

Centrum För Livslångt Lärande

Rapport om Programmering projekt

SFXIT-Svenska för programmerare

Programmering 2 HT-17

Edo Widiyadi

Innehåll

[Uppgift 1: Vector och MyInteger 2](#_Toc498862534)

[Uppgift 2: MyList, FromFile och ToFile 3](#_Toc498862535)

[Uppgift 3: Person, BioStudent, BioClass och MyStudent 4](#_Toc498862536)

[Uppgift 4: Comparable, Book och MyBooks 8](#_Toc498862537)

[Uppgift 5: leapYear och MyLeapYear 9](#_Toc498862538)

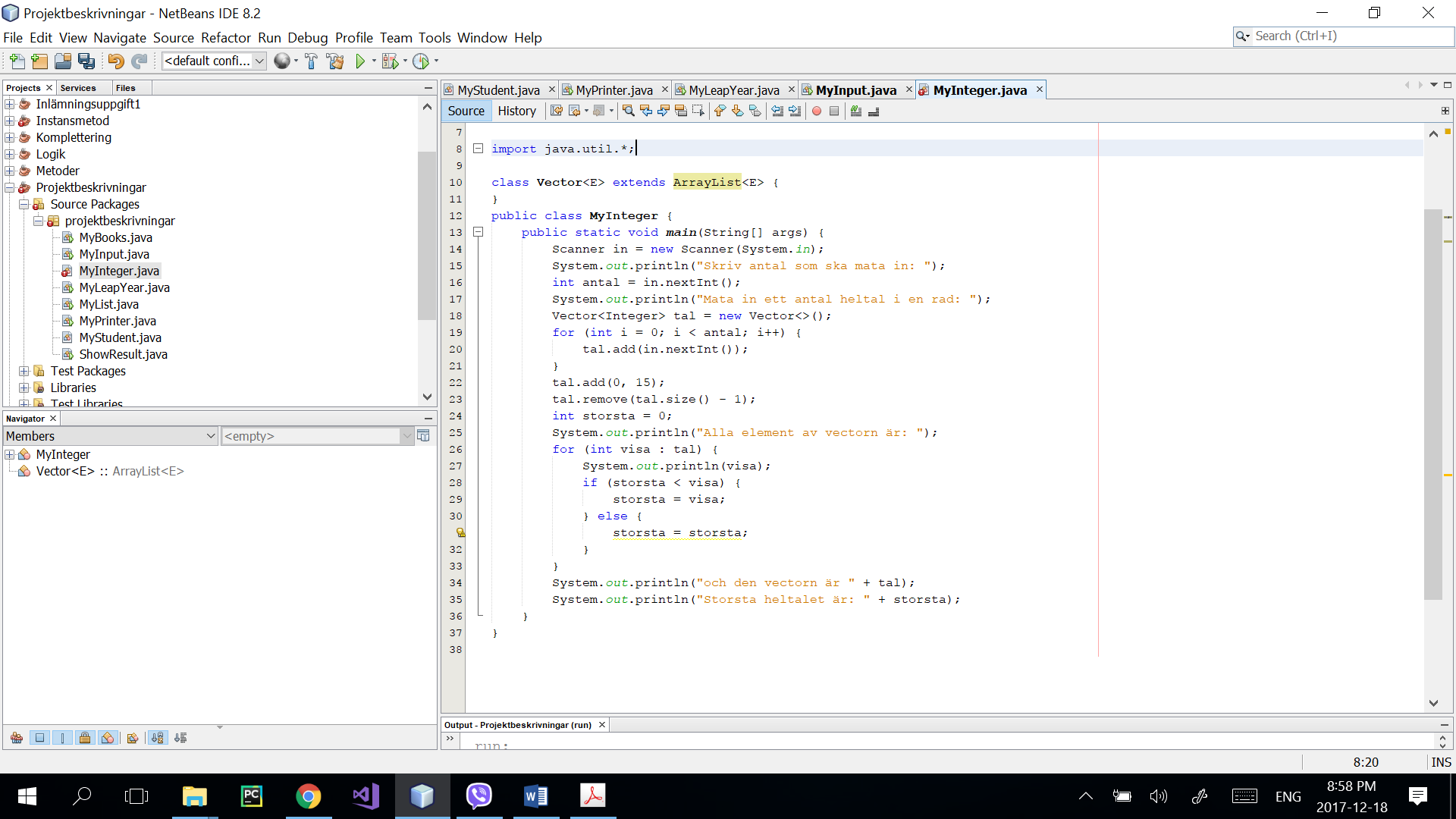
[Uppgift 6: show och MyResult 10](#_Toc498862539)

[Uppgift 7: InputInteger och MyInput 11](#_Toc498862540)

[Uppgift 8: StringArray och MyStringArray 12](#_Toc498862541)

