Regler

Regler - generellt

- Frågor ställs genom Zoom. Klicka på knappen (Ask for Help) så hjälper vi dig så fort som möjligt.
- Slutinlämningen av din kod ska vara skriven enligt god programmeringssed för C++. Det betyder att tekniker som finns för att underlätta användning, utveckling, felsökning och underhåll ska användas.
- Du ska sitta i en ostörd miljö utan andra personer i samma rum. Du ska hela tiden vara uppkopplad och synlig i Zoom. Om du är klar med en deluppgift och har lämnat in den kan du sätta en lapp framför din kamera där det står "Rast".
- Du kommer behöva legitimera dig under tentamen. Ha ditt foto-id redo.
- Var beredd på att i efterhand kunna redogöra för dina svar.
- Om du behöver ta en rast, lämna in de filer du jobbar på just då till "2020-06-01: Rast" i Lisam.
- Alla former av regelbrott medför att din tentamen underkänns direkt.
- Information och dina givna filer kommer att publiceras under tentans gång på följande sida: `https://www.ida.liu.se/~TDDI82/exam/2020/2020-06-01/index.sv.shtml`

Regler - Tider

- STL-uppgiften publiceras 14:30 och lämnas senast in 16:00.
- 16:00 till 16:15 är det rast.
- Mall-uppgiften publiceras 16:15 och lämnas senast in 17:45.

Regler - hjälpmedel

- All kommunikation är förbjuden, undantaget frågor till kurspersonal.
- All form av kopiering eller avskrift är förbjuden.
- Alla källor du tar inspiration av ska redovisas.
- Du får använda `cppreference.com` fritt.
- Du får använda en valfri C++ bok
- Du får använda en A4 sida med anteckningar. Du måste lämna in dina anteckningar i inlämningen "2020-06-01: Anteckningar" i Lisam innan du börjar skriva tentan.

Regler - betyg

Tentan består av två uppgifter, en för STL och en för Mallar. Varje del bedöms som U, G eller VG. För att få G på en uppgift måste du ha uppfyllt *majoriteten* av kraven som presenteras i uppgiften. Du måste även ha gjort ett försök till att svara på alla frågor i uppgiften. Detta betyder alltså att du kan få godkänt även om du inte har en fullt fungerande lösning.

I tabellen nedan presenteras betygsgränserna:

Betyg	STL	Mallar
3	G	G
4	G	VG
4	VG	G
5	VG	VG

Det krävs också att du fyllt i tentans regelinlämning, vilket bekräftar att du läst tentans regler och tänker följa dem. Bonus insamlad under kursen gäller inte denna tenta. All bonus behåller du till dess att nästa ordinarie datortenta kan genomföras.

Regler - inlämning

Inlämning görs via Lisam, på följande sida: `https://studentsubmissions.app.cloud.it.liu.se/Courses/TDDI82_2020VT_CF/`. Du kan även hitta sidan genom att gå in på TDDI82 i `https://lisam.liu.se` och sedan klicka på “Inlämningar” i menyn.

Följande inlämningar kommer vara tillgängliga vid de specificerade tiderna:

- 2020-06-01: Regler (14:00-17:45)
- 2020-06-01: Anteckningar (14:00-15:00)
- 2020-06-01: Rast (14:30-17:45)
- 2020-06-01: STL Kod (14:30-16:00)
- 2020-06-01: STL PDF (14:30-16:00)
- 2020-06-01: Mallar Kod (16:15-17:45)
- 2020-06-01: Mallar PDF (16:15-17:45)

I “2020-06-01: STL Kod” lämnar du in koden till din lösning för STL-uppgiften och i “2020-06-01: STL PDF” lämnar du in dina svar på frågorna till STL uppgiften som en PDF fil. Likadant för Mall-uppgiften.

Regelinlämning

Innan du börjar arbeta på första uppgiften måste du göra en inskickning till “2020-06-01: Regler” i Lisam (se ovan).

