Avdelningen för informatik och datateknik, JTH
Introduktion till programmering, TIPK17
Anders Carstensen

JÖNKÖPING UNIVERSITY

Riktlinjer för utvärdering av den frivilliga delen av projektuppgiften

Nedan följer riktlinjer för hur projektet bedöms för olika betygsnivåer. Riktlinjerna tjänar två syften dels är det en anvisning för studenten om vad som krävs för ett visst betyg dels är det en bedömningsgrund för examinator vid bedömningen av det totala resultatet av projektarbetet. Emellertid kommer det alltid att vara en bedömningsfråga. Även om man i sin kod har ”bakat in” en viss aspekt måste det utifrån helheten bedömas hur relevant detta är.
Projektuppgiften för betyg 4 eller 5 är inte obligatorisk (se kursguiden). Nedan beskrivs även krav på betygsnivå 3 för att ge en balanserad bild av kraven.

Betygsnivån utgörs av en samlad bedömning av kod och projektrapport (ev. muntlig redovisning om så krävs av examinator).

[För betygsnivå 3 krävs:

- Beskriva problemet, och hur det har lösts
- Strukturera programkoden med sekvens, selektion och iteration
- Ha gjort rätt val av datatyp vid deklaration av variabler, samt gjort en korrekt konvertering mellan datatyper
- Ha infört basal felhantering, t.ex. ta hand om felaktig inmatning
- Kunna använda funktioner med värdeanrop
- Använda fält (arrayer) där så är tillämpligt och då ofta i ett iterativt förlopp
- Dokumentera på ett grundläggande sätt användningen av variabler och olika funktioner, framför allt genom bra och beskrivande variabelnamn och namn på funktioner. Bra kommentarer i koden där så behövs. Funktioner och deras gränssnitt skall förklaras så att deras användning blir tydlig.
- Ha gjort en enkel filhantering: skapa och knyta ett filobjekt till en textfil för läsning resp. för skrivning till fil; avluta filhanteringen på ett korrekt sätt.
- Dokumentera problemlösningen på ett förståeligt sätt]

För betygsnivå 4 krävs:

- Uppfylla betygsnivå 3 samt lämna in projektrapporten före deadline.
- Använda konstanter för återkommande värden (istället för att hårdkoda dessa). (Koden görs mer generell då en ändring av värdet endast behöver göras på ett ställe i koden)
- Använda komplexa datatyper där det är tillämpligt och väl motiverat. Exempel på detta kan vara att skapa en egendefinierad datatyp mha struct.
- Utföra mer avancerad filhantering. Här kan det handla om att hantera olika situationer beroende på om filen redan existerar eller inte.
- Visa på basala kunskaper i användning av pekare.
- Utföra en mer avancerad dokumentation. Programmets struktur och uppdelning skall tydligt beskrivas. Det skall finnas en tydlig och genomtänkt strukturering mha olika funktioner samt hantering av data med väl genomtänkta variabler
- Följa en konsekvent stil för t.ex. namngivning av variabler, konstanter och funktioner.

För betygsnivå 5 krävs:

- Uppfylla betygsnivå 4
- Ha genomfört en omfattande analys av de krav som ställs på applikationen och presenterat detta dels i projektrapporten dels i dokumentationen i koden. Här ställs ännu hårdare krav på dokumentation och en gediget genomförd projektrapport.
- Visa på kunskaper i dynamisk minnesallokering och förståelse för när detta är tillämpbart.
- Skickligt genomföra problemlösningen. Det kan t.ex. handla om att undvika redundanta kodrader (sådana som egentligen inte för något värde till applikationen). Det kan också handla om en mer elegant lösning istället för en mer omfattande och klumpig lösning.
- Klara av att skriva generell kod som är återanvändbar. Det kan handla om välskrivna återanvändbara funktioner.

Generellt:

Kod skall i möjligaste mån vara självbeskrivande. Med det menas bl.a. att man skall undvika tillkrånglade lösningar som är svåra att förstå. Kommentarer i koden skall endast finnas där det är nödvändigt t.ex. för att förklara längre förlopp eller orsakssamband som är svåra att direkt överblicka i koden. Rapporten skall dock ge en överblick av den lösning som tagits fram.

Projektuppgiften kan för betygsnivå 4 innebära att man gör vissa begränsningar av hur funktionaliteten är beskriven. Det är upp till den som löser projektuppgiften att inse vad som behöver göras (dvs hur implementeringen bör vara gjord). Man bör dock alltid starta med att ”Top down” försöka analysera uppgiften och dela upp ett större problemområde i mindre delproblem. Dessa delproblem kan sedan lösas med relevanta funktioner. Tänk igenom en funktions gränssnitt och testa detta gränssnitt innan resten av koden skrivs.