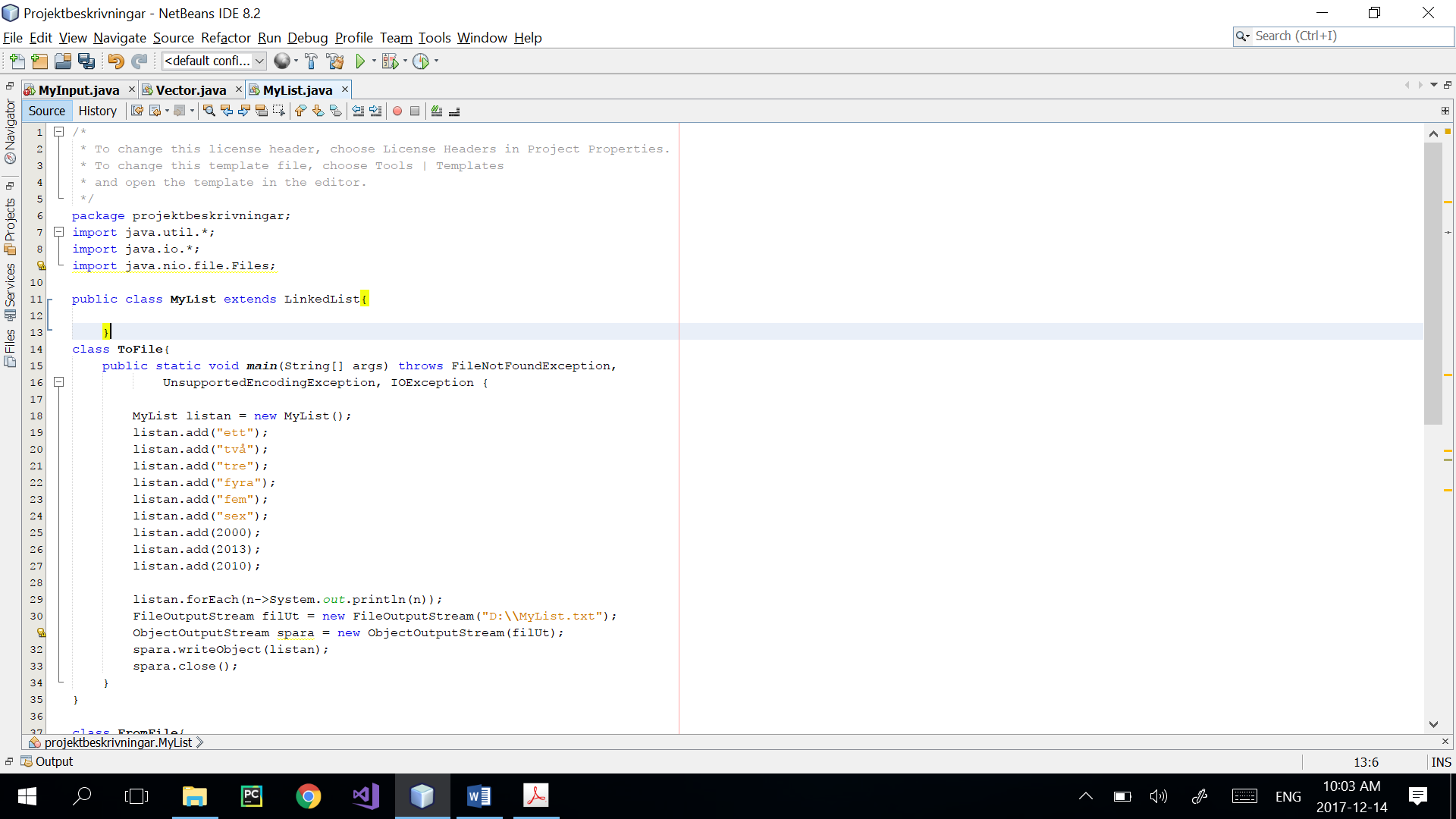
[Uppgift 9: Printer och MyPrinter 12](#_Toc498862542)

