

Programmering 2: Inlämningsuppgift

Version: 1.0 *

Patrick Wentzel, Józef Swiatycki, Beatrice Åkerblom
jozef@dsv.su.se, beatrice@dsv.su.se

13 mars 2023

Detta dokument beskriver den obligatoriska inlämningsuppgiften på kursen PROG2 för kursomgången VT2023. Uppgiften är giltig under kursomgången fram till uppsamlingstillfället i augusti 2023. Uppgiften upphör därefter att gälla och kan komma att ersättas kursomgången 2024.

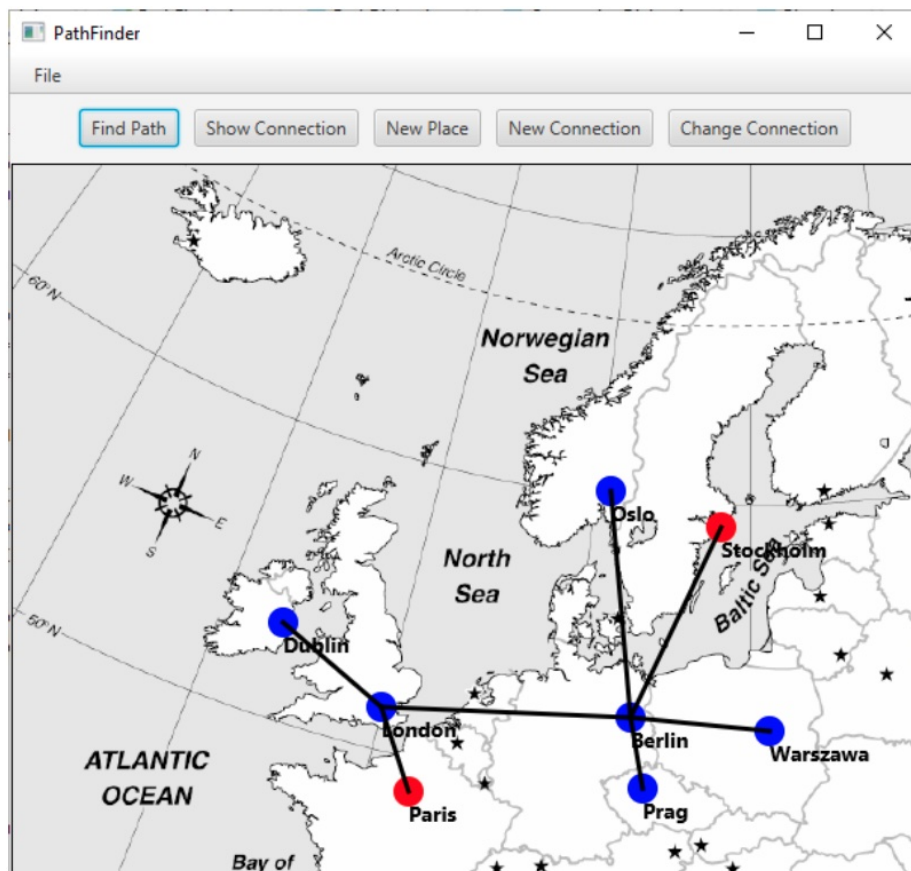
Uppgiften består av två delar: ett mindre graf-bibliotek ¹ (del 1) och ett tillämpningsprogram (del 2).

*Se historiken i tabell 2 på sista sidan i dokumentet för information om eventuella ändringar.

¹Grafer går igenom på föreläsning nummer 9 och 10 (F9 och F10 i schemat.)

Innehåll

1	Inledning	3
2	Inlämningsdatum (deadlines)	4
3	Del 1: Datastrukturen ListGraph	4
3.1	Klassen ListGraph	4
3.2	Klassen Edge	6
4	Del 2: Ett tillämpningsprogram	7
4.1	Menyn File	7
4.1.1	Menyvalet New Map	7
4.1.2	Menyvalet Open	8
4.1.3	Menyvalet Save	9
4.1.4	Menyvalet Save Image	10
4.1.5	Menyvalet Exit	10
4.2	Knapparna	10
4.2.1	Knappen New Place	11
4.2.2	Platser	11
4.2.3	Knappen New Connection	12
4.2.4	Knappen Show Connection	13
4.2.5	Knappen Change Connection	14
4.2.6	Knappen Find Path	15
5	Krav för inlämningen	16
5.1	Gemensamt för båda delarna	16
5.2	Specifikt för del 1	16
5.3	Specifikt för del 2	16
6	Tips	17
7	Dokumenthistorik	18



Figur 1: Ett exempel på hur programmet kan se ut under körning.

1 Inledning

Uppgiften går ut på att skriva ett program som låter användaren öppna en karta (en bakgrundsbild), och med musklickningar på kartan definiera platser på kartan samt lägga till förbindelser mellan dessa platser. Programmet kan sedan användas för att hitta en väg mellan två av användaren valda platser.

Alla platser har ett namn (t.ex. "Kista", "New York", "Andromedagalaxen" o.s.v.). Platserna kan vara förbundna med varandra. Alla förbindelser har ett namn (t.ex. "Buss 514", "Flyg", "Teleportering" o.s.v.) samt en tid det tar att färdas genom denna förbindelse. Förbindelsens namn och tid behöver dock inte skrivas ut på kartan.

