Labbrapport: Labb 4

Författare: Felix Wiklund och Rezzan Bayar

Datum: 2026-01-09

Kursnamn: GIK299 – Objektorienterad Programmering

Examinator: Elin Ekman och Ulrika Artursson Wissa

Innehåll

1. Introduktion

I den här labben har vi genomfört programmeringsuppgiften för Labb 4. Uppgiften gick ut på att skapa ett konsolprogram där användaren kan mata in information om personer. Programmet samlar in uppgifter som förnamn, efternamn, kön, hårfärg, ögonfärg och födelsedatum. När användaren har registrerat fler än en person kan programmet skriva ut en lista med all information om personerna. Programmet avlutas när användaren väljer det. Syftet med labbrapporten är att beskriva hur vi arbetade med labben, vilka metoder vi använda och vilka erfarenheter vi har fått under arbetets gång.

2. Metod

2.1. Verktyg

- Rider
- Tidigare lektionslabbar som t.ex. Bibliotekslabb
- Föreläsningar och kursmaterial
- Kodexempel från instruktionerna

2.2. Stegvis beskrivning av tillvägagångssätt

För att lösa labbuppgiften började vi med att läsa instruktionerna noggrant. Vi delade upp arbetet enligt SCRUM-metoden. Under sprint 1 var Felix “Driver” och Rezzan “Navigator”, och under sprint 2 bytte vi roller.

Vi arbetade stegvis, började med sprint 1 och skapade de egendefinerade typerna (enum, struct och class). Vi testade koden för att säkerställa att allt fungerade som det skulle innan vi gick vidare. Sedan byggde vi med sprint 2, där vi implementerade listor, meny, felhantering och möjligen att lägga till flera personer.

2.3. Förutsättningar för att göra labben

Alla nödvändiga program och verktyg var redan installerande, och vi hade tillgång till kursmaterial och tidigare labbar som gjordes innan.

2.4. Testning av koden

Vi testade koden ofta, steg för steg, för att kunna identifiera problem tidigt. Varje funktion testades i konsolen innan vi gick vidare till nästa ({}).

2.5. Etiska överväganden

Vi hanterade inga riktiga personuppgifter eller känslig information i laborationen.

2.6. Flödesschema

Programmet har ett tydligt flöde som börjar med att visa huvudmenyn där användaren kan välja mellan:

1. Lägg till person
2. Lista personer
3. Avsluta

Vid val av ”Lägg till person” frågar programmet stegvis efter:

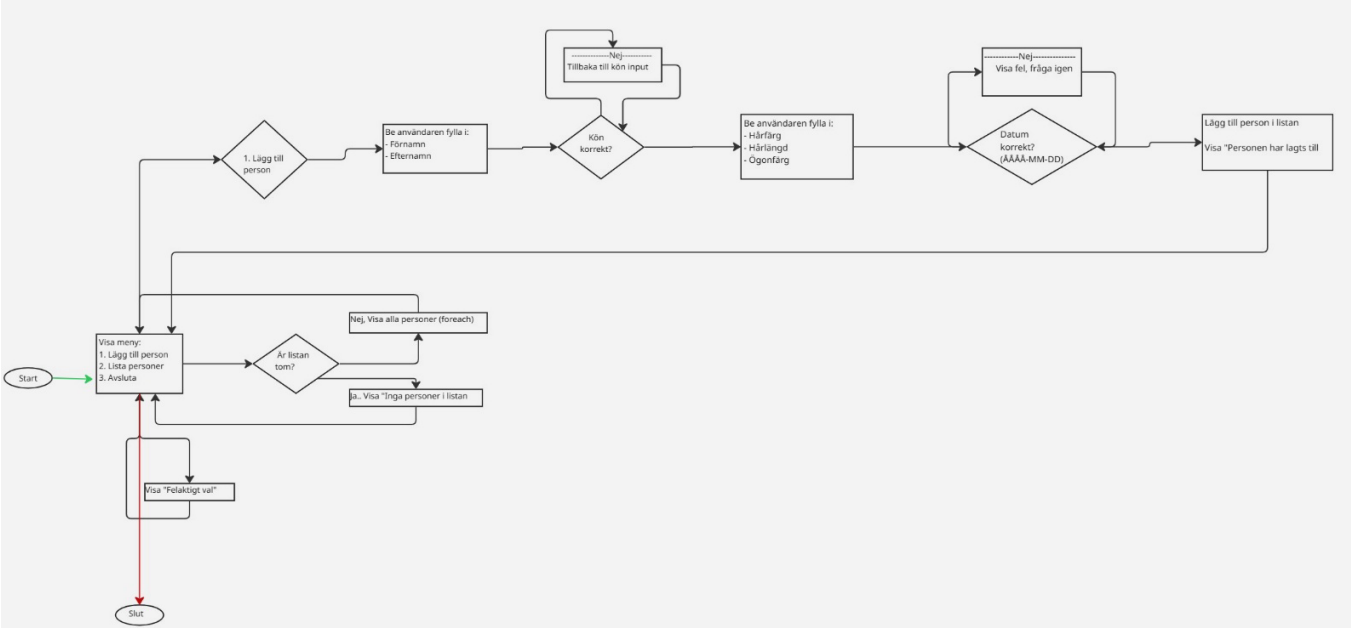
- Förnamn och efternamn
- Kön (kontrolleras mot giltiga alternativ)
- Hårfärg och hårlängd
- Ögonfärg
- Födelsedatum (valideras för korrekt format)

