ganska fort, men det tog en del tid att testa olika utfall och verkligen säkerställa att allting fungerade som det skulle. Vi kom fram till att en kombination av TryParse och null-coalescing (??()) var mycket robust i att hantera de felaktiga inmatningarna som vi kunde komma på. Vi la även ner en stor del tid på att alla utskrifter skulle se snygga och fina ut, samt att all kod var väl kommenterad.

Ett exempel var att listan av personer returnerade färger så här “Color [Red]” om man t.ex. valt röd, när vi istället ville att utskriften bara skulle visa “Red”. Detta löste vi genom att hämta färgens namn från färgvariabeln istället för att bara skriva ut själva variabeln i listan.



På bilden kan vi se några exempel av inmatningar vi gjorde för att testa felaktiga inputs, samt utskriften av en person i en lista.

4. Diskussion och reflektion

Då vi har en tidigare erfarenhet av C# så var det inte jättemycket nytt vi fick använda här, men det var samtidigt en väldigt bra uppgift för att fräscha upp sina kunskaper då det var ganska längesen vi använde oss av C# och då vi har arbetat med flera olika språk sen dess. Det var även väldigt intressant att lära oss om agila arbetssätt och att testa att dela upp arbetet på detta vis, samt att använda sig av parprogrammering vilket var nytt för oss.

4.1. Diskussion kring resultat

Vi fick inte så mycket oväntade resultat, det var väl isåfall mest fula utskrifter isåfall, som vi nämnde i resultatdelen, där förklarar vi även hur vi hanterade dessa.

4.2. Reflektion kring sprint 1

Vi lärde oss hur man skapar en struct-instans utan att spara den i en variabel (lite annorlunda från C++). Parprogrammering var något vi inte jobbat med tidigare så att jobba på det sättet var nytt och intressant.

4.3. Reflektion kring sprint 2

Det svåraste med sprint 2 var att faktiskt kunna sätta sig och arbeta. Först blev Fredrik fast i en stuga i Hälsingland under en röd stormvarning vilket gjorde att tidsplanen för vårt arbete sprack och vi kom igång mycket senare än tänkt. Sen var det de tekniska problemen kring Fredriks .NET som inte ville fungera vilket även det försenade arbetet avsevärt (Fredrik har lagt flera timmar mer på felsökningen av detta problem än på faktiskt arbete av labben).

4.4. Reflektion kring alternativa lösningar

Vi tycker att koden känns robust och lättförståelig, så vi hade nog inte gjort på något annat sätt.

Frågor till AI-verktyg

Verktyg: Copilot

Fråga/prompt: Förklara varningarna som uppstår på ett simpelt sätt och vilka konsekvenser de kan ha.

På vilket sätt svaret användes: Programmet gav ett flertal varningar vid körning som vi inte riktigt förstod. Programmet gick fortfarande att köra och fungerade väl, så vi bad copilot förklara vad felet var i simplare termer och hur dessa kan påverka programmet för att då kunna förebygga eventuella problem. Det visade sig vara null hanteringen som saknades, vilket vi åtgärdade med hjälp av null-coalescing-operatörer (?? ()) för att fånga null-inmatningar i våra console.ReadLine() rader.

—----------

Fråga/prompt: Uppfyller programmet följande krav: (kravlistan från canvas)

På vilket sätt svaret användes: Efter att ha kontrollerat koden och programmet själva så lät vi copilot dubbelkolla så vi inte missat något specifikt i kravlistan.

Parprogrammeringslogg

Labb

Datum	Tid i timmar	Ensam eller i par	% fördelat i att sitta vid tangentbordet
20/10-2025	0:20	par	Dennis: 100% Fredrik: 0%
05/01-2026	0:45	par	Dennis: 0% Fredrik: 100%