DD1352 Algoritmer, datastrukturer och komplexitet, hösten 2016
Lösningsförslag till mästarprov 2: komplexitet

1. **Ultrasmart maraton**
   Betrakta beslutsproblemet SPECIALPRIS definierat på följande sätt:

   Inmatning: En oriktad graf $G =< V, E >$ med positiva kantvikter.

   Fråga: Finns det en stig som besöker alla hörn precis en gång och som har en vikt $\geq M/2$, där M är den totala kantvikten i grafen?

   Vi ska visa att SPECIALPRIS är NP-fullständigt.

   Låt en lösning vara en stig given som en delmängd kanter $S \subseteq E$. Om vi får en lösning S given så kan vi kontrollera att den är giltig genom att kontrollera att S är en stig som verkligen besöker varje hörn i V precis en gång och att summan av vikterna i S är minst $M/2$. Detta tar bara linjär tid i grafens storlek. Därmed har vi visat att SPECIALPRIS ligger i NP.

   För att visa att SPECIALPRIS är NP-svårt så reducerar vi det kända NP-fullständiga problemet HAMILTONSTIG till SPECIALPRIS.

   HAMILTONSTIG definieras på följande sätt:

   Inmatning: En oriktad graf $G' =< V', E' >$.

   Fråga: Finns det en stig som besöker varje hörn i G' precis en gång?

   Givet grafen G' så konstruerar vi en ny graf G som innehåller samma hörn och kanter som G' och vars kanter alla har vikt 1. Dessutom lägger vi till ett nytt hörn t och kanter som förbinder t med varje hörn i V'. Dessa kanter har också vikt 1. Slutligen lägger vi till en nytt hörn s, samt en kant mellan s och t med en vikt som är större än den sammanlagda vikten av alla andra kanter i grafen G. Denna reduktion går att göra på linjär tid i grafens storlek.

   Låt oss först visa att om det finns en hamiltonsk stig i G', så går det att vinna ett specialpris i G. Den vinnande stigen går från s till t, vidare till startpunkten i en hamiltonsk stig i G' och följer sedan denna hamiltonska stig. Denna stig besöker varje hörn i G precis en gång och dess vikt är tillräckligt stor.

   Nu återstår att visa att om det finns ett specialpris i G, så finns det också en hamiltonsk stig i G'. En väg som ger ett specialpris måste börja (eller sluta) i s. Därefter måste den gå till t och sedan följa en av de nya kanterna från t. Eftersom vägen inte kan återvända till t, så måste den sedan bestå av en hamiltonsk stig i grafen G'.

   Alltså är SPECIALPRIS NP-fullständigt.

2. **Schemaproblemet**

   Låt oss införa ett mål *timelimit* och därigenom formulera följande beslutsproblem EXISTSSCHEDULE:

   Indata: antalet klassrum k, lektioner $lesson[1..n]$, mål *timelimit*. Varje lektion kan ha en uppsättning krav såsom uppgiftslydelsen beskriver.

   Fråga: Finns det en schemaläggning av dom n lektionerna i dom k klassrummen som inte bryter mot några av kraven och där alla lektioner slutar senast *timelimit*?

   Vi kan verifiera en ja-lösning till beslutsproblemet i polynomisk tid på följande sätt. Låt lösningen vara en schemaläggning av alla lektioner, dvs ett utpekat klassrum och en starttid för varje lektion. Verifikatorn går först igenom alla krav och kontrollerar att dom är uppfyllda. Sedan kollar den att alla lektioner är schemalagda i ett tillåtet klassrum vid en ickenegativ tidpunkt och att inga lektioner överlappar i tid i samma klassrum. Slutligen kollas att alla

lektioner slutar senast *timelimit*. Detta kan göras i linjär tid i antalet krav och kvadratisk tid i n. Därmed vet vi att problemet ligger i NP.

För att visa NP-svårighet kan vi reducera SUBSET SUB (delmängdssumma), där indata är en mängd positiva heltal P och ett mål K och frågan är ifall det finns en delmängd av talen i P vars summa är exakt K.

Låt σ vara summan av alla tal i P. Anta först att $\sigma \leq 2K$. Idén är att vi för varje tal i P skapar en lektion som är så lång som talet anger och schemalägger dessa lektioner i två klassrum med $timelimit = K$ så att dessa blir helt fyllda. För att båda klassrummen ska bli helt fyllda krävs att vi lägger till en extra lektion med längd $2K - \sigma$ (om inte $\sigma = 2K$, för då lägger vi inte till någon extra lektion alls). Lektionerna har alltså inga speciella krav.

Om det finns en delmängd av P med summan K så kan vi schemalägga motsvarande lektioner i klassrum 1 och övriga lektioner i klassrum 2. Det går bra eftersom återstående lektioner har totala längden $\sigma + 2K - \sigma - K = K$ och vi inte har några andra krav på lektionerna. Ordningen mellan lektionerna spelar alltså ingen roll.

Åt andra hållet, om det finns en schemaläggning av lektionerna i två klassrum så måste dom vara helt fyllda upp till $timelimit = K$ eftersom totala längden av alla lektioner är $\sigma + 2K - \sigma = 2K$. Kolla på ett klassrum där utfyllnadslektionen med längd $2K - \sigma$ inte schemalagts. Lektionerna i detta klassrum har tillsammans längd K, vilket innebär att motsvarande tal i P har summan K.