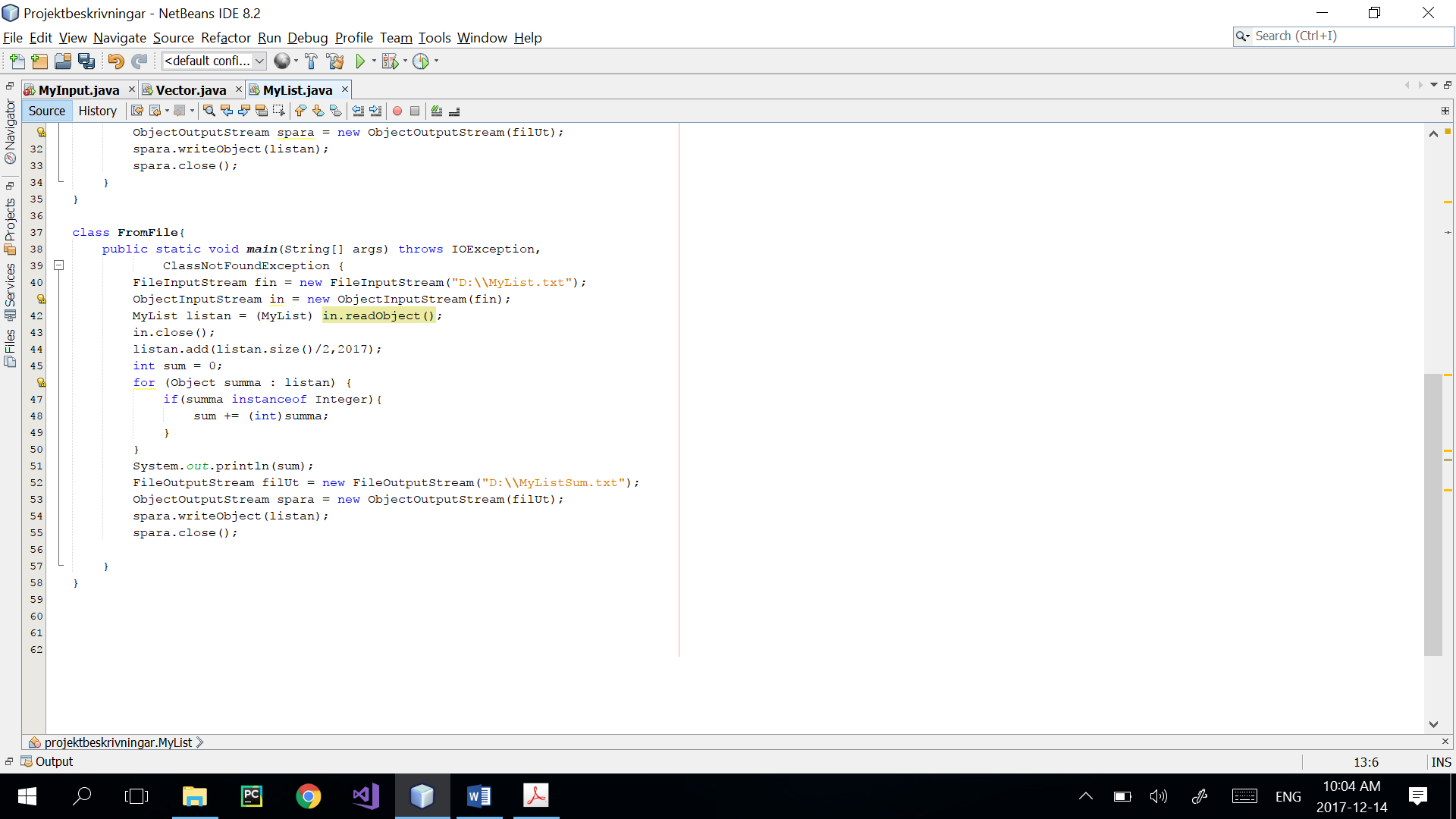
# Uppgift 1: Vector och MyInteger



Klassen Vector är barn klass till klassen vektorn typ av ArrayList och ärver alla metod som finns i klassen ArrayList. Klassen MyInteger skapar en ny typ objekt av Vector klassen som storleken definierar av användare från standardinmatningsenheten och hanterar ett antal heltal som matas in från standardinmatningsenheten till denna Vector. Sedan läggar programmet i ett heltal (15) på första element och ta bort sista element av Vector. Sedan jämför programmet alla element i Vector med loop och if/else påstående för att bestämma vilken elementen som är största och lagras den i en variable. Avslutningsvis visa vektorn och största element i vektorn.