Användaren kan definiera nya platser på kartan genom att klicka på knappen "New place" och sedan markera med en musklickning var på kartan platsen skall finnas (se detaljer längre ner). Användaren kan också välja en existerande plats genom att klicka på den, platsen skall då byta färg.

Användaren kan välja två platser genom att klicka på dem på kartan. När två platser är valda kan användaren definiera en förbindelse mellan dessa platser,

visa information om förbindelsen, ändra tiden för förbindelsen eller få en utskrift av vägen mellan dessa två platser.

Inlämningsuppgiften är uppdelad i två delar:

- en datastruktur (graf) för att representera nätverket av platser och förbindelser (denna del innehåller inga grafiska komponenter)
- ett tillämpningsprogram som tar hand om grafiken och om kommunikationen med användaren

2 Inlämningsdatum (deadlines)

För del 1 är det rekommenderade inlämningsdatumet den 3/5, vilket betyder att vi rekommenderar att del 1 är färdig vid detta datum för att det skall finnas tillräckligt med tid för att slutföra del 2. Detta är alltså ingen hård deadline utan del 1 måste senast lämnas in tillsammans med del 2.

För del 2 är sista inlämningsdatum den 4/6.

Uppsamlingstillfälle för inlämning av inlämningsuppgiften är den 13/8, dagen efter omtentan i augusti.

3 Del 1: Datastrukturen ListGraph

För att representera nätverket av platser och förbindelser mellan dem behövs en datastruktur, en graf. Vi kan anta att platserna har ett fåtal förbindelser med andra platser, varför en lösning med kopplingslistor är att föredra framför en lösning med en kopplingsmatris. Det finns inga klasser för graf-representation i Javas standardbibliotek, så i uppgiften ingår att skriva en klass `ListGraph` och en klass `Edge` för representation av enkla² viktade oriktade grafer implementerade med kopplingslistor (*simple weighted undirected graphs implemented by adjacency lists*). Dessa klasser skall implementeras för att vara generella, så att de skall kunna användas av andra, för oss okända, tillämpningar.

Klasserna `ListGraph` och `Edge` skall vara generiska, med nod-typen som generisk parameter. Funktionaliteten i `ListGraph` är deklarerad i ett gränssnitt (*interface*) `Graph`.³

3.1 Klassen ListGraph

Klassen `ListGraph` skall innehålla följande metoder:

- `add` – tar emot en nod och stoppar in den i grafen. Om noden redan finns i grafen görs ingen förändring.

²Påminnelse: enkla grafer är grafer utan parallella kanter eller kanter till start-noden själv.

³Det kan vara klokt att vänta med att låta `ListGraph` implementera gränssnittet `Graph` till dess att du implementerat alla metoder i gränssnittet `ListGraph`. Börja med att implementera och testa `ListGraph`-metoderna var för sig.

- **remove**⁴ – tar emot en nod och tar bort den från grafen. Om noden som skall tas bort saknas i grafen skall undantaget `NoSuchElementException` från paketet `java.util` genereras. När en nod tas bort skall även dess kanter tas bort, och eftersom grafen är oriktad skall dessa kanter även tas bort från de andra berörda noderna.
- **connect** – tar emot två noder, en sträng (namnet på förbindelsen) och ett heltal (förbindelsens vikt) och kopplar ihop dessa två noder med kanter med det angivna namnet och den angivna vikten. Om någon av noderna saknas i grafen skall undantaget `NoSuchElementException` från paketet `java.util` genereras. Om den angivna vikten är negativ skall undantaget `IllegalArgumentException` genereras. Om en kant redan finns mellan dessa två noder skall undantaget `IllegalStateException` genereras (det får finnas maximalt en förbindelse mellan två noder).

Observera att grafen skall vara oriktad, d.v.s. en förbindelse skall alltid representeras av två kanter. Till exempel gäller att det för en förbindelse mellan nod A och B alltid finns en kant från nod A till nod B samt en kant från nod B till nod A, där båda kanterna har samma namn och vikt.

- **disconnect**⁵ – tar emot två noder och tar bort kanten som kopplar ihop dessa noder. Om någon av noderna saknas i grafen skall undantaget `NoSuchElementException` från paketet `java.util` genereras. Om det inte finns någon kant mellan dessa två noder skall undantaget `IllegalStateException` genereras. (Här kan säkert metoden `getEdgeBetween` vara till nytta.) Observera att eftersom grafen är oriktad, d.v.s. att en förbindelse alltid representeras av två kanter, så måste kanten tas bort från båda noderna.
- **setConnectionWeight** – tar emot två noder och ett heltal (förbindelsens nya vikt) och uppdaterar vikten hos förbindelserna mellan dessa två noder till det angivna heltalet. Om någon av noderna saknas i grafen eller ingen kant finns mellan dessa två noder skall undantaget `NoSuchElementException` från paketet `java.util` genereras. Om vikten är negativ skall undantaget `IllegalArgumentException` genereras.
- **getNodes** – returnerar en kopia av mängden innehållande alla grafens ingående noder.
- **getEdgesFrom** – tar emot en nod och returnerar en kopia av samlingen av alla kanter som leder från denna nod. Om noden saknas i grafen skall undantaget `NoSuchElementException` genereras.
- **getEdgeBetween** – tar emot två noder och returnerar kanten mellan dessa noder. Om någon av noderna saknas i grafen skall undantaget `NoSuchElementException` genereras. Om det inte finns någon kant mellan noderna returneras `null`.
- **toString** – returnerar en sträng sammansatt av strängar hämtade från alla de ingående nodernas `toString`-metoder och deras kanter `toString`-metoder, gärna med radbrytningar så att man får information om en nod per rad för förbättrad läsbarhet.