Om användaren matar in felaktiga värden (t.ex fel kön eller fel datum) skickas hen tillbaka till frågan tills korrekt värde anges. När all information är korrekt läggs personen till i listan.

Vid val av ”Lista personer” skrivs alla registrerade personer ut i konsolen.

Vid val av ”Avsluta” stängs programmet.

Flödet kan beskrivas som en loop som fortsätter tills användaren väljer att avsluta.



Figur 1: Flödesschema över Person-programmets huvudflöde.

3. Resultat

Programmet blev ett fungerande konsolprogram som uppfyller både sprint 1 och sprint 2. När programmet startas visas en meny med tre alternativ:

1. Lägg till person
2. Lista personer
3. Avsluta

När användaren väljer att lägga till en person får man skriva in all information steg för steg. Programmet validerar att data är korrekt, till exempel att födelsedatum är i rätt format och att kön är ett av de val som finns. Om användaren skriver fel data får man möjlighet att föröka igen.

När användaren väljer att lista personer skrivs all information ut med hjälp av ToString()-metoden i Person-klassen. Varje persons information visas på ett tydligt och strukturerat sätt.

Under arbetet stötte vi på ett mindre problem där listan inte visades korrekt på första försöket. Felet berodde på felplacerad måsvingar i koden, med det löstes snabbt och fungerade sedan som det skulle.

Figur 2: Skärmbild av programmet när personer har lagts till.

4. Diskussion och reflektion

Labben gick bra då vi känner att labbens olika delar har vi sätt tidigare i kursen. Allt som vi har arbetet med innan labb 4 har underlättat arbetet med labb 4.

4.1. Diskussion kring resultat

Genom att testa programmet ofta kunde vi snabbt identifiera problem och lösa dem. Vi märkte att små fel, som felplacerade måsvingar, kan påverka hela programmet. Att arbeta stegvis och testa varje del först gjorde det lättare att se vad som fungerade och vad som behövde ändras.

Även när vi bad programmet att skriva ut listan i konsolen så hoppade den över listan och gav oss i stället vårt felmeddelande trots att vi lagt in flera personer och uppgifter om personerna i programmet. Misstaget var lätt så det löste sig fort men vi har lärt oss nu att ibland är det lätt missa väldigt små saker som kan göra en stor betydelse för resultatet.

