

Labbrapport: Labb 4

Författare: Shaho Sharifzadah och Theodor Herrdin

Datum: 2026-01-02

Kursnamn: GIK299 – Objektorienterad Programmering

Examinator: Elin Ekman och Ulrika Artursson Wissa

Innehåll

1. Introduktion	3
2. Metod.....	4
2.1. Verktyg.....	4
2.2. Stegvis beskrivning av tillvägagångssätt	4
2.3. Förutsättningar för att göra labben	4
2.4. Testning av koden	4
2.5. Etiska överväganden.....	5
3. Resultat	6
4. Diskussion och reflektion	7
4.1. Diskussion kring resultat	7
4.2. Reflektion kring sprint 1	7
4.3. Reflektion kring sprint 2	7
4.4. Reflektion kring alternativa lösningar	7
Frågor till AI-verktyg.....	9
Referenser	10

1. Introduktion

Denna laboration har som mål att skapa ett konsolbaserat program i programmeringsspråket C#. Programmet ska hantera information om personer genom att använda egendefinierade datatyper. Fokus ligger på att använda enum, struct och class samt att bygga ett menybaserat program där användaren kan interagera med system.

Laborationen är uppdelad i två sprintar. Den första sprinten fokuserar på datamodellering och objektorienterad struktur. Den andra sprinten fokuserar på programflöde, användarinmatning och felhantering. Arbetet syftar till att ge en grundläggande förståelse för hur ett större program kan struktureras på ett tydligt och stabilt sätt.

2. Metod

2.1. Verktyg

Följande verktyg används vid genomförandet av laborationen:

- Programmeringsspråket C#
- Visual Studio
- .NET Console Application

Visual Studio användes för att skriva, testa och köra programmet. Konsolen användes för in- och utmatning av data

2.2. Stegvis beskrivning av tillvägagångssätt

Arbetet genomfördes stegvis enligt instruktionerna för respektive sprint.

I sprint 1 skapades först enum för kön samt en struct för hår, därefter skapades klassen Person som innehåller information om en person, såsom förnamn, efternamn, kön, födelsedatum, ögonfärg och hår. Klassen innehåller properties för alla värden samt en konstruktor för att skapa nya objekt. Metoden ToString() överskreds för att kunna skriva ut personens information på ett tydligt sätt.

I Sprint 2 skapades en lista av typen List<Person> för att lagra alla skapade personer. Programmet byggdes upp kring en while-loop som styrs av en bool-variable. En meny skrivs ut i konsolen där användaren kan välja att lägga till en ny person, visa alla personer eller avsluta programmet. Användarens val hanteras med en switch-sats.

För att undvika fel används validering av inmatning med TryParse, loopar och try-catch. Användaren får möjlighet att försöka igen vid felaktig inmatning.

2.3. Förutsättningar för att göra labben

För att kunna genomföra laborationen krävs grundläggande kunskaper i C# och objektorienterad programmering. Det krävs även förståelse för variabler, loopar, villkor, listor och metoder. En fungerande utvecklingsmiljö med stöd för .NET är nödvändig för att kunna köra programmet.

2.4. Testning av koden

Testning av programmet kördes hela tiden under kodandet. Olika typer av koder testades för att säkerställa att programmet hanterar både giltiga och ogiltiga värden. Menyval testade för att kontrollera att rätt funktion anropas. Programmet testades även för att säkerställa att det inte kraschar vid felaktig kod och att användaren alltid får möjlighet att försöka igen.

2.5. Etiska överväganden

Vi använde oss inte av något som skulle vara känslig data, all info är påhittad.

3. Resultat

Resultatet av laborationen är ett fungerande konsolprogram som uppfyller samtliga krav, programmet kan skapas och lagra flera personer i en lista samt skriva ut alla personer i konsolen. Alla menyval fungerar korrekt och programmet avslutas endast när användaren väljer att göra detta.

Testningen visade att programmet hanterar felaktig inmatning på ett stabilt sätt. Programmet fortsätter att fungera även när användaren matar in ogiltiga värden.

4. Diskussion och reflektion

Fokus ligger på hur programmet byggdes upp, vilka val som gjordes och vad som kunde ha gjorts annorlunda. Avsnittet är uppdelat i flera delar för att tydligt behandla olika aspekter av arbetet.

4.1. Diskussion kring resultat

Resultatet visar att programmet uppfyller samtliga krav i laborationen. Programmet kan hantera flera personer, lagra dom i en lista och skriva ut korrekt information till användaren. Alla menyval fungerar som förväntat och programmet avslutas på ett kontrollerat sätt.

4.2. Reflektion kring sprint 1

Sprint 1 fokuserade på att skapa programmets grundstruktur genom att använda egendefinierade datatyper. Arbetet med enum struct och class gav en tydlig förståelse för hur data kan organiseras på ett strukturerat sätt. Genom att definiera kön som enum och hår som struct blev informationen tydlig och lätt att använda i klassen Person.

Att skapa klassen Person med properties och konstruktor gjorde det enklare att hantera personinformation på ett kontrollerat sätt. Metoden ToString() bidrog till att utskriften av objektet blev tydlig och lätt att förstå. Sprint 1 skapade en stabil grund som var nödvändig för att kunna arbeta i sprint 2

4.3. Reflektion kring sprint 2

Sprint 2 byggde vidare på strukturen från sprint 1 och fokuserade på programflöde och användareinteraktion. Arbetet med while-loop, meny och switch-sats gav en bättre förståelse för hur ett program kan styras av användarens val. Användningen av en bool-variabel för att kontrollera programmet körnings gjordes flödet tydligt och lätt att följa

Felhantering och validering av inmatning var en viktig del av sprint 2. Genom att använda TryParse, loopar och try-catch kunde programmet göras mer stabilt. Sprint 2 visade hur viktigt det är att kombinera struktur med användarvänlighet för att skapa ett fungerande program.

4.4. Reflektion kring alternativa lösningar

Det finns flera alternativa lösningar som hade kunnat användas i programmet. En lösning hade varit att separera användarinmatningen från logiken genom att placera mer ansvar i Program-klassen i stället för i Person-klassen. Det hade kunnat göra koden ännu mer uppdelad och tydligare.

Ett annat alternativ hade varit att lagra personerna i en fil eller databas i stället för i en lista som endast finns under programmets körning. Detta hade gjort det möjligt att spara information mellan körningar. Dessa lösningar valdes bort för att hålla programmet på en nivå som är godkänt i laborationen.

Frågor till AI-verktyg

Verktyg: ChatGPT PRO och Google Gemini PRO

Fråga/prompt: Uppladdning av föreläsningarna och bett om en djupare förklaring.

På vilket sätt svaret användes: AI har använts som stöd under arbetet med laborationen samt kodandet i Visual Studio. AI användes för att få förklaringar av programmeringskoncept, strukturera text till labbrapporten samt formulera akademiskt språk. AI användes inte för att generera färdig kod utan fungerade som ett stöd för förståelse och dokumentation.

All kod är skriven av Shaho Sharifzadeh och Theodor Herrdin. Detta med stöd av kursmaterial och undervisning. Användning av AI har bidragit till tydligare formuleringar och bättre struktur.

Referenser

Högskolan Dalarna. (2024). *Föreläsningsmaterial i objektorienterad programmering* [PowerPoint-presentation].

Canvas <https://canvas.du.se/>