⁴ `remove` och `disconnect` används enbart i del 1 och utnyttjas inte i del 2.

⁵Se fotnot nummer 4

- **pathExists** – tar emot två noder och returnerar **true** om det finns en väg genom grafen från den ena noden till den andra (eventuellt över många andra noder), annars returneras **false**. Om någon av noderna inte finns i grafen returneras också **false**. Använder sig av en hjälpmetod för djupetförstsökning genom en graf.
- **getPath** – tar emot två noder och returnerar en lista (`java.util.List`) med kanter som representerar en väg mellan dessa noder genom grafen, eller **null** om det inte finns någon väg mellan dessa två noder. I den enklaste varianten räcker det alltså att metoden returnerar någon väg mellan de två noderna. En frivillig utökning av uppgiften är att implementera en lösning som returnerar den kortaste vägen (i antalet kanter som måste passeras) eller den snabbaste vägen (med hänsyn tagen till kanternas vikt).

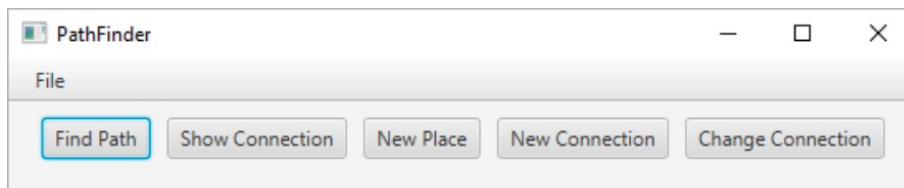
3.2 Klassen Edge

Klassen **Edge** skall representera en kant, alltså den ena riktningen av en förbindelse mellan två noder. Värden för alla instansvariabler i denna klass skall skickas som argument till konstruktorn. Klassen skall innehålla följande metoder:

- **getDestination** – returnerar den nod som kanten pekar till.
- **getWeight** – returnerar den aktuella kantens vikt
- **setWeight** – tar ett heltal och sätter kantens vikt till detta heltal. Om vikten är negativ skall undantaget `IllegalArgumentException` genereras.
- **getName** – returnerar kantens namn
- **toString** – returnerar en sträng med information om kanten

4 Del 2: Ett tillämpningsprogram

Del två av inlämningsuppgiften består av ett tillämpningsprogram⁶. När man först startar programmet skall ett fönster öppnas. Utseendet hos detta fönster visas i figur 2. Fönstret har en meny "File" och fem knappar för de olika operationerna användaren kan använda sig av. "File"-menyn skall innehålla menyvalen "New Map", "Open", "Save", "Save Image" och "Exit".

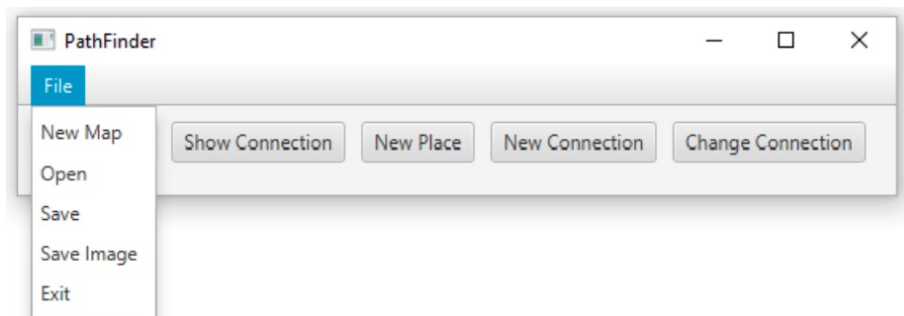


Figur 2: När man först startar programmet öppnas ett fönster med detta utseende.

4.1 Menyn File

I figur 3 visas programfönstret med "File"-menyn nedfälld.

Egentligen borde menyvalen "New Map", "Open" och "Save" öppna en fildialog där användaren kan välja en fil att arbeta med, men det leder till svårigheter vid rättning av inlämningsuppgiften. Av rättningstekniska skäl skall alltså programmet inte fråga användaren om filnamnet utan jobba på hårdkodade filnamn.

