

UMEÅ UNIVERSITET

December 8, 2014

Institutionen för Tillämpad fysik och elektronik

Laborationsrapport

Laboration 4

Script och webbrogrammering 7,5 hp

Python nr 2

Namn Christer Jakobsson

E-mail dv12cjn@cs.umu.se

Handledare

Ola Ågren, Kalle Prorok

1 Del 1

1.1 Problembeskrivning

Detta python program skall fråga användaren om information om en bok användaren har läst och sedan skriva det till en fil som heter *bibliotek.txt*. Den information som användaren skall ange är:

- Författare
- Titel
- Genre
- Genre2
- Datum när boken lästes
- Betyg i en skala, 1-5
- Egna kommentarer om boken

All information måste anges i rätt format och om användaren har matat in felaktig data till ett fält så upprepas frågan tills rätt format angetts.

1.2 Algoritmbeskrivning

Fråga användaren om all information som ska föras in i filen. Användaren får mata in varje del tills dess att rätt format har angetts.

Format på varje del:

Författare Får ej innehålla några siffror

Titel Får innehålla tecken och siffror, måste vara en sträng.

Genre Genre måste vara en av de tre fördefinierade titlarna som finns.

Genre2 Genre2 måste vara en av de fördefinierade subgenre's av den genre som valdes innan.

Datum läst Måste vara i ett datum format (yy-mm-dd)

Betyg Måste vara en siffra mellan 1-5

Kommentar Får innehålla siffror och tecken, måste vara en sträng

Efter att användaren har matat in rätt format i alla fält så öppnas filen *bibliotek.txt* för skrivning och sedan skrivs en rad med hela bokens information till filen, varje fält separerat med ett kommatecken[,] och de fält som är strängar .

1.3 Exempelkörningar

- Körning 1: Korrekt indata

```
1 lab4$ python del1.py
2 Author: "TestAuthor"
3 Title: "TestTitle"
4 Genre: "Fiction"
5 Genre2: "Horror"
6 Type Date yy-mm-dd: 14-01-01
7 Grade (1-5): 1
8 Comments: "TestComments"
```

- Körning 2: inkorrekt author

```
1 lab4$ python del1.py
2 Author: "test123"
3 Author cant contain digits, try again.
```

- Körning 3: inkorrekt title

```
1 lab4$ python del1.py
2 Author: "Test"
3 Title: asd
4 Title should be text with surrounding "", ex: "Title
   "
5 Title:
```

- Körning 4: inkorrekt genre

```
1 lab4$ python del1.py
2 Author: "Test"
3 Title: "Test"
4 Genre: "Test"
5 Genre must be: ['Fiction', 'Fact book', 'Poetry']
```

- Körning 5: inkorrekt genre2

```
1 lab4$ python del1.py
2 Author: "Test"
3 Title: "Test"
4 Genre: "Test"
5 Genre must be: ['Fiction', 'Fact book', 'Poetry']
6 Genre: "Fiction"
7 Genre2: "Test"
8 Genre2 must be: ['Detective', 'autobiography', '
   Horror', 'Comedy']
9 Genre2:
```

- Körning 5: inkorrekt datum

```
1 lab4$ python del1.py
2 Author: "Test"
3 Title: "Test"
4 Genre: "Fiction"
5 Genre2: "Horro"
6 Genre2 must be: ['Detective', 'autobiograph', '
    Horror', 'Comedy']
7 Genre2: "Horror"
8 Type Date yy-mm-dd: 14 01 01
9 Wrong format, try again!
10 Type Date yy-mm-dd: 2014-01-01
11 Wrong format, try again!
12 Type Date yy-mm-dd:
```

- Körning 5: inkorrekt datum

```
1 Grade (1-5): 0
2 Grade needs to be in range 1-5
3 Grade (1-5): 6
4 Grade needs to be in range 1-5
5 Grade (1-5):
```

- Körning 5: inkorrekt comments

```
1 Comments: test
2 Comments should be text with surrounding "", ex: "
    Comments".
```

2 Del 2

2.1 Problembeskrivning

Detta programs uppgift är att fråga användaren om en boktitel och sedan söka efter denna titel i *bibliotek.txt* och om den hittas skriva ut den raden. Programmet kan ta en titel när det har startats, eller ta flera titlar att söka efter som argument.