# Uppgift 2: MyList, FromFile och ToFile



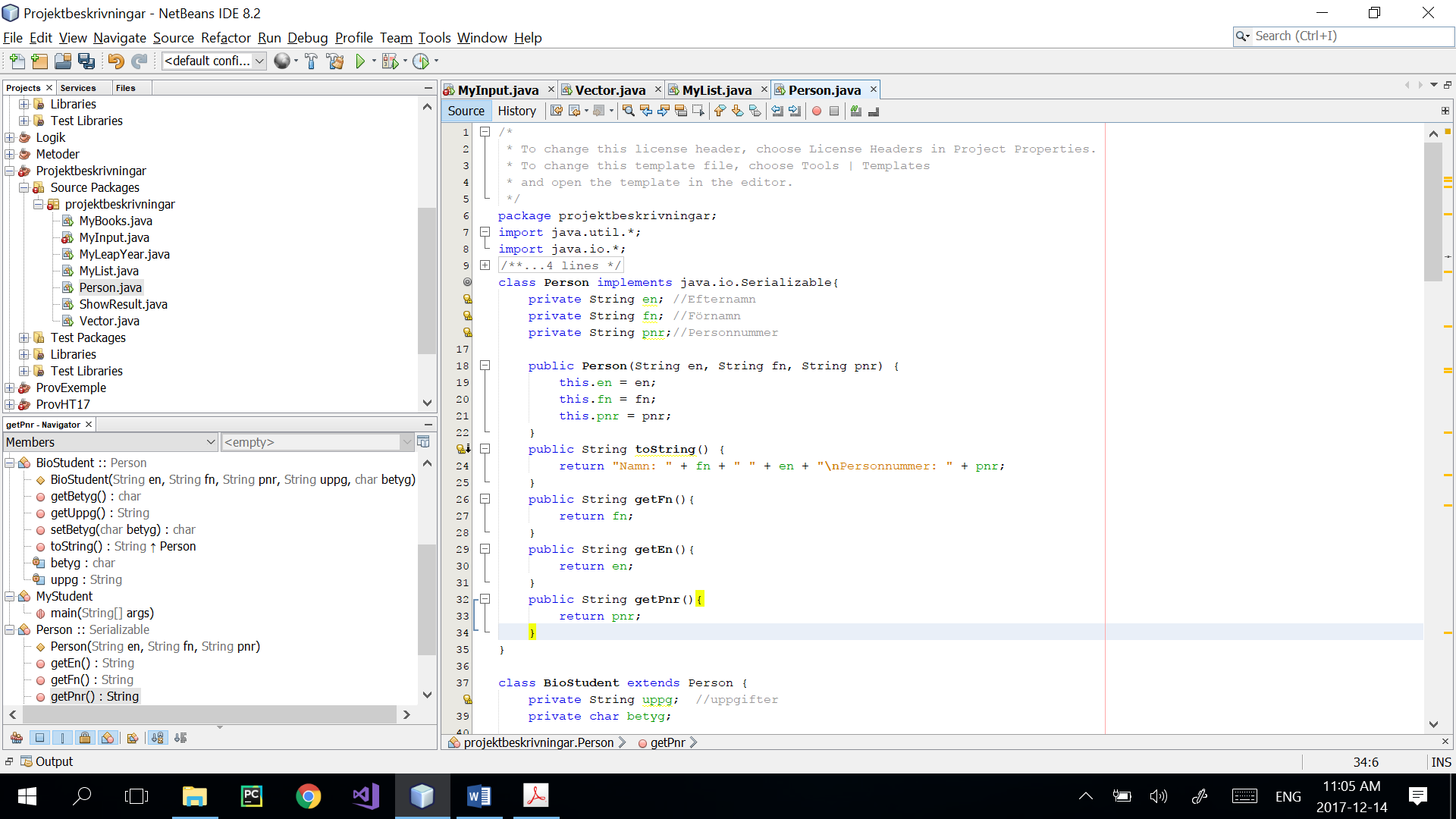


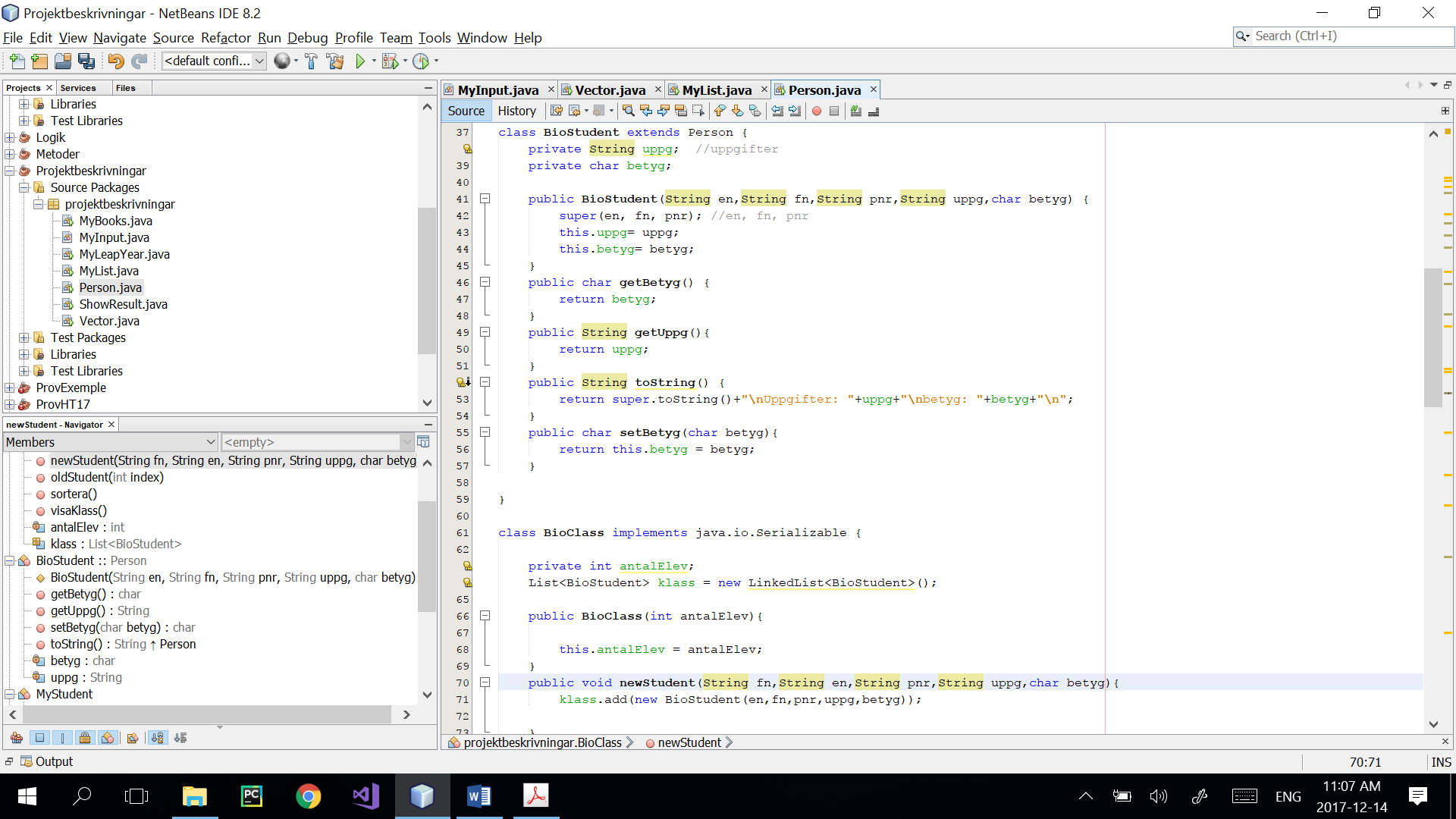
Klassen MyList ärva från klassen LinkedList och fungerar liksom klassen LinkedList.