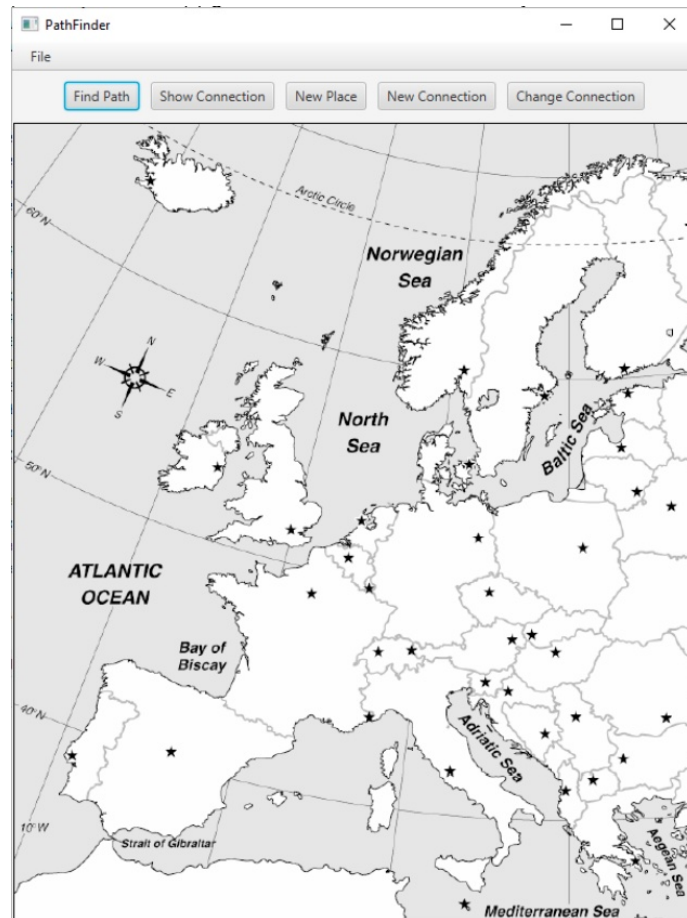


Figur 3: Så här ser programfönstret ut med "File"-menyn nedfälld.

4.1.1 Menyvalet New Map

Om användaren vill börja arbeta med en tom karta skall menyalternativet "New Map" användas. Själva kartan är bara en bakgrundsbild som skall visas i fönstrets centrala area. Bilden ska läsas in från en URL "file:europa.gif" i projekt-mappen. Bildfilen finns i iLearn på samma plats som denna uppgiftstext. Hur fönstret skall se ut visas i figur 4.

⁶I den här delen av inlämningsuppgiften används komponenter och tekniker som går igenom på föreläsningarna F11-F16.



Figur 4: Så här ser programfönstret ut efter att en ny kartbild laddats in med hjälp av menyalternativet "New Map".

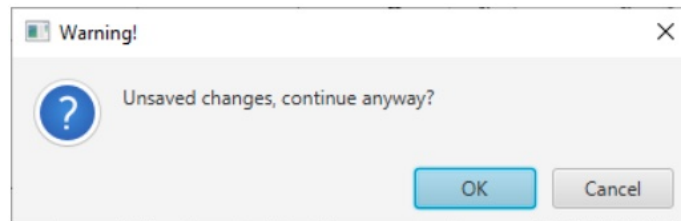
Kartan är som sagt bara en bakgrundsbild. Fönsterstorleken bör anpassas till bilden så att hela kartan visas. Observera att kartan inte bör förändras om användaren förstörar eller förminskar fönstret. Eftersom platser kommer att definieras på fasta koordinater på kartan skulle deras position på kartan förändras om man tillät att kartan skalades om.

4.1.2 Menyvalet Open

Menyvalet "Open" skall öppna filen `europa.graph` som tidigare skall ha skapats genom menyvalet "Save". Om det inte finns någon sådan fil skall ett felmeddelande ges. Annars laddas kartan och objekten (platserna och förbindelserna) in och visas i fönstret. Filen `europa.graph` skall ligga på toppnivå i projekt-mappen, den skall vara en textfil enligt det format som beskrivs under menyvalet "Save". Filen skall innehålla filnamnet på bildfilen som innehåller kartan, information om alla noder (namn och koordinater) och information om alla förbindelser

(från-nod, till-nod, namnet och vikten). Filtypen (ändelsen) för denna fil skall vara ".graph". Med hjälp av informationen i filen skall man kunna återskapa kartan med dess platser och förbindelser mellan platserna.

Observera att om användaren tidigare hämtat en karta eller öppnat en fil och definierat nya platser eller förbindelser, eller ändrat tiden för en förbindelse – kort sagt gjort några förändringar och inte sparat så skall följande dialogfönstret i figur 5 visas.



Figur 5: Dialogfönster som visas om användaren klickar på menyvalet "Open", men tidigare gjort förändringar utan att spara.

Vid svaret "OK" förkastas de osparade ändringarna och det nya "dokumentet" öppnas, vid "Cancel" avbryts operationen.

Ovanstående beteende gäller även vid "New Map", vid "Exit" och när användaren klickar i fönstrets stängningsruta. Programmet **ändrat från "bör": behöver** alltså hålla reda på om det finns osparade ändringar.

4.1.3 Menyvalet Save

Menyvalet "Save" skall spara data om objekten på en textfil med namnet "europa.graph" på följande format:

1. En rad med URL till bildfilen med kartan
2. En (lång) rad med semikolonseparerade uppgifter om alla noder
 - För varje nod skall uppgifter finnas med om nodens namn, nodens x-koordinat och nodens y-koordinat
 - Uppgifter om de olika noderna skall vara separerade med semikolon
3. Flera rader med uppgifter om förbindelserna, en förbindelse per rad
 - Raderna med uppgifter om förbindelserna skall för varje förbindelse innehålla namnet på från-noden, namnet på till-noden, namnet på förbindelsen och förbindelsens vikt.
 - De olika uppgifterna skall vara separerade med semikolon.

