Laboration 3 1DV404 – Testning

Uppgifterna skall lösas *individuellt*. Om ni använder er av material/lösningar som andra tagit fram skall detta tydligt framgå av redovisningen(referenser). Vi lägger stor vikt vid att du kan förklara och därmed visa på en förståelse för vad du gjort. Använd företrädelsevis UML för diagram. Tänk på att alla diagram måste dokumenteras så att vi förstår att du har förstått. Jag använder orden klass och metod i denna och andra laborationsbeskrivningar. De allra flesta programmeringsspråk har något liknande, inte nödvändigtvis klasser men i alla fall moduler. Om jag skulle skriva så alla begrepp passade in då vi är öppna för flera olika programmeringsspråk så skulle laborationsbeskrivningen bli än mer svårtydd.

Lämna in diagram, kod och övrig dokumentation enligt beskrivningen i uppgifterna. Du skall föra en tidslog för hela laborationen. I loggen anger du hur mycket tid du förbrukat, vilken dag, och vad du arbetat med. Sammanställ dessa siffror och inkludera i din slutrapport. Tänk även på att försöka formulera dig som om rapporten var ”på riktigt” och att språk och struktur lever upp till kraven på en formell rapport!

Om något saknas kommer inlämningen att underkännas direkt!

Glöm inte att reflektion är en förutsättning för att man skall dra nytta av erfarenheter. Reflektera gärna kring er planerade tid och det verkliga utfallet. Vilka orsaker ser du? Gör detta kontinuerligt så bli sista uppgiften enklare.

Redovisning.
Kontakta din handledare för redovisning av laborationen. **All redovisning sker på schemalagd tid!!! Gäller även distansstudenter.**

Paketera allt material i en fil. Namnge filen:
1DV404_HT14_[labgrupp]_3_[användarnamn]

Alternativt kan ni skicka en länk till ett repositorium där dina filer finns. Alla dokument skall vara i PDF format. Skicka direkt till er laborationshandledare enligt deras instruktion.

Laborationen skall skickas in senast den 18:e december kl. 15:00
Boka redovisning senast i samband med det att ni skickar in!

Linnaeus University

Department of Computer Science
Dr. Jesper Andersson, Head of Department
SE-351 95 Växjö • *Visiting address* B-House, Campus Växjö • *Telephone* +46 470 708460 • *Email*
jesper.andersson@lnu.se • Lnu.se

Gymnastiktävlingssystem - problembeskrivning

Systemet är tänkt att automatisera, uppsättningen och hanteringen av en gymnastiktävlingssäsong. Systemet skall hantera träffar med flera tävlingar, registrering av lag, poängbedömning och poängräkning samt administration.

Här följer en kort beskrivning av en ”gymnastikligan” och en av deras tävlingar. ’Ligan’ är en grupp av lag som tävlar mot varandra. Varje lag tillhör en klubb som tillhör ligan och rekryterar medlemmar för att delta i tävlingar.
En typisk träff med gymnastikligan består av flera deltävlingar som arrangeras under loppet av en dag. Till exempel kan det finnas en kvinnor allround, en kvinnor individuell, en för män allround, och så vidare. Det kan också finnas junior och senior tävlingar.

Ett lag som anmäler sig till en träff anmäler sig automatiskt till alla deltävlingar för den träffen där de kan delta. I varje deltävling kan det finnas ett antal grenar och alla lagen i en deltävling anmäler samma antal tävlande till alla grenar.
Varje deltävling består av ett antal grenar. Till exempel består damernas tävlingar av balansbom, hopp, barr, och fristående. Alla grenar i en deltävling genomförs parallellt. Alla tävlande gymnaster i ett lag utför sina övningar och poängbedöms, för att sedan rotera vidare till nästa gren.
Varje gren har en jury för poängbedömning. Juryn består av kvalificerade domare som är certifierade för att döma grenen. Varje domare poängbedömer gymnasternas insatser och skickar resultatet vidare till en sekreterare. Sekreteraren kastar ut de högsta och lägsta poängen och beräknar därefter ett medelvärde. Medelvärdet utgör gymnastens poäng i grenen. Lagets poäng är summan av alla gymnasters "poäng”. Tävlingspoängen för ett lag blir på samma sätt summan av poängen för varje gren och så vidare.
Förutom att arrangera och genomföra gymnastikligans träffar, planerar ligan schemat för säsongen, ser till att kvalificerade domare finns på alla träffar och fördelas på deltävlingar och grenar, registrerar lag och gymnaster och publicerar säsongsresultat.

Uppgift 1 – Planera
Planera arbetet i steg/moment efter uppgifterna nedan. Ange din planerade tid med en noggrannhet på 15 min. Exempelvis 3h och 15 min eller 4h och 30min. Ta hänsyn till att du kan behöva läsa in dig, så planera även tid för detta. Använd fasta punkter för att underlätta din planering, exempelvis handledningstillfällen och uppgiftsuppdelningen nedan. Kanske skall du planera med en eller två dagar tillgodo för att vara på den säkra sidan.

Uppgift 2 – Testplan

Lämna in: Planeringsdokument

I ett första steg behöver du ta fram en testplan som beskriver ditt testprojekt. Utnyttja kravspecifikationer och annat material från laboration 2. Glöm inte kvalitetskraven som kräver lite speciell hantering. Här kan ni själva behöva leta upp lämpliga testtekniker. Det finns ingen testplansmall för Open UP som ni kan använda som inspiration men där finns annan text som beskriver testfall mm. Mallar för planen kan man hitta om man söker.
http://epf.eclipse.org/wikis/openup/

Lämna in: Testplan

Uppgift 3 – Design och implementation
För att vi skall kunna testa måste vi ha lite kod. Målet med den här uppgiften är att du skall designa och implementera ett helt eller stora delar av ett användningsfall. Välj ut ett eller två användningsfall. Analysera dem för att identifiera klasser. Implementera och dokumentera klasserna.

Lämna in: Användningsfallsmodell Användningsfallsbeskrivningar Klasser med beskrivningar Kod

Uppgift 4 – Enhetstestning
Välj ut minst två av dina klasser från föregående användningsfall och genomför en analys av dessa. Använd xUnit arkitekturen (se föreläsning) och specificera en testsvit som innehåller, testfixturer och testfall. Varje testfall skall beskriva testdata och test mot förväntat resultat. Beskriv även hur du kommit fram till detta.

Lämna in: Testsvitsdokumentation med Beskrivningar av testfixturer och testfall. Motivering till testsvitens sammansättning och testfallen

Uppgift 5 – Implementera testsviten och kör
Bygg testklasser för dina klasser, Exekvera testklasserna och samla in testdata. Analysera och föreslå förbättringar.

Lämna in: Kommenterad kod för testklasserna Dokumentation av de utförda testerna. Testdata

Uppgift 6 – Integrationstestning
Ta de användningsfall som du valde i uppgift 3. Använd metoden från föreläsningen och ta fram testbeskrivningar för integrationstestfall. Implementera testklasser och exekvera. Dokumentera testdata.

Lämna in: Dokumentation av testfall och test data. Stegvis beskrivning för hur du arbetade fram testfallen och testdatan. Kommenterad kod för testklasserna Dokumentation av de utförda testerna. Testdata

Uppgift 7– Reflektion
Reflektera kring svårigheter i att planera och genomföra uppgiften (min 1/2 A4, max 1 A4)