2.2 Algoritmbeskrivning

Programmet frågar efter en boks titel och när denna har angetts så öppnas filen *bibliotek.txt* för läsning. Sedan så läses datat in ifrån filen rad för rad och för varje rad så kollar programmet om titeln som eftersöktes stämmer med titeln på raden, om så är fallet så skrivs raden ut på stdout. För att skriva ut alla poster i filen så skriver man "ALL".

2.3 Exempelkörningar

- Körning 1: Korrekt title

```
1 lab4$ python del2.py
2 Title To Search For (For all, type "ALL"): "
  TestTitle"
3 "TestAuthor", "TestTitle", "Fiction", "Horror",
  2014-01-01, 1, "TestComments"
```

- Körning 2: input som ej är en sträng

```
1 lab4$ python del2.py
2 Title To Search For (For all, type "ALL"): ejstrang
3 Error: input needs to be text with surrounding "",
  ex: "ALL".
4 Title To Search For (For all, type "ALL"):
```

- Körning 3: Sökt titel som inte finns

```
1 lab4$ python del2.py
2 Title To Search For (For all, type "ALL"): "HEJ"
3 Title HEJ not found.
```

Kommentar: Om programmet inte hittar den titel det söker efter så skriver det ut det å på standard error för att möjligheten att pipa eller omdirigera output till en fil skall vara möjlig.

- Körning 4: Titlar som argument

```
1 lab4$ python del2.py "asd" "TestTitle"
2 "asd", "asd", "Fact book", "Chemistry", 2014-01-01,
  1, "No comments"
3 "TestAuthor", "TestTitle", "Fiction", "Horror",
  2014-01-01, 1, "TestComments"
```

3 Del 3

3.1 Problembeskrivning

Detta program skall implementera ett grafiskt gränssnitt som ska ha funktionaliteten ifrån de två programmen som beskrivits i Del 1 och Del 2. Det ska alltså både innehålla en del där användaren kan lägga till en bok i filen och kan söka efter en boks titel och få reda på all information om den titeln.

3.2 Algoritmbeskrivning

Detta program ska binda samman de två tidigare delarna och implementera dessa i ett grafiskt gränssnitt. Därmed har den underliggande koden i princip samma algoritmbeskrivning som föregående delar förutom att det är kopplat till ett grafiskt gränssnitt som ska vara så tydligt, enkelt och intuitivt att använda som möjligt.

3.3 Exempelkörningar

- Körning 1: Add Data

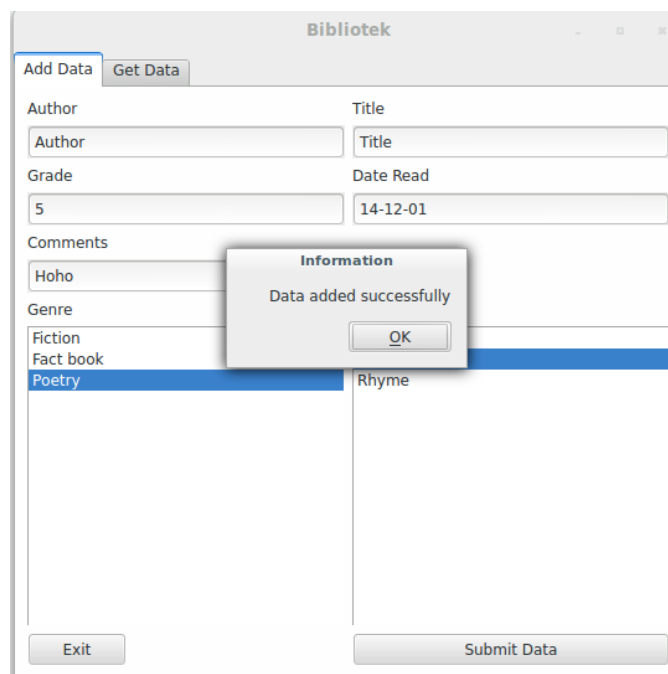


Figure 1: Add Data

Kommentar: Har matat in korrekta värden i alla fält och sedan tryckt på *Submit Data*.