Ett exempel på en textfil som motsvarar beskrivningen i listan ovan kan ses i figur 6 (observera att det inte skall finnas några mellanslag efter semikolon).

```
file:europa.gif
Stockholm;469.0;242.0;Oslo;398.0;219.0;Warszawa;503.0;377.0;Berlin;410.0;367.0
Stockholm;Oslo;Train;3
Stockholm;Warszawa;Airplane;2
Oslo;Stockholm;Train;3
Warszawa;Stockholm;Airplane;2
Warszawa;Berlin;Train;6
Berlin;Warszawa;Train;6
```

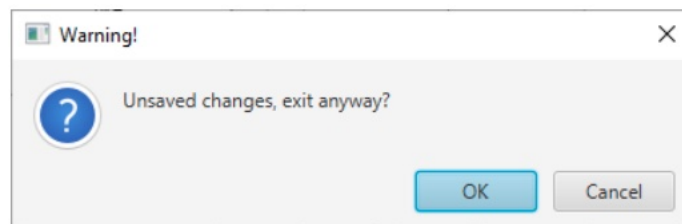
Figur 6: Ett exempel på en textfil som sparats av programmet. Observera att det inte finns några mellanslag efter semikolon.

4.1.4 Menyvalet Save Image

Menyvalet "Save Image" skall spara en ögonblicksbild av fönstrets innehåll på en fil i PNG-format med namnet "capture.png"⁷. Filen skall sparas på toppnivå i projektmappen.

4.1.5 Menyvalet Exit

Menyvalet "Exit" skall avsluta programmet. Programmet skall dock hålla reda på om det finns osparade ändringar och i så fall varna användaren med en dialogruta som kan ses i figur 7.



Figur 7: Denna dialogruta skall visas om användaren klickar på menyvalet "Exit" och det finns osparade ändringar.

Vid svaret "OK" förkastas de osparade ändringarna och programmet avslutas. Vid svaret "Cancel" avbryts operationen.

Programmet skall också kunna avslutas genom att klicka i fönstrets stängningsruta. Detta sätt att stänga programmet skall i övrigt ge samma beteende som menyvalet "Exit".

4.2 Knapparna

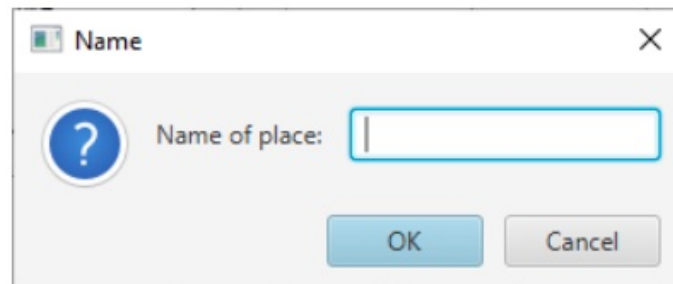
Knapparna i programfönstret skall bara kunna användas efter att en karta har laddats in i programmet.

⁷Koden för hur detta görs visas på en av föreläsningarna F11-F16.

4.2.1 Knappen New Place

Knappen "New Place" skall som namnet antyder användas för att lägga till platser på kartan. Operationen skall dock göras i ett par steg:

1. Användaren klickar på knappen "New Place". När detta sker skall två saker hända:
 - (a) Muspekaren ändras till ett + ("crosshair") i väntan på att användaren klickar på den position på kartan där hen vill ha sin plats.
 - (b) Samtidigt skall även knappen göras inaktiv (göras "disabled").
2. Användaren klickar på kartan för att placera en ny plats:
 - (a) Dialogrutan i figur 8 visas.
3. Användaren matar in platsens namn i textfältet och klickar "OK" i dialogrutan:
 - (a) Den nyskapade platsen dyker upp på kartan.
 - (b) Muspekaren återgår till att vara en vanlig pil.
 - (c) Knappen "New Place" skall aktiveras igen (göras "enabled").



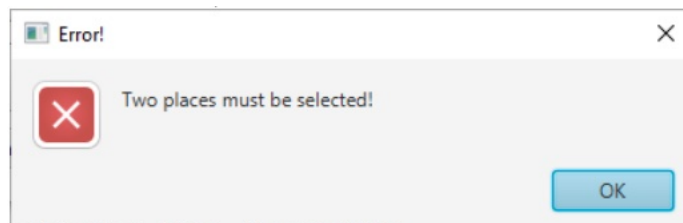
Figur 8: Denna dialogruta skall visas när användaren klickar på kartan för att markera var en ny plats skall placeras.

4.2.2 Platser

Användaren skall kunna markera platser genom att klicka på dem. Platserna skall då byta färg från blå till röd. Om man klickar på platsen en gång till så avmarkeras den (och byter färg till blå igen). Det skall aldrig gå att markera fler än två platser åt gången, försöker man markera en tredje plats så skall ingenting hända.

