TDP004 - Dugga

2016-12-01

Regler

- All kod som skickas in för rättning ska kompilera och vara väl testad.
- Inga elektroniska hjälpmedel får medtas. Mobiltelefon ska vara avstängd och ligga i jacka eller väska.
- Inga ytterkläder eller väskor vid skrivplatsen.
- Student får lämna salen tidigast en timme efter tentamens start.
- Vid toalettbesök ska pauslista utanför salen fyllas i.
- All form av kontakt mellan studenter under tentamens gång är strängt förbjuden.
- Böcker och anteckningssidor kan komma att granskas av assistent, vakt eller examinator under tentamens gång.
- Frågor om specifika uppgifter eller om tentamen i stort ska ställas via tentasystemets kommunikationsklient.
- Systemfrågor kan ställas till assistent i sal genom att räcka upp handen.
- Endast uppgifter inskickade före tentamenstidens slut rättas.
- Ingen uppgift kan kompletteras under tentamens sista kvart.
- En uppgift kan som regel kompletteras tills den är antingen "Godkänd" eller "Underkänd". En uppgift bedöms som "Underkänd" om ingen markant förbättring skett sedan tidigare inlämning.
- Kompilerande kod, fullständig kravuppfyllnad och följande av god stil och goda konventioner enligt god programmeringssed är krav för att en uppgift ska bedömas "Godkänd".

Hjälpmedel	En C++-bok (t.ex. Stroustrup) Ett A4-ark med egna anteckningar

Information

Betygssättning vid tentamen

Tentamen består av ett antal uppgifter på varierande nivå. Uppgifter som uppfyller specifikationen samt följer god sed och konventioner ges omdömet "Godkänd". Annars ges omdömet "Kompletteras" eller "Underkänd". Tentamen kräver två godkända uppgifter för betyg 3. Alla betygsgränser ses i tabell 1 och 2. *För betyg 3 har du alltid hela tentamenstiden, varken mer eller mindre.* (För student som från LiU fått rätt till förlängd skrivtid förlängs betygsgränserna i proportion till den förlängda skrivtiden.)

Tid	**Lösta uppgifter**	**Betyg**	**Tillgodo**
4 h	3	5	inget
2.5 h	2	4	inget
4 h	2	3	inget
4 timmar	1	|	löst uppgift

Tabell 1: Betygsättning vid förtentamen (dugga), 4 uppgifter ges

Tid	**Lösta uppgifter**	**Betyg**
3 h $+B$	3	5
4 h $+B$	4	5
4 h $+B$	3	4
2 h $+B$	2	4
5 timmar	2	3

Tabell 2: Betygsättning vid sluttentamen, 5 uppgifter ges

Bonustid ($+B$)

All bonustid gäller endast under den första ordinarie tentamen i samband med kursen (januari). Varje moment i kursen som ger bonus ger 5 minuter extra tid för högre betyg på sluttentamen, upp till maximalt 45 minuter. Detta är markerat med $+B$ i tabellen där bonus räknas.

Tillgodoräknanden

Tillgodoräknanden gäller endast under den första ordinarie tentamen i samband med kursen. Om du på förtentamen endast lyckas lösa en uppgift kan du tillgodoräkna motsvarande uppgift på sluttentamen (se tabell 1). Du har då bara en uppgift kvar till betyg 3. För högre betyg (om du löser mer än en uppgift på förtentamen) kan du inte tillgodoräkna något. Om du på sluttentamen löser en andra uppgift snabbt och vill sikta på högre betyg kan du lösa den tillgodoräknade uppgiften "igen", men den räknas endast mot högre betyg, **inte** som andrauppgift för betyg 3.

Inloggning

Innan du loggar in ska du välja "Exam system" från sessionsmenyn i nedre vänstra hörnet av inloggningsrutan. Därpå loggar du in med dina LiU inloggningsuppgifter. Du kommer nu in i tentainloggningen och ska börja med att välja språk. Ta den flagga som ser minst Engelsk ut så blir det Svenska (valet spelar ingen roll för dig som kan båda språken). Följ instruktionerna på skärmen så långt det går tills du ska mata in ett engångslösenord. Tag fram ditt LiU-kort och visa det för assistent eller vakt i sal för att få detta lösenord.

Skrivbordsmiljön

När du kommit in i tentasystemet har du en normal skrivbordsmiljö (Mate-session i Linux Mint) med grön skrivbordsbakgrund. Efter en stund kommer även din kommunikationsklient att dyka upp automatiskt. Startmenyn är nedskalad till enbart det som examinator bedömt relevant. Andra program kan fortfarande finnas tillgängliga genom att starta dem från en terminal. Observera att en del program som använder nätverkstjänster inte fungerar normalt, eftersom nätverket inte är åtkomligt.