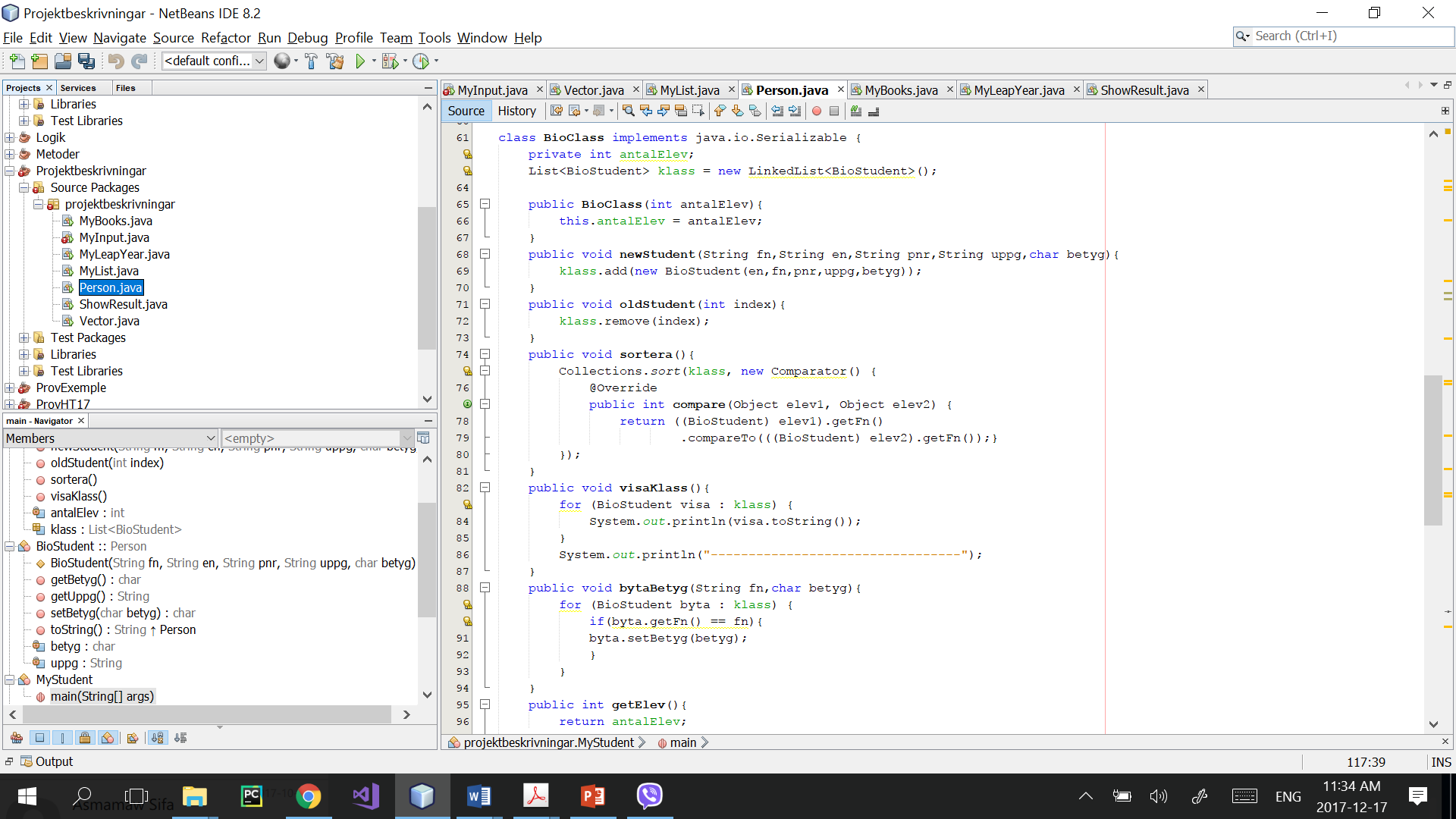
Programmet ToFile skapar ett nytt objekt av MyList som fungerar liksom LinkedList. Sedan matar det in flera heltal och flera strängar och lagrar dem i detta objekt MyList. Sedan visar alla element i objekt. Eftersom objekt MyList är LinkedList som är serial data använd programmet objekt stream för att spara dem i en fil. FileOutputStream skapar en ny fil och ObjectInputStream skapar en ny inmatning till FileOutputStream. ObjectInputStream har en metod att läsa (read() metod) objekt MyList som vill spara i en fil och en metod att sluta läsa (close() metod).

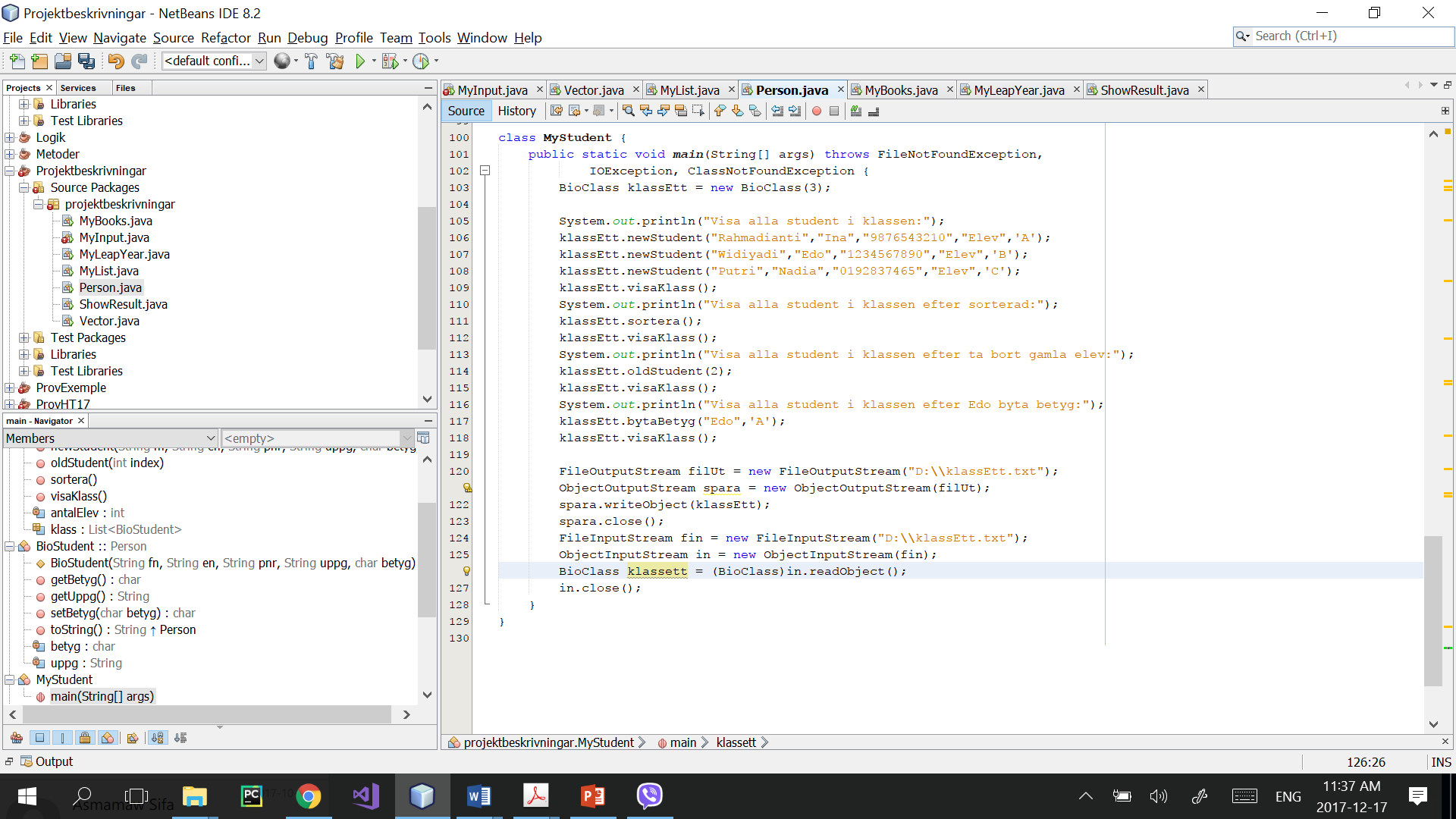
Programmet FromFile läser filer som har redan skapat med objekt stream eftersom data är länk list. FileInputStream skapar ett nytt object som hämta filen, ObjectInputStream skapar ett nytt object som har metod readObject() att läsa filen och när det slutar använder metod close(). Sedan infogar listan med heltalet 2017 i mitten av listan med metod add() som använd parametrar halva av listans storlek (size()). Programmet summar alla heltal som finns i element av listan med loopar alla element i listan och använd if påstående om element är instanceof Integer.