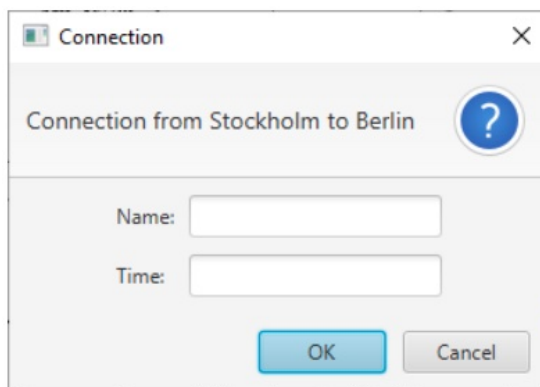
4.2.3 Knappen New Connection

Knappen "New Connection" skall användas för att skapa förbindelser mellan platser. Två platser måste vara valda, annars skall felmeddelandet i figur 9 visas. Eftersom det bara kan finnas en förbindelse mellan två platser skall det även visas ett lämpligt felmeddelande om det redan finns en förbindelse mellan de markerade platserna.



Figur 9: Om användaren klickar på knappen "New Connection" utan att först markera två platser visas detta felmeddelande.

Om ingen förbindelse finns sedan tidigare mellan de två markerade platserna skall en ny förbindelse skapas. Platsen som markerats först blir från-platsen för denna förbindelse och den andra markerade platsen blir till-platsen i förbindelsen. När knappen "New Connection" klickas på skall ett dialogfönster visas där användaren kan mata in namnet på förbindelsen och hur lång tid (heltal, t.ex. antal timmar) det tar att ta sig igenom förbindelsen. Av dialogen skall framgå mellan vilka platser förbindelsen skapas. Ett exempel på hur detta dialogfönster kan se ut visas i figur 10



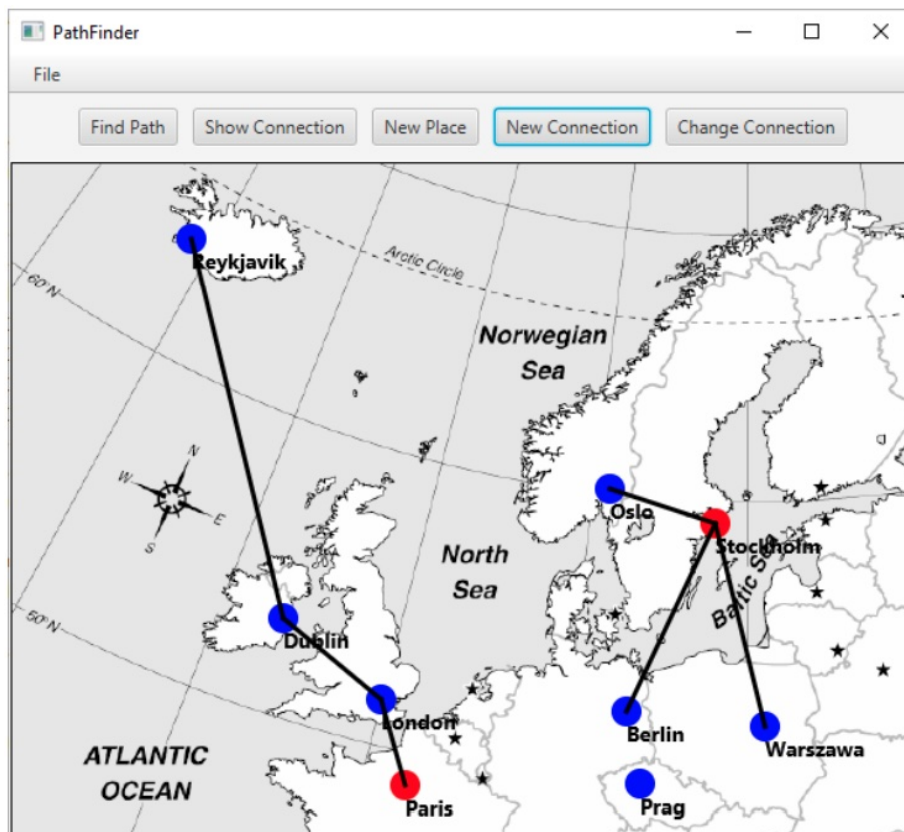
Figur 10: Om användaren klickar på knappen "New Connection" efter att först ha markerat två platser visas detta dialogfönster.

Om användaren trycker på knappen "OK" skall inmatningen kontrolleras. Namnfältet får inte vara tomt och tidfältet måste bestå av siffror. Om inmatningen uppfyller dessa villkor skall förbindelsen skapas och dyka upp på kartan som en linje mellan de två platserna. Om inmatningen inte uppfyller villkoren skall ett

felmeddelande ges och operationen skall avbrytas.

Om användaren trycker på knappen "Cancel" skall operationen avbrytas.

Ett exempel på hur fönstret skulle kunna se ut efter att några platser och förbindelser har lagts till kan ses i figur 11. I detta exempel är Stockholm och Paris valda platser och man kan definiera en förbindelse mellan dem genom att klicka på knappen "New Connection".



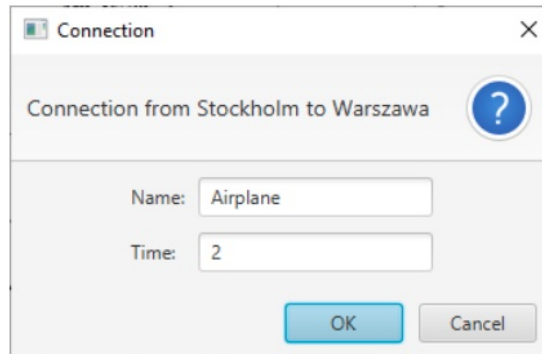
Figur 11: Ett exempel på hur fönstret skulle kunna se ut efter att några platser och förbindelser har lagts till, samt att platserna "Paris" och "Stockholm" markerats.