- Körning 2: Fält ej ifyllt korrekt: Exempel där datum inte är ifyllt korrekt

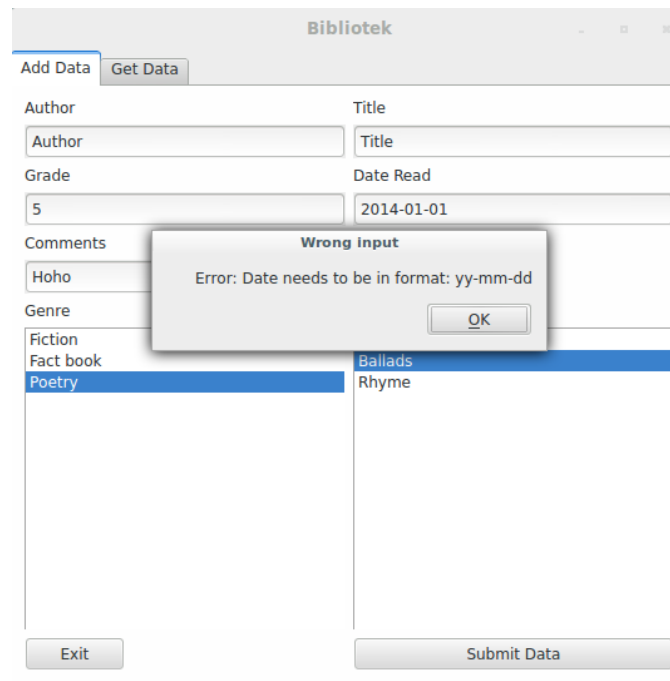
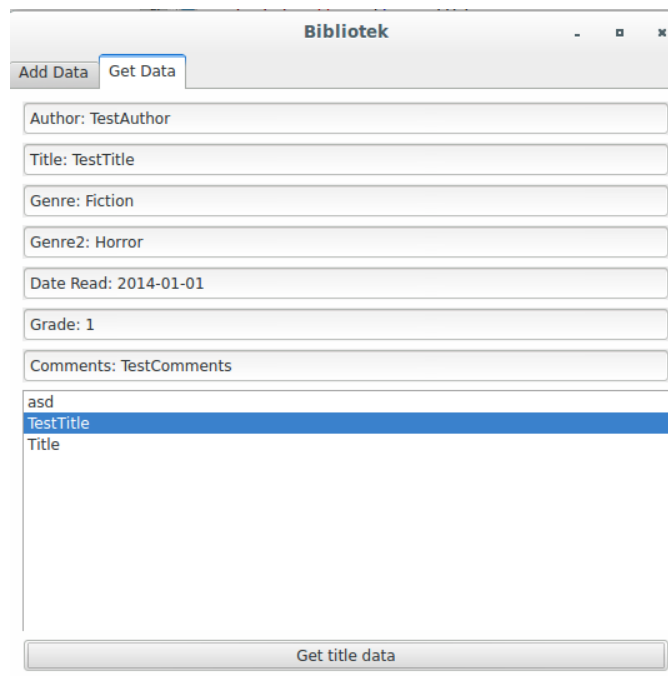


Figure 2: Fel input

Kommentar: Har matat in korrekta värden i alla fält förutom i datumfältet och sedan tryckt på *Submit Data*, datat läggs inte in och användaren får ett felmeddelande.

- Körning 3: Get data



The screenshot shows a window titled "Bibliotek" with two tabs: "Add Data" and "Get Data". The "Get Data" tab is active. It contains several input fields with the following values: "Author: TestAuthor", "Title: TestTitle", "Genre: Fiction", "Genre2: Horror", "Date Read: 2014-01-01", "Grade: 1", and "Comments: TestComments". Below these fields is a list box containing three items: "asd", "TestTitle" (which is highlighted with a blue background), and "Title". At the bottom of the window is a button labeled "Get title data".