Du kan skicka ett meddelande via Lisam där du bekräftar att du har läst reglerna och att du tänker följa dem.

Kom ihåg att om du har en sida med anteckningar måste du skicka in denna i “2020-06-01: Anteckningar” i Lisam. Om det är datorskrivet kan du skicka in dem som de är. Om de är handskrivna kan du antingen skanna in dem eller ta en bild.

Gör detta innan du börjar jobba på tentan!

STL

Tidsplan

Vi rekommenderar att du lägger 45-60 minuter på att skriva kod och att du sedan lägger resten av tiden på att svara på frågorna.

Uppgift

I denna uppgift ska du endast använda STL algoritmer. Du ska undvika loopar i den mån du kan. Din uppgift är att skapa ett program som utför följande steg:

Notera: Steg 4, 5, 6 och 8 varierade per person under tentan. Alla möjliga alternativ presenteras nedan.

1. Läs in ett heltal `m` från `std::cin`.
2. Skapa en `std::vector<int>` vid namn `v`.
3. Fyll `v` med heltal från `std::cin` tills användaren trycker `ctrl+D`.
4. Utför en av följande operationer på varje heltal `x` i `v`:
   - Multiplicera `x` med två.
   - Om `x` är udda, subtrahera 1, annars gör ingenting.
   - Multiplicera `x` med (`x` + 1).
   - Multiplicera `x` med fyra och subtrahera två.
   - Addera tre till `x` och multiplicera resultatet med fyra.
5. Sortera `v` enligt följande kriterie: `x` ska sorteras före `y` om:
   - avståndet från `x` till `m` är mindre än avståndet från `y` till `m`. Avståndet mellan två heltal `a` och `b` beräknas genom att ta absolutbeloppet av `a - b`.
   - `x` modulo `m` är mindre än `y` modulo `m`.
   - `x` är större än dubbla `y`.
6. Ta bort
   - alla heltal som är mindre eller lika med noll.
   - det största heltalet i `v`.
   - det första heltalet i `v` som är jämnt delbart med tre.
7. Skriv ut innehållet av `v` på en egen rad, där varje heltal är separerat med mellanslag.
8. Skriv ut
   - summan av `v`.
   - antalet jämna tal i `v`.

Notera att det är ditt ansvar att visa på hur väl du kan standardbiblioteket. Det är därför viktigt att du demonstrerar så mycket som bara möjligt. Det är bättre att ha utkommenterade delar som är nästan rätt än att ha en fungerande loop.

Frågor

Svara på ***ALLA*** frågor nedan angående din lösning. Skriv dina svar i ett separat PDF dokument. Detta dokument ska som mest vara 1000 ord långt.

1. Redogör för minst två *olika* algoritmer som kan användas för att implementera stegen ovan. Förklara *varför* dessa algoritmer är lämpliga. Notera att kodexempel kan underlätta, men det är inte ett krav.
2. Vilka fördelar finns det med att lösa problemet med algoritmer istället för med loopar? Förklara varför dessa fördelar inte finns när vi använder loopar.
3. Finns det några nackdelar med att använda algoritmer?
   - Om ja: vilka är nackdelarna och varför uppstår de?
   - Om nej: Förklara vilka fördelar det finns med att använda loopar istället för algoritmer.
4. Skulle ditt program kunna förbättras med ett bättre val av databehållare? Diskutera varför eller varför inte.

Svara på frågorna ovan så gott du kan. Vi letar inte nödvändigtvis efter ett specifikt svar utan vi vill se din tankegång.

Betyg

Denna uppgift har inget speciellt du behöver göra för VG. Vi kommer istället bedöma enligt följande kriterier:

G: Uppgiften löst med flera algoritmer utan allvarliga brister. Rimliga svar på frågorna.

VG: Uppgiften löst fullt ut med algoritmer utan misstag. Rimliga svar på frågorna med djupa resonemang och god motivering.

Detta innebär alltså att du måste lösa uppgiften enligt din bästa förmåga och svara på frågorna för att få chans till VG.