4.2.4 Knappen Show Connection

Knappen "Show Connection" visar uppgifter om förbindelsen mellan två valda platser. Om det inte finns två valda platser i kartan visas ett felmeddelande (se figur 9). Även om det inte finns någon förbindelse mellan platserna visas ett felmeddelande.

Om två platser som har en förbindelse är markerade visas ett fönster med uppgifter om förbindelsen. Ett exempel på hur ett sådant fönster kan se ut visas i figur 12. Uppgifterna för förbindelsens namn och tiden det tar att färdas genom den skall inte gå att redigera i detta fönster.

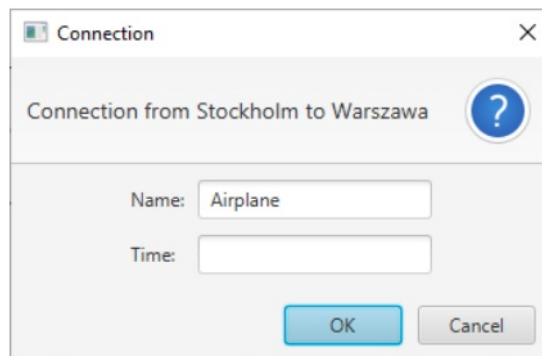
Om användaren klickar på "OK" eller "Cancel" i detta fönster stängs det.

A screenshot of a Windows-style dialog box titled "Connection". The title bar includes a standard icon, the text "Connection", and a close button (X). The main content area has a header section with the text "Connection from Stockholm to Warszawa" and a blue circular help icon with a question mark. Below this, there are two text input fields: "Name:" containing the text "Airplane" and "Time:" containing the number "2". At the bottom right, there are two buttons: "OK" (highlighted in blue) and "Cancel" (disabled, in grey).

Figur 12: Om användaren klickar på knappen "Show Connection" efter att först ha markerat två platser som har en förbindelse visas detta fönster. Observera att textfälten "Name" och "Time" inte kan redigeras.

4.2.5 Knappen Change Connection

Knappen "Change Connection" tillåter ändring av tiden för en förbindelse. Om två platser är markerade när användaren klickar på knappen visas ett fönster där namnet på förbindelsen är ifyllt, men uppgifter om tiden saknas. Ett exempel på hur ett sådant fönster kan se ut visas i figur 13. Uppgifterna för förbindelsens namn skall inte gå att redigera i detta fönster.

A screenshot of a Windows-style dialog box titled "Connection". The title bar includes a standard icon, the text "Connection", and a close button (X). The main content area has a header section with the text "Connection from Stockholm to Warszawa" and a blue circular help icon with a question mark. Below this, there are two text input fields: "Name:" containing the text "Airplane" and "Time:" which is empty. At the bottom right, there are two buttons: "OK" (highlighted in blue) and "Cancel" (disabled, in grey).

Figur 13: Om användaren klickar på knappen "Change Connection" efter att först ha markerat två platser som har en förbindelse visas detta fönster. Observera att textfältet "Name" inte kan redigeras.

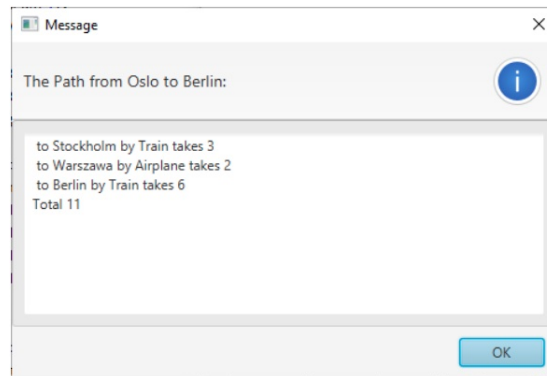
Om användaren klickar på "OK" i detta fönster sparas den nya tiden och fönstret stängs. Tänk på att grafen är oriktad, så ändringar av förbindelsen åt ena hållet måste återspeglas i förbindelsen åt andra hållet.

Om användaren klickar på "Cancel" stängs fönstret utan att några förändringar görs i förbindelsen.

Om två platser inte är valda när användaren klickar på knappen "Change Connection" visas ett felmeddelande (se figur 9). Även om det inte finns någon förbindelse mellan platserna visas ett felmeddelande.

4.2.6 Knappen Find Path

Knappen "Find Path" används för att söka igenom grafen efter en väg mellan två valda platser. Detta sker genom att programmet använder sig av den relevanta metoden i graf-klassen och skriver ut resultatet i en dialogruta. Dialogrutan skall innehålla all relevant information om resan, alltså var man börjar och slutar, vilka platser och förbindelser som passeras, hur lång tid varje delsträcka tar samt hur lång tid resan tar totalt. Ett exempel på en sådan dialogruta visas i figur 14.



Figur 14: Om användaren klickar på knappen "Find Path" efter att först ha markerat två platser som har en förbindelse visas detta fönster.

Om två platser inte är valda när användaren klickar på knappen "Find Path" visas ett felmeddelande (se figur 9). Även om det inte finns någon väg mellan platserna visas ett felmeddelande.