4.2. Reflektion kring sprint 1

I sprint 1 fick vi arbeta med egendefinierade datatyper som enum, struct och class. Vi lärde oss hur enum kan begränsa vilka värden som är möjliga, till exempel för kön. Struct gjorde det möjligt att samla relaterad information, som hårfärg och hårlängd, på ett tydligt sätt. Vi fick också bättre förståelse för properties och hur ToString() kan användas för att visa information på ett snyggt sätt. Sprint 1 tog längre tid än vad vi trodde, vilket visade oss att det är viktigt att planera och ta det steg för steg.

4.3. Reflektion kring sprint 2

I sprint 2 byggde vi vidare med menyer, listor och felhantering. Vi lärde oss hur listor kan användas för att lagra flera objekt och hur while-loopar och switch-satser styr flödet i programmet. Felhantering med try-catch och TryParse gjorde programmet mer stabilt och användarvänligt. Att arbeta iterativt och testa ofta hjälpte oss att hitta och åtgärda fel snabbare. Vi upplevde att bytet av roller mellan sprintarna var bra, eftersom båda fick en helhetsförståelse av programmet.

5. Frågor till AI-verktyg

Verktyg: Gemini, ChatGPT

Fråga/prompt 1: ”Varför får man inte ut listan i konsolen efter att man lagt in uppgifter om fler än en person i programmet?”

- **På vilket sätt svaret användes:** Vi kunde först inte förstå varför programmet inte gav oss listan efter att vi lagt in flera personer i programmet. Vi fick ut ”Listan är tom” i konsolen. Vi förstod sen att programmet hoppade över listan och i stället gav oss ”Listan är tom” som om vi inte hade

lagt till personer i programmet. Lösningen på problemet var lätt ”måsvingarna {}” var fel på en del i koden.

Fråga/prompt 2:”Hur kan man få en hel tom sida när man trycker på menyvalen?”

- **På vilket sätt svaret användes:** ChatGPT förklarade hur Console.Clear() fungerar för att rensa skärmen mellan menyval. Vilket vi använde för att förbättra användarupplevelsen och göra programmet mer överskådligt.

Fråga/prompt 3:”Kan du hjälpa oss att förstå hur flödesschemat för Person-programmet kan se ut?”

- **På vilket sätt svaret användes:** Vi använde ChatGPT för att få stöd med att skapa ett tydligt och strukturerat flödesschema över programmet. AI gav oss förslag på hur flödet kunde beskrivas och vilka steg som bör ingå, vilket hjälpte oss att visualisera menyvalen, inmatningen av personuppgifter och valideringarna. Därefter skapade vi själva flödesschemat i Miro utifrån förslaget.

6. Parprogrammeringslogg

- **Datum:** 2025-12-19
- **Deltagare:** Rezzan Bayar och Felix Wiklund
- **Arbete:** Vi började med att skapa de grundläggande egendefinerande typerna, alltså enum för kön, struct för hår och Person-klassen. Vi testade även ToSting-metoden för att se att informationen skrivs ut korrekt.
- **Reflektion:** Samarbetet fungerade bra. Vi lärde oss hur enum och struct fungerar och hur man kan använda dem i en klass.

- **Datum:** 2025-12-20
- **Deltagare:** Rezzan Bayar och Felix Wiklund
- **Arbete:** Vi implementerade Addperson-metoden och gjorde menyvalen för att lägga till personer. Vi testade felhantering för felaktig kön och felaktigt datum.
- **Reflektion:** Lärde oss hur man kan använda try-catch och validering för att göra programmet mer stabilt.

- **Datum:** 2025-12-21
- **Deltagare:** Rezzan Bayar och Felix Wiklund
- **Arbete:** Vi byggde ListPersons-metoden och testade att skriva ut flera personer. Vi felsökte problem där listan inte visades korrekt på första försöket.
- **Reflektion:** Vi insåg hur viktigt det är att testa ofta steg för steg. Att byta roller mellan Driver och Navigator gav båda bättre helhetsförståelse.

Datum	Tid i timmar	Ensam eller i par	% fördelat i att sitta vid tangentbordet
19/12	4	Par	Felix: 100%
20/12	5	Par	Rezzan: 100%
21/12	3	Par	Felix: 50% Rezzan: 50%