# Uppgift 3: Person, BioStudent, BioClass och MyStudent











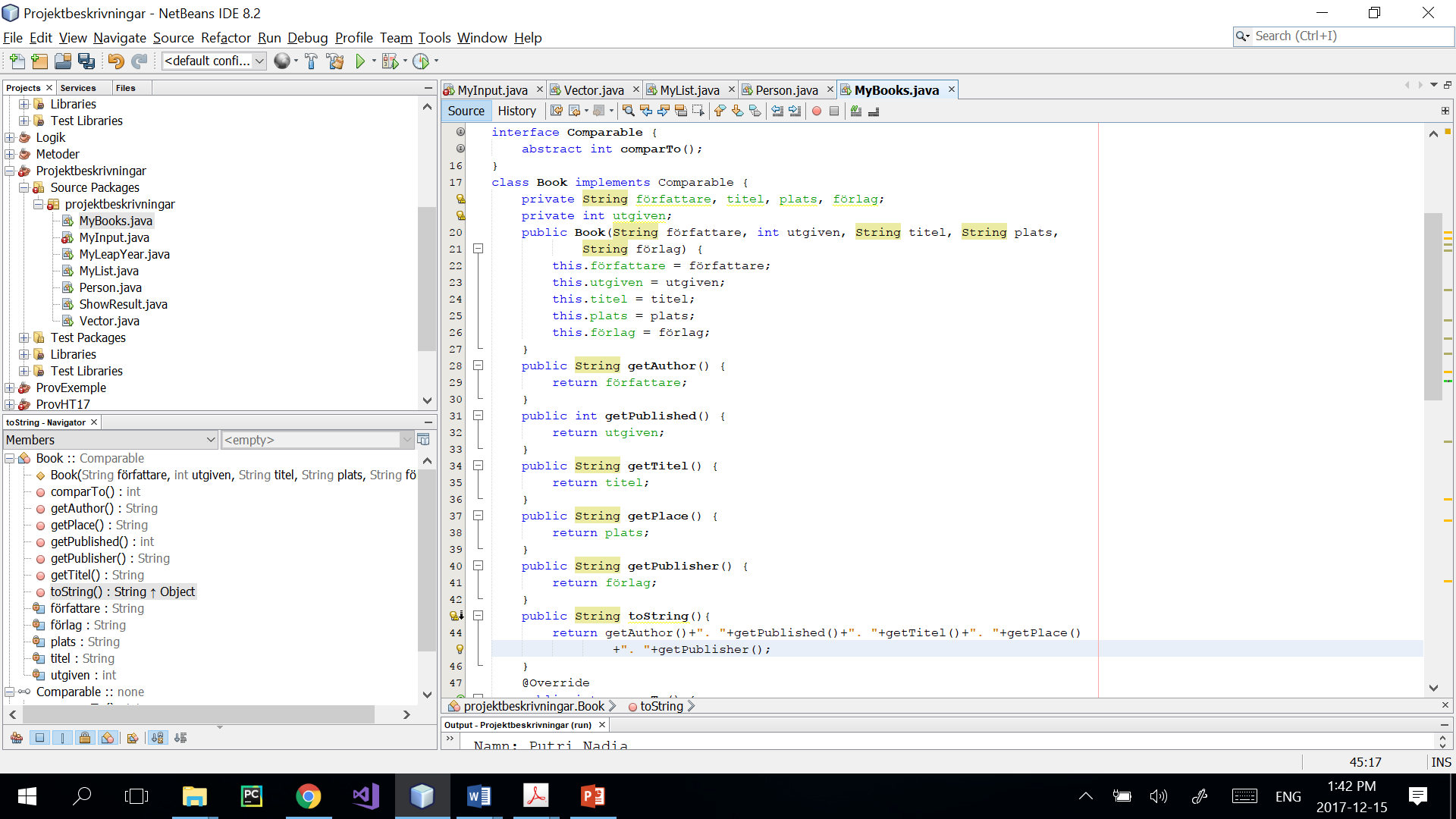
Klassen Person har 3 instans variabler som deklarera efternamn, förnamn och personnummer. Klassen Person har metod utan parametrar get…() i klassen Person finns för att hämta variablerna som deklarerar private och metod toString() för konvertera variablerna till String. Klassen Person implementera Serializable från klass java.util för att göra data till länk/serial som senare behöver ObjectStream länk list för att spara till en ny fil med sin metod.