Observera att det inte är bestämt i uppgiften vilken väg som visas av "Find Path". Det kan vara vilken väg som helst, den kortaste vägen eller den snabbaste vägen. Det beror på hur du har implementerat metoden `ListGraph.getPath()` i första delen av inlämningsuppgiften och är alltså valfritt.

5 Krav för inlämningen

För att inlämningarna skall kunna testas (och för del 1 även rättas) automatiskt finns det en del krav som måste vara uppfyllda.

5.1 Gemensamt för båda delarna

Inlämning sker precis som på PROG 1 genom VPL i olika inlämningslådor för de olika uppgifterna.

För att kunna lämna in behöver man ha valt en grupp, och efter att man har lämnat in kan man inte byta grupp (för den inlämningen).

Det som skall lämnas in är enbart Java-källkodsfiler. Inget annat. I vissa delar ställs krav på specifika namn på klasser och filer.

Vid inlämning kommer koden att utvärderas på olika sätt med bl. a. CheckStyle och sedan kommer funktionen testas med enhetstester (JUnit).

Överst i samtliga inlämnade filer skall det finnas en kommentar med namn och användarnamn för samtliga gruppmedlemmar. Exempelvis:

```
// PROG2 VT2023, Inlämningsuppgift, del 1
// Grupp 004
// Dewey Duck dedu1111
// Huey Duck hudu1234
```

Notera att all kod som lämnas in måste vara er egen. Allt fusk rapporteras.

Det ställs krav på kodutseende: koden skall vara korrekt indenterad, namn på klasser, variabler och metoder skall följa Javas namngivningskonventioner, variabler i de klasser som är tänkta som biblioteksklasser skall vara private-deklarerade och metoderna i klasserna skall ha korrekta skyddsnivåer (public, protected eller private).

Varje klass och gränssnitt (interface) skall finnas i en egen fil (utom om det finns ett motiverat behov av inre klasser).

5.2 Specifikt för del 1

För del ett gäller inget särskilt utöver det som nämnts ovan, men det är extremt viktigt att man följer det givna gränssnittet (interface) **Graph** exakt.

5.3 Specifikt för del 2

För del två gäller att man dels måste använda just de namn som finns i beskrivningen ovan för sina menyval och för knapparna. Det skall vara just det namnet som anges i beskrivningarna.

Grafen måste innehålla samma objekt som är klickbara på kartan. Detta beror på att testprogrammet behöver kunna klicka på två noder på kartan och sedan använda de markerade objekten för att se om det finns en kant eller väg mellan platserna genom att undersöka grafen.

Samtliga JavaFX-komponenter, det vill säga knappar samt den komponent som utgör ytan (förmodligen en Pane), skall ha ett ID satt med metoden `setId(String)` enligt uppgifterna i tabell 1.

Komponent	ID
Menyn (en menuBar)	menu
File-menyn	menuFile
Menyvalet New Map	menuNewMap
Menyvalet Open	menuOpenFile
Menyvalet Save	menuSaveFile
Menyvalet Save Image	menuSaveImage
Menyvalet Exit	menuExit
Knappen Find Path	btnFindPath
Knappen Show Connection	btnShowConnection
Knappen New Place	btnNewPlace
Knappen Change Connection	btnChangeConnection
Knappen New Connection	btnNewConnection
Den komponent som platser läggs till på (ritytan)	outputArea
Platser som läggs till på kartan	namnet på platsen

Tabell 1: Komponenter i programmet som skall vara märkta med ett ID.

6 Tips

I samband med testningen har vi uppmärksammat en del ofta förekommande problem:

- Problem: testet avbryts direkt med meddelande om att någon komponents id inte kunde hittas (t.ex. `outputArea`).
 - En tänkbar orsak är att komponenten inte är tillagd i scen-grafen från början utan läggs till först i ett senare steg, t.ex. när ny karta laddas in. Lösningen är att lägga till alla delar redan i start-metoden.
- Problem: en plats blir inte markerad eller avmarkerad som den borde.
 - Det här kan t.ex. orsakas av att någon annan komponent ligger ovanpå platsen och ”stjäl” klicket. Det skulle kunna vara en plats som borde ha varit borttagen i samband med att en ny karta eller ny graf laddats in, eller en text/label eller linje som ligger ovanpå. Allt som läggs till på kartan som inte skall vara klickbart bör inaktiveras (`setDisable(true)`).
 - En annan anledning kan vara att det som håller reda på vad som är markerat kanske inte fungerar helt, eller inte nollställs när ny karta eller ny graf laddas in.

- Problem: filer verkar inte kunna laddas in (antingen karta eller graf)
 - Detta kan orsakas av felaktig sökväg. Klassen `Image` vill ha en URL: `file:europa.gif` men `File` och `FileReader` vill ha ett filnamn: `europa.graph`.
- Problem: när ny graf-fil laddas in blir det fel med kartan.
 - I samband med att graf-filen `europa.graph` laddas in skall bilden med kartan bytas ut till den som anges på första raden i filen. Filnamnet `file:europa.gif` skall inte vara hårdkodat i inläsningen.

7 Dokumenthistorik

Version	Datum	Ändringar
1.0	2023-03-21	Första version

Tabell 2: Versionshistorik för detta dokument.