# Inlämningsuppgift 3

**Syfte och utförande**

Med dessa inledande inlämningsuppgifter ska du befästa dina kunskaper i grunderna i programmerings-språket såväl som att du får förståelse och erfarenhet i att tolka och skriva enkla algoritmer. I denna uppgift ska du även befästa och visa upp dina kunskaper i att skapa grafiska gränssnitt och klasser.

Du ska redovisa uppgifterna individuellt, men du får naturligtvis samarbeta med andra och söka information på nätet när du löser uppgiften. Uppgiften ligger till viss del som en del av examinationen, men är även viktig för slutexaminationen och dina framsteg som programmerare.

Tänk på arbetsordningen när ni tar er an uppgifterna.

- Analys - Läs uppgiften. Sök vad som är givet och sökt.
- Algoritmformulering - Försök att beskriva lösningen på problemet så exakt det går med diagram, text eller liknande.
- Kodning - Översätt algoritmen från föregående punkt till en algoritm beskriven i ett specifikt programmeringsspråk.
- Testa och dokumentera - Provkör ditt program och kontrollera så att det fungerar. Du kan förutsätta att all indata är korrekt, så du behöver ej kontrollera dessa.

När du skriver din källkod, algoritm i programspråket, så inled den med dokumentation i stil med Bilaga 1 - Header. Du kan ändra den efter behov. Kontrollera sedan din uppgift med det lilla ”besiktningsprotokollet”, som återfinns i bilaga 2. Tänk speciellt på att ha vettiga namn på variabler, exempelvis antal istället för x. Om du har en variabel som är skapad av två eller flera ord, börja med liten bokstav för att sedan ha stora vid nytt ord. Exempel: antalkatthund blir antalKattHund. Klassnamn ska inledas med en versal, exempelvis Konto.

**Uppgifter**

**1. Bankomaten**

Du ska i denna deluppgift skapa ett program som simulerar en bankomat. Du ska skriva programmet med en klass för konto i grunden. Det ska vara möjligt att via en meny sätta in och ta ut pengar. Det ska även finnas ett val att göra en utskrift av aktuellt saldo.

Betygskriterier:

E: Programmet bygger på en klass Konto och hanterar dialog och menyer via ett textbaserat (terminalen) gränssnitt. Samtliga uppgifter ska uppfylla kraven i lydelsen. För betyget E ska du använda bra identifierarnamn och konsekvent språk. Det ska vidare vara informativa utskrifter och en enkel inmatning där det är tillämpbart. Programmet ska ha en dokumenterande inledning.

C: Programmet har ett grafiskt gränssnitt. Programmet har klassen Konto som bas, men nyttjar arv för att skapa olika kontotyper, ex bankkonto, kreditkonto. Du implementerar metoderna __str__ och __getitem__.

A: Programmet har stöd för flera konton/användare (någon form av identifikation krävs). Programmet ska även kunna hantera att visa de senaste tio transaktionerna för varje konto. Programmet ska lagra kontoinformation i en textfil, som läses in vid programstart och uppdateras vid avslut. Hantering av data på fil kan även hanteras via menyval.

**Redovisning**

Källkodsfilerna laddas upp i Showbie (www.showbie.com) med kod YFNDD. Föreslagen deadline är 19/12.

**Bilagor**

Detta är ett förslag på header i dina källkodsfiler. Du kan modifiera den så den passar dig och dina behov.

```
# -*- coding: utf-8 -*-
#====================================================== #
#  Program: <Programname (with path)>.                  #
#  Author: <Name of programmer of originalversion>      #
#  Version: <Current version (date/programmer)>         #
#  <Previous versions>                                  #
#  Purpose: <Description of the program's purpose>      #
#  Files: <Short description of in/out- and logfiles>   #
#  Action: <Does the program mail someone etc?>         #
#  Call: <program options, who starts it? cron?>        #
#  Examples: <How to run/use the program>.              #
#  Todo: <Future updates etc>                           #
#  Misc: <Text>.                                        #
#                                                       #
# Related bins: <Scripts/bins which are related>        #
#======================================================#
#
```

**Granskningsprotokoll**

Detta protokoll är valfritt att använda, men kan ge ett gott stöd och lite tid och möjlighet till reflektion över det man skrivit. I den här uppgiften är inte alla punkter i protokollet tillämpbara.

```
Uppgift: uppgiftens namn, kursdeltagare


Användarvänlighet
_ Informativa utskrifter / lättbegripligt (grafiskt) gränssnitt
_ Enkel inmatning


Programmerarvänlighet
_ Vettiga namn
_ Kommentarer
_ Konsekvent språk
_ Konsekvent typografi
_ Felhantering


Strukturering
_ Lämplig uppdelning i funktioner
_ Lämplig uppdelning i klasser (ej obligatoriskt)
_ Temporära variabler så lokalt som möjligt
_ Återanvändbara funktioner/klasser
```

_ In- och utdata till funktioner
_ Flexibelt/utbyggbart program
_ Ingen koduppprepning
_ Ingen hårdkodning

Följande punkter är nödvändiga (måste alltid åtgärdas)

_ Uppfyller kraven i lydelsen

_ Detaljförståelse