Test och kvalitetssäkring

Uppgiftstid: 2020-01-06—2020-01-10, **Tid:** Inlämning senast 23.55 10 januari 2020

Hjälpmedel
Kodstandardsdokument på moodle, referensbok, material på moodle, dator, internet, Unity cheet sheet. Kodexempel på er dator från kursen.

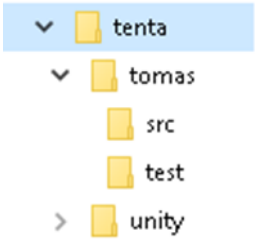
Länk till sammanfattningen om möjliga asserts för unity (finns även under docs i unity)

https://github.com/ThrowTheSwitch/Unity/blob/master/docs/UnityAssertionsCheatSheetSuitableforPrintingandPossiblyFraming.pdf

Länk till unity ifall ni inte redan har det

https://github.com/ThrowTheSwitch/Unity/archive/master.zip

Allmänna anvisningar
Alla uppgifter löses i kod. Det finns mycket att göra med uppgifterna och det är inte krav på att allt ska vara perfekt, men gör det så bra som möjligt. Ni behöver visa att ni kan skriva, kod och lämpliga testfall med runners samt kompilera och köra dessa med hjälp av make-file. Ni har fram till fredag kväll på er, se därför till att använda tiden på bästa sätt. Det kan vara värt att lämna vissa delar för stunden och återvända om det finns tid kvar på slutet. Om ni vill jobba testdrivet fullt ut genom att första skriva testfall, eller första skriva kod och sen testfall avgör ni själva. **Följ de konventioner för kod** som finns på moodle i separat dokument.



Unity ska ligga i roten, parallellt ska det finnas en mapp med ert fornamn, däri en testmapp och en srcmapp och en makefile. Då uppgifterna handlar om samma kodbas ska koden ligga under samma uppgift. Lägg alla lösningar i en gemensam mapp, förslagsvis direkt under eller nära roten för att minimera risken för problem med långa sökvägar, t.ex. c:/uppgift. Undvik också å, ä och ö och ha inga mellanslag i era sökvägar. När ni är klara eller tiden börjar ta slut, packa ihop allting (alla uppgifter ni skrivit och unity så att filerna ligger på rätt ställe tillsammans med makefile till en zip-fil (inte rar) med ert namn och ladda upp den på moodle till den inlämning som heter hemtentamen. Var noga med att ni lämnar in rätt fil. Varje påbörjat försök till lösning på någon av uppgifterna kan visa på kunskap även om uppgiften inte är löst fullt ut. Varje uppgift kommer sedan köras mot ett antal testfall så var noga med alla detaljer och tänk test och kvalitet.

Om något strular och inte fungerar, gör en tillfällig lösning som hjälper er komma vidare, kommentera att den är tillfälligt och förklara vad som strulat, tänk på att kursen handlar om test och kvalitet!

Lycka till!

Ansvarig lärare: Tomas Berggren

Test och kvalitetssäkring

Företaget TomasBData AB håller på att utveckla ett program där man bland annat ska kunna räkna ut omkrets, area och volym.

Uppgift 1 (20 poäng)

Förslagsvis med hjälp av TDD, skriv ett program som låter användaren först välja om programmet ska räkna ut omkrets, area eller volym på en fyrkantig rektangel/låda i lämplig enhet. Därefter ska användaren mata in längden på två sidor samt höjd när det är aktuellt.

De värden som matas in ska vara numeriska, större än 0 och mindre än 1000. Om värdena inte uppfyller kraven ska ett felmeddelande skrivas ut och användaren ska ges möjlighet att mata in ett nytt värde.

När uträkningen är klar ska resultatet presenteras på skärmen. Programmet ska kunna göra flera beräkningar utan omstart.

Testfall

Om alla sidors längd är 10 är omkretsen 40, arean 100 och volymen 1000.

Uppgift 2 (20 poäng)

Med hjälp av unity, gör minst tre lämpliga enhetstest för att testa respektive funktionalitet i uppg. 2

Tänk på att det inte bara är ASSERT_EQUAL som går att använda

Förslag på testfall:

Ett testfall som testar omkretsberäkningen

Ett testfall som testar areaberäkningen

Ett testfall som testar volymberäkningen

Tänk på att testa med både giltiga och ogiltiga värden

Uppgift 3 (10 poäng)

gör en testrunner och ett alltest-program som kör alla testen skapade i uppgift 3 och kan kompileras till en körbar exefil.

Uppgift 4 (10 poäng)

Skapa en makefile med minst två targets, ett för att bygga programmet och ett för att bygga test av funktionalitet med hjälp av den testrummer som skapats i uppgift 3. Om make-regeln får till svar, already up to date behövs en så kallad .phony i er makefile som tvingar omkompilering.

.PHONY: all test clean (ex från makefile på de targets den gäller)

Lämna in en zip av hela mapp som innehåller kod, testkod, makefile och unity på moodle.

Varje uppgift bedöms och poängsätts, lämna in vad ni gjort, varje steg kan ge poäng även om koden ger kompileringsfel

De filer ni lämnar in kommer stämmas av mot kodstandard och utsättas för ett antal test, varje test som inte ger godkänt kan ge poängavdrag.