Låt oss nu titta på det andra fallet, dvs att $\sigma > 2K$. Vi skapar en lektion för varje tal samt en utfyllnadslektion av längd $\sigma - 2K$. Låt $timelimit = \sigma - K$.

Om det finns en delmängd av P med summan K så kan vi schemalägga motsvarande lektioner samt utfyllnadslektionen i klassrum 1 och övriga lektioner i klassrum 2. Det går bra eftersom lektionerna i klassrum 1 har totala längden $K + \sigma - 2K = \sigma - K$ och återstående lektioner har totala längden $\sigma + \sigma - 2K - (\sigma - K) = \sigma - K$ och vi inte har några andra krav på lektionerna.

Åt andra hållet, om det finns en schemaläggning av lektionerna i två klassrum så måste dom vara helt fyllda upp till $timelimit = \sigma - K$ eftersom totala längden av alla lektioner är $\sigma + \sigma - 2K = 2(\sigma - K)$. Anta att utfyllnadslektionen har schemalagts i klassrum i. Titta på övriga lektioner som schemalagts i det klassrummet. Dessa lektioner har tillsammans längd $\sigma - K - (\sigma - 2K) = K$, vilket innebär att motsvarande tal i P har summan K.

SUBSETSUM(P, K) =
    $\sigma \leftarrow 0$
    $L \leftarrow \emptyset$
    for $p \in P$ **do**
        $\sigma \leftarrow \sigma + p$
        $L \leftarrow L \cup \{$ **new** Lektion$(p)\}$
    if $\sigma \neq 2K$ **then** $L \leftarrow L \cup \{$ **new** Lektion$(|2K - \sigma|)\}$
    if $\sigma \leq 2K$ **then** $timelimit \leftarrow \sigma$
    else $timelimit \leftarrow \sigma - K$
    return EXISTSSCHEDULE$(2, L, timelimit)$

Reduktionen går i polynomisk tid (närmare bestämt linjär tid), så problemet är alltså NP-fullständigt.

3. **Konstruktion av schema**
   Problemet är att hitta det schema som minimerar tidpunkten för den sista schemalagda lektionens slut. Låt S vara summan av längderna för alla lektioner i indata.

   För att hitta detta optimala värde gör vi en binärsökning med *timelimit* mellan 1 och S och anropar ExistsSchedule $\log S$ gånger.

Låt *opttimelimit* vara det optimala värdet.

Nu tar vi reda på vilken lektion som ska schemaläggas i vilket klassrum. Ta en lektion i taget och lägg till ett krav på att lektionen ska schemaläggas i först klassrum 1, sedan i klassrum 2, osv. Prova varje gång med ett anrop till ExistsSchedule och gå vidare till nästa lektion så fort ExistsSchedule svarar ja; behåll då det tillagda kravet.

Detta kräver kn anrop av ExistsSchedule.

Nu ska vi hitta ordningen mellan lektionerna som schemaläggs i samma rum. Gå igenom ett klassrum i taget och gå därefter igenom lektionerna som schemalagts där. Lägg till ett krav på lektionen att den ska schemaläggas efter en annan lektion. Prova varje gång med ett anrop till ExistsSchedule och gå vidare till nästa lektion så fort ExistsSchedule svarar ja; behåll då det tillagda kravet. Efter $O(n^2)$ anrop vet vi exakt vilken ordning lektionerna ska schemaläggas i.

Då återstår bara att hitta starttidpunkterna för varje lektion. Det går att göra genom att vi försöker lägga in extra lektioner som fyller ut alla mellanrum mellan dom ursprungliga lektionerna. Prova att lägga in en lektion av längd 1 före den första lektionen i klassrum 1 (genom att lägga till ett krav). Om ExistsSchedule säger att det går bra, dubblar vi längden och testar igen, osv tills det inte går längre. Då binärsöker vi mellan sista längden som fungerade och den som inte fungerade. När vi hittat den största möjliga extralektionen som ryms före lektion 1, säg att den har längd b, så vet vi att starttiden för lektion 1 är b. Gå sedan på samma sätt vidare med lektion 2. Efter $2n \log opttimelimit$ anrop av ExistsSchedule har vi hittat starttiderna för alla lektioner.

För att konstruera lösningen görs alltså högst $O(\log S + kn + n^2 + n \log S)$ anrop. Tidskomplexiteten blir därför begränsad av $O(\log S + kn + n^2 + n \log S)B(2n, S))$.

Korrektheten för algoritmen inses på följande sätt. Binärsökningen efter optimala värdet är en standardalgoritm som bevisas med en invariant som säger att det sökta värdet finns mellan dom aktuella gränserna i binärsökningen.

Efter binärsökningen vet vi att det finns en lösning med värdet *opttimelimit* men ingen lösning med värdet *opttimelimit* $- 1$. Under resten av algoritmen utökar vi instansen med fler och fler krav på ett sånt sätt att det alltid finns en lösning med värdet *opttimelimit* som uppfyller kraven.

Resonemanget i samband med algoritmbeskrivningen ovan visar att kraven hårdnar alltefter som tills det bara finns en enda möjlig lösning som kan uppfylla kraven, och det är vår sökta schemaläggning.