Figure 3: Get Data

Kommentar: Har valt titeln *TestTitle* och sedan klickat på *Get title data*, Datat visas i fälten ovanför listan.

4 Del 4

4.1 Problembeskrivning

Denna del skall innehålla två stycken program, ett program som ska vara en klient med ett gui som ser ut och har samma funktionalitet som föregående del. Men som inte ska skriva och läsa direkt ifrån filen *bibliotek.txt* utan ska kommunicera med den andra delen som skall vara en server som skall sköta all skrivning och läsning ifrån filen. Kommunikationen ska ske genom sockets så att klienten och servern inte behöver ligga på samma dator.

4.2 Algoritmbeskrivning

När klienten skickar till server så skickar den ett kommando följt av ett mellanslag sedan eventuell data som server behöver, Ex *[GET_ ALL_ DATA]* de kommandon som klienten och server använder sig av är:

- GET_ALL_DATA

Klienten skickar detta meddelande till servern när den vill få ut information om alla böcker som finns i filen.

- GET_TITLE_DATA *Boktitel*

Detta meddelande skickar klienten när den vill få all information om en specifik titel ifrån servern.

- SEND_ROW_DATA *Bokdata*

Används när klienten vill lägga in en ny bok i biblioteket.

4.2.1 Klient

För att användaren ska kunna lägga till eller hämta data ifrån servern måste användaren ansluta till servern. Därefter så kan användaren lägga till böcker via *Add tab* eller hämta en boks data via *Get data* tabben.

Add Data:

- Användaren får skriva in värden i varje fält, Author, Title etc
- Användaren trycker på *Submit Data*, om alla fält har korrekt input
Öppna filen för skrivning och skriv raden med bokens information till slutet på filen
Stäng filen, En popup som meddelar användaren att det gick bra visas.
- Om något fält inte är korrekt ifyllt kommer en popup visas och meddela användaren vilket fält som är fel.

Get data

- Användaren får klicka på den boktitel den vill ha all information ifrån.
- När användaren har valt en titel och klickar på *Get title data*
Filen öppnas för läsning, den eftersökta titelns bokrad söks efter
När boktiteln hittas så fylls fälten i med information från bokens alla poster.
- Filen stängs och användaren ser bokens information i textfälten.

4.2.2 Server

Add data:

- Servern får kommandot [SEND_ROW_DATA *Bokdata*]
- Plockar ut *Bokdata* och lagrar det i en datastruktur
- Skriver datat till filen, stänger filen och skickar ett ok till klienten att det gick bra.

Get all data:

- Servern får ett kommando [GET_ALL_DATA] ifrån klienten
- Öppnar filen för läsning, läser in alla poster ifrån filen och skickar det till klienten.

Get row data:

- Servern får ett kommando [GET_ROW_DATA *Boktitel*]
- Öppnar filen för läsning
- Söker efter raden med den titeln
- Stänger filen och returnerar titels data till klienten

4.3 Exempelkörningar

Exempelkörningarna kommer att se likadana ut i denna del som i Del 3, i och med att denna dels klient ser likadan ut som i del 3. Servern har inget grafiskt gränssnitt men den har utskrifter när den tar emot ett kommando och när den skickar data till klienten.

- Körning 1: Ansluta till servern.