När du är inloggad är det viktigt att du har tentaklienten igång hela tiden. Om den inte dykt upp fem minuter efter inloggning och inte heller när du startar den manuellt från menyn (fisken) tar du kontakt med assistent eller vakt i sal.

Terminalkommandon

`e++17` används för att kompilera med "alla" varningar *som fel*.
`w++17` används för att kompilera med "alla" varningar. **Rekommenderas.**
`g++17` används för att kompilera `utan` varningar.
`valgrind --tool=memcheck` används för att leta minnesläckor.

C++ referenssidor

På tentan har du *experimentiell* tillgång till referenssidorna på `http://www.cppreference.com/` via webbläsaren Chrome. Starta `chromium-browser` i terminalen eller välj lämpligt alternativ från startmenyn. Observera att allt utom referenssidorna är avstängt. Om du inte kan komma åt en sida du tycker hör till referenssidorna (som kanske blockerats av misstag) kan du skicka ett meddelande via tentaklienten. Då lösningen är på experimentell nivå kan det hända att problem uppstår, t.ex. att proxyn inte går att nå. Tag då hjälp av assistent i sal.

Givna filer

Eventuella givna filer finns i katalogen `given_files`. Denna underkatalog är skrivskyddad, så det är ingen risk du råkar ändra på dessa filer. Skrivskyddet gör dock att du måste kopiera in de givna filer du vill använda till tentakontots hemkatalog. Hur du listar och kopierar filer ska du kunna. Hemkatalogen står du i från början, och du kommer alltid tillbaka till den genom att bara exekvera kommandot `cd` i terminalen. Hemkatalogen heter alltid `/home/student_tilde` om du undrar.

Avslutning

När dina uppgiftsbetyg och ditt slutbetyg i kommunikationsklienten stämmer överens med det du förväntar och du är nöjd, eller när tentamenstiden är slut, är det dags att logga ut. Hinner du inte se ditt betyg får du höra av dig till examinator via epost efter tentamen.

Avsluta alla öppna program och tryck på knappen märkt "Exit" i menyn längst ner på skärmen och välj "ok". Vänta ett tag och tryck sedan på knappen "Avsluta tentamen" när det är möjligt. När detta är gjort är det omöjligt att logga in igen. Lämna inte datorn förrän du ser den vanliga inloggningsskärmen med blå bakgrund. Anmäl till assistent eller vakt om inloggningsskärmen inte dyker upp inom en minut så åtgärdar vi problemet.

Uppgift 1 - Yatzy

I spelet Yatzy ska spelaren kasta fem tärningar och välja vilka hen vill spara för att uppnå vissa kombinationer av tärningar. Några exempel på kombinationer är “ettor” där man endast vill få så många tärningar med en prick som möjligt, “par” då man vill ha två tärningar med samma värde och “liten stege” då man ska få en tärning av varje typ i intervallet 1 till 5.

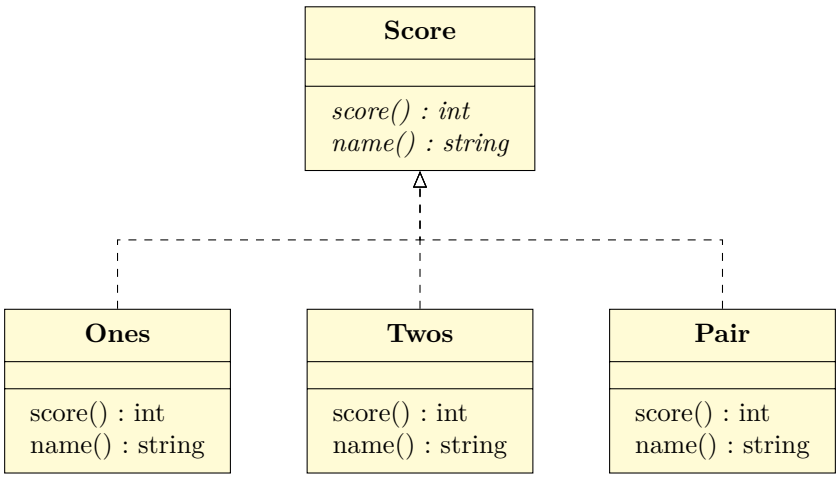
Funktionella krav

I denna uppgift ska du skapa en del av poängberäkningen i spelet Yatzy genom att skapa en klassheirarki där varje klass beräknar värdet hos en samling tärningar (som representeras av en `vector<int>`). De klasser du ska skapa ges av klassdiagrammet i figur 1. Kopiera filen `given_files/uppgift1.cc` till din arbetskatalog och lägg till klasserna för att uppnå resultatet i kodruta 1.Klassernas score-funktioner ska göra följande:

- `Ones` och `Twos` ska returnera summan av alla ettor respektive tvåor som skickas in.
- `Pair` ska ge summan av det högsta paret.