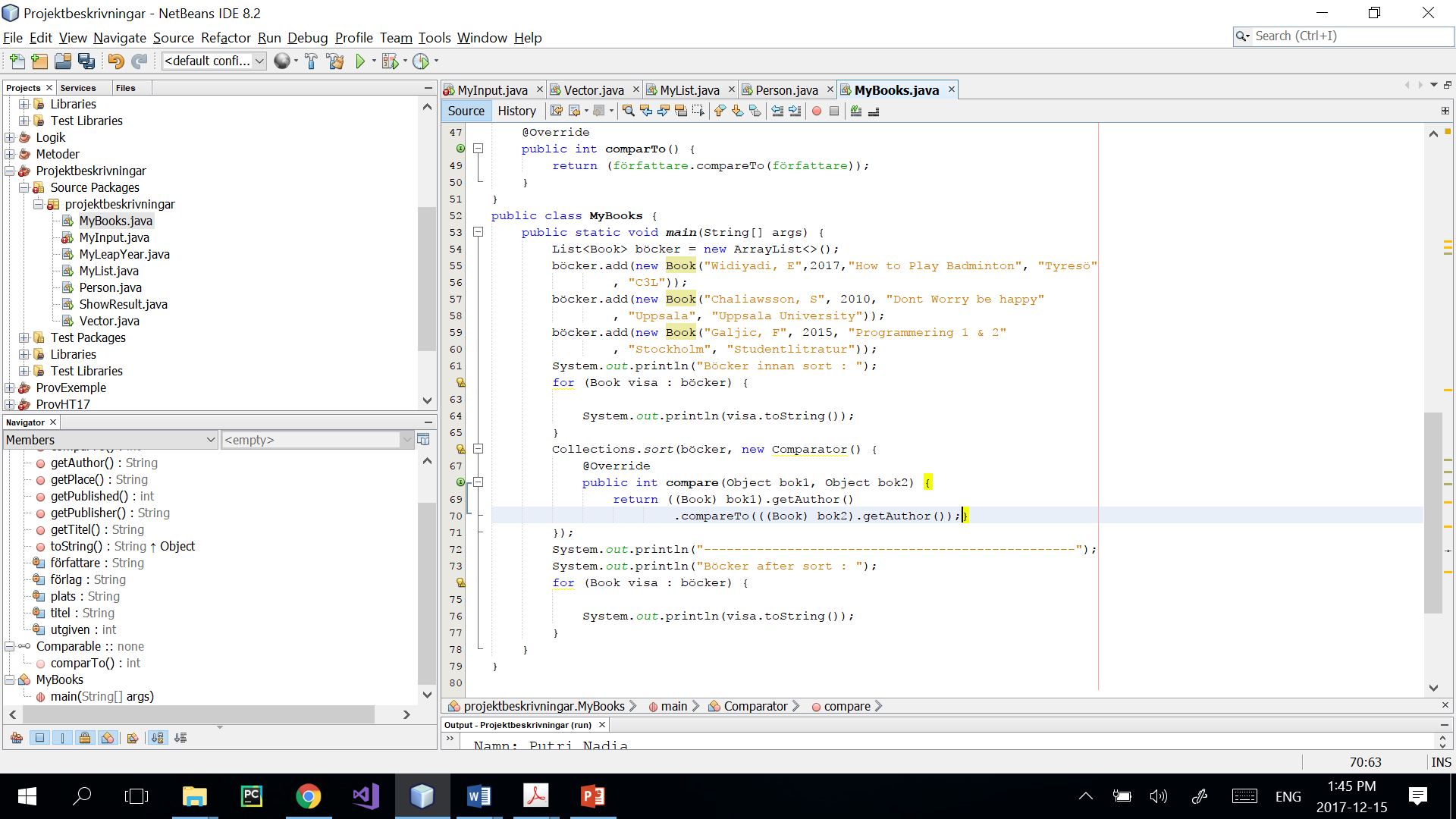
Klassen BioStudent är barnklassen till klassen Person och ärver alla variabler och metoder som deklarerar i klassen Person. Med syntax super har klassen BioStudent tillgång klassen Person instans variabler och metoder. Klassen BioStudent har 2 instans variabler och de är uppgifter and betyg. Klassen BioStudent har också metod utan parametrar get…() i klassen Person finns för att hämta variablerna som deklarerar private och metod toString() för konvertera variablerna till String. Klassen BioStudent har en metod med parametrar setBetyg(char) som kan ändra betyget.

Klassen BioClass implementera också Serializable från klass java.util liksom klassen Person. ObjectStream behöver länk list för att spara till en ny fil. Klassen BioClass deklarerar instans variabel antalElever som beskriver hur många elever av BioStudent som anmäler i BioClass. Denna klass deklarerar också nya länk list med typ av objekt BioStudent som beskriver elever information. Klassen BioClass har metod med parametrar elever information att lägga in nya elever till länk list och metod med parametrar index i listan att ta bort gamla elever från länk list. Klassen BioClass har också metod utan parameter att sortera elever i listan baserad av förnamn, metod utan parameter att visa eller utskriva elever information i listan, metod med parameter att byta elev betyg i listan och metod utan parameter att få antal elever i en klass (storlek av en objekt BioClass).