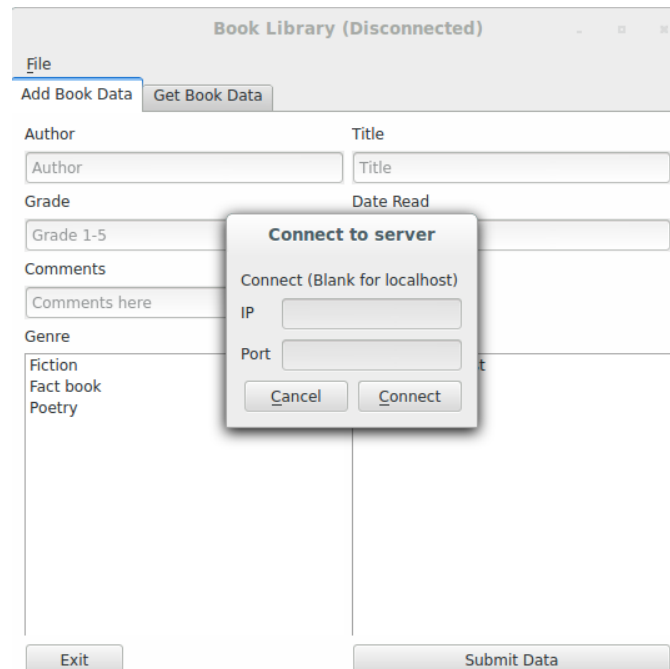


Figure 4: Anslutning till server

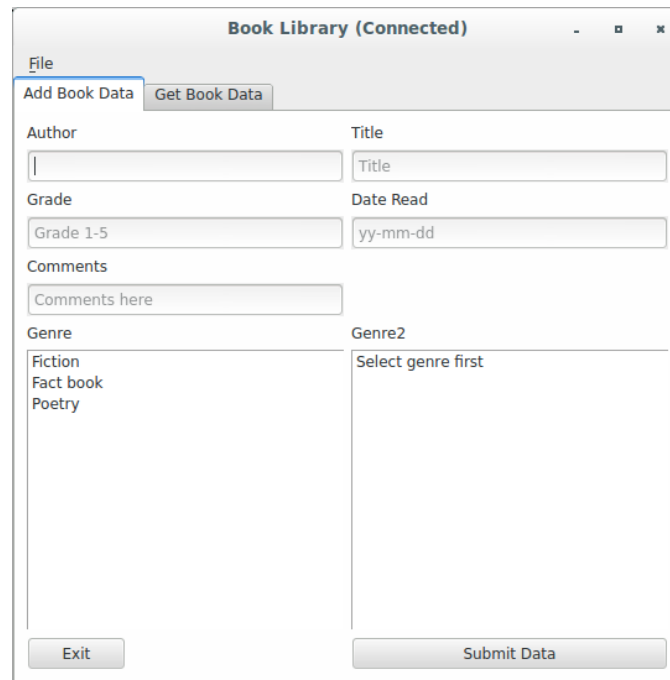


Figure 5: Anslutning lyckats

```

1 lab4$ python server.py
2 Socket now listening (Exit server with CTRL+C)
3 Server: got connection from client 127.0.0.1:34447

```

Kommentar: Den första bilden 4 visar hur programmet ser ut vid start, användaren upmanas att skriva in en server att ansluta till, eller lämna blank om server ligger på den lokala datorn. Bild ?? visar hur programmet ser ut vid en lyckad anslutning, notera (Connected) uppe i fönstrets list. Koden visar hur serversidan ser ut när användaren lyckats anslutit.

- Körning 2: Add Data

```

1 [SEND\_ROW\_DATA]
2 127.0.0.1:34447 sends SEND\_ROW\_DATA
3 Sending 200 OK
4 Server: got connection from client 127.0.0.1:34447

```

- Körning 3: Refresh List

```

1 [GET\_ALL\_DATA]
2 127.0.0.1:34447 sends GET\_ALL\_DATA
3 sending reply
4 Server: got connection from client 127.0.0.1:34447

```

Kommentar: I del 3:s lösning så uppdateras listan med bokdata automatiskt när programmet startar, men i en server/klient lösning så kan det vara bra att skicka så lite data emellan servern och klienten

som möjligt. Därför måste klienten klicka på *Refresh List* för att skicka en förfrågan till servern på av få all data.

- Körning 4: Get Row Data

```
1 [GET\_TITLE\_DATA]
2 127.0.0.1:35171 sends GET\_TITLE\_DATA
3 Title [TestTitle]
4 Sending reply
5 Server: got connection from client 127.0.0.1:35171
```

Kommentar: Titlen som efterfrågas av klienten är *TestTitle*