Ickefunktionella krav

- Klassen `Score` ska vara en abstrakt klass
- Nyckelordet `const` ska användas på ett korrekt sätt där det passar
- Ingen av klasserna ska ha datamedlemmar



Figur 1: Klassdiagram för Yatzy
Parametertypen till medlemsfunktionen score är utelämnad av utrymmesmässiga själ

Kodruta 1: Utskrift vid korrekt implementation, obeservera att Pair alltid ger värdet av högsta paret

```
Tärningar   1:or  2:or   Par
1 2 1 3 4      2     2     2
3 3 1 1 1      3     0     6
5 5 6 6 3      0     0    12
```

Uppgift 2 - Presidentsval

Än en gång har en kandidat vunnit presidentsvalet i USA trots att han inte haft majoriteten av rösterna (popular vote) på sin sida. Anledningen till detta är att varje person inte innebär en röst utan en persons röst indikerar för så kallade elektorer vilken kandidat som denne ska rösta på. Varje stat har ett antal elektorer och i denna uppgift gör vi det något förenklade antagandet att den som vinner majoriteten av rösterna i en viss stat får alla denna stats elektorsröster på sin sida.

I filen `given_files/election.txt` finns, utöver en rubrik, resultatet för valet uppdelat med en stat per rad. Du kan anta att filen alltid har formatet enligt nedan (där `<TAB>` är ett tabulatortecken, `\t`):

`Stat <TAB> Elektorer <TAB> Antal röster <TAB> Röster på Trump <TAB> Röster på Clinton`

Din uppgift är att sammanställa filens innehåll och skriva ut hur stor andel av rösterna samt hur många elektorsröster varje kandidat fick enligt körexempel 2.

Funktionella krav

- Utskriften ska matcha körexemplet exakt.
- Namnet på filen ska ges på kommandoraden, om något går fel ska programmet skriva ut ett tydligt felmeddelande och avslutas.

Tips

En klass eller struct för att hantera antingen informationen för en stat eller för en kandidat kan vara bra att ha (eller kanske båda).

Kodruta 2: Körexempel

```
$ ./a.out given_files/election.txt
          Andel röster  Antal elektorer
Clinton         48.02%               233
Trump           47.01%               305
```

Uppgift 3 - Formaterad utskrift

Kommunikation med användaren är bland det svåraste man kan göra som programmerare. I denna uppgift ska du skapa en klass som kan användas för att hjälpa programmerare att formatera text. I filen `given_files/uppgift3.cc` finns exempel på hur denna klass ska användas och fungera.

Funktionella krav

Skapa klassen `Formatter` med medlemsfunktioner enligt nedan där varje funktion returnerar en sträng med en viss förinställd bredd. Om strängen som skickas till funktionen är kortare än den inställda bredden ska den genererade strängen fyllas ut med mellanslag, annars returneras den inskickade strängen.

- Konstruktor som tar emot och sparar en bredd, `width`, på den formaterade texten.
- `string adjust_left(string str)` returnerar en sträng med längd av minst `width` tecken där `str` är vänsterjusterad.
- `string center(string str)` returnerar en sträng med `str` centrerad.
- `string adjust_right(string str)` högerjusterar `str`.

Uppgift 4 - Skulder

När man är flera personer som brukar turas om att exempelvis köpa lunch och fika åt varandra kan det vara svårt att hålla koll på alla skulder. I denna uppgift ska du skapa ett program som låter användaren mata in valfritt antal köp på formatet
"`<Betalarens initialer> <Mottagarens initialer> <pris>`" och därefter skriva ut en sammanställning av samtliga skuldsaldon enligt körexemplet nedan.

Funkionella krav

- Programmet låter användaren mata in köp på standard inmatningsström (cin) till filslut (ctrl-d).
- Utskriften ska vara sorterad på namn (initialer).
- Formatet på utskriften ska matcha körexemplet nedan. Ett positivt värde betyder att någon är skyldig personen pengar (spelar ingen roll vem) och negativt värde indikerar att personen bör stå för ett inköp.

Ickefunktionella krav

- Du ska använda någon lämplig STL-container för att lagra informationen om skulder.

Tips

- Summan av alla värden i tabellen ska bli 0.
- Du behöver inte hålla reda vem som är skyldig vem pengar, det är endast totala skulden/tillgodot för varje person som är av intresse.

Körexempel

```
AE EM 102
EM BD 23.1
CK EM 15
FF CK 17.2
AE CK 9.95
DH FF 35.2
DH EM 65.8
AE EM 91.55
<CTRL-d>
Namn    Saldo
-------------
AE     203.50
BD     -32.10
CK     -12.15
DH     101.00
EM    -251.25
FF     -18.00
```