Applikation MyStudent skapar en lista av typ BioClass med parameter listas storlek så det definierar hur många elever i en klass till exempel skapar klassEtt med 3 elever. Sedan lägga i nya elever som ska skrivas in i en klass med anropa metod newStudent(…) med parametrar elevs information och sedan utskriver alla elever information i denna klass med anropa metod utan parameter visaKlass(). Sedan sorterar eleverna i klassen baserad förnamn med anropa metod utan parameter sortera() och utskriver igen med anropa metod visaKlass(). Nästa steg tas bort gammal elev med anropa metod oldStudent(int) med parameter index i listan som ska tas bort och utskriver igen med anropa metod visaKlass(). Sedan byter en elev betyg med anrop metod bytaBetyg(String, char) med parametrar förnamn elev och betyg som ska bytas på och utskriver igen med anropa metod visaKlass(). Sista steg sparar alla information i lista(klassEtt) i en ny fil med programmet objekt stream. FileOutputStream skapar en ny fil (klassEtt.txt) och ObjectInputStream skapar en ny inmatning till FileOutputStream. ObjectInputStream har en metod att läsa (read() metod) objekt MyList som vill spara i en fil och en metod att sluta läsa (close() metod). Sedan läser filer som har redan skapat med objekt stream eftersom data är länk list. FileInputStream skapar ett nytt object som hämta filen, ObjectInputStream skapar ett nytt object som har metod readObject() att läsa filen och när det slutar använder metod close().

# Uppgift 4: Comparable, Book och MyBooks



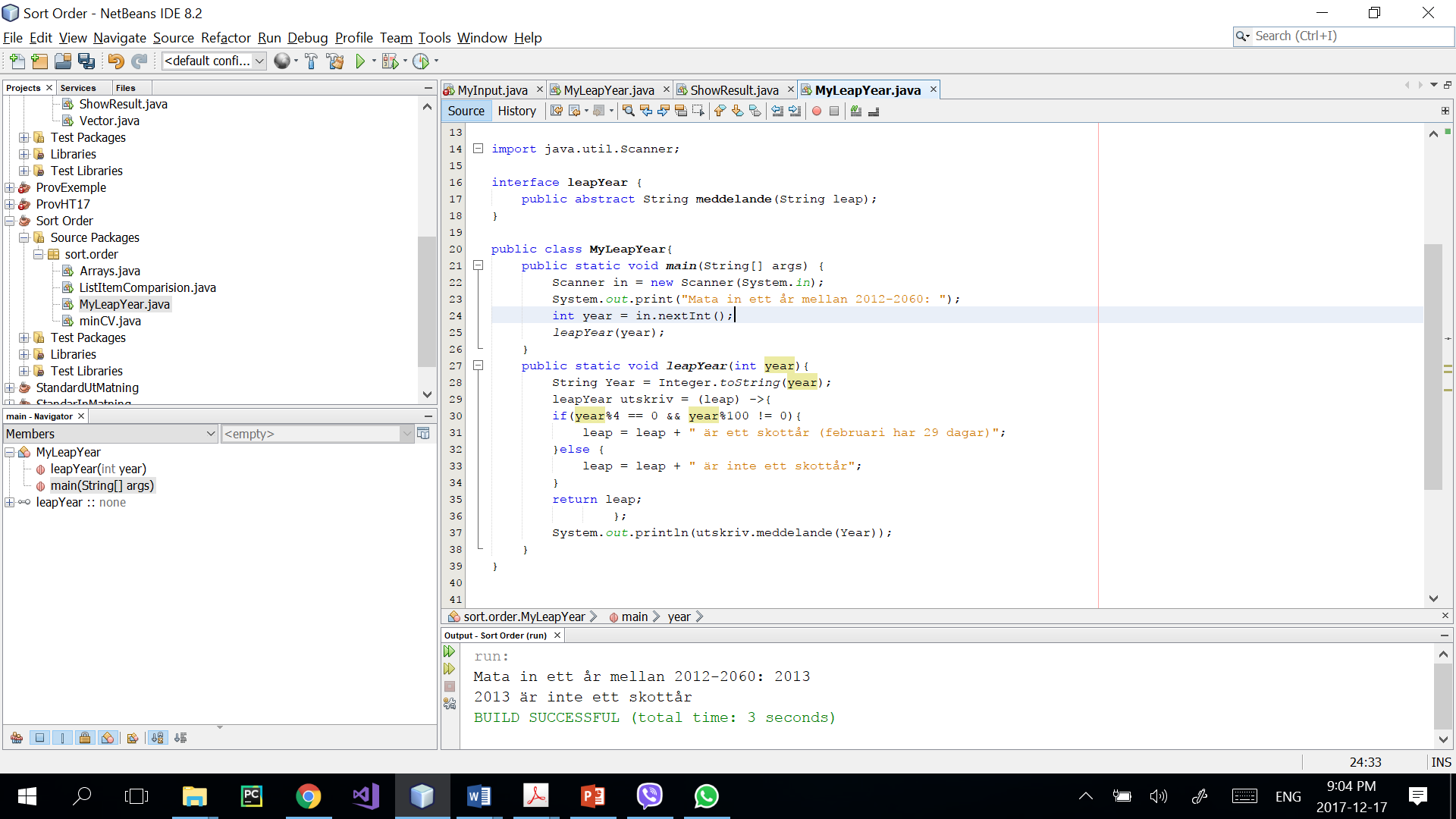


Gränssnittet Comparable har en abstrakt metod comparTo() som är tomt utan implementera någonting.

Klassen Book implementerar gränssnittet Comparable och deklarerar private instans variabler författare, titel, plats, förlag och utgiven. Klassen Book har metod getAuthor(), getPublished(), getTitel(), getPlace(), och getPublisher() som kan hämta information till boken som deklarerar private. Klassen Book har metod toString() som konvertera alla information i boken till ett string och implementera metod comparTo() som är deklarerar redan i gränssnittet Comparable.

Applikationen MyBooks skapar en lista som nästa innehåller flera böcker med implementerar ArrayList av typ objekt klassen Book. Med metod add() lägger i böcker listan alla informationer av författare, titel, plats, förlag och utgiven. Sedan utskriver alla element som alla informationer i böcker listan med loop. Sedan sorterar alla böcker i listan baserad av författare namn med metoden Collection.sort() använder override metoden compare() och utskriver dessa list efter sorteringen.

# Uppgift 5: leapYear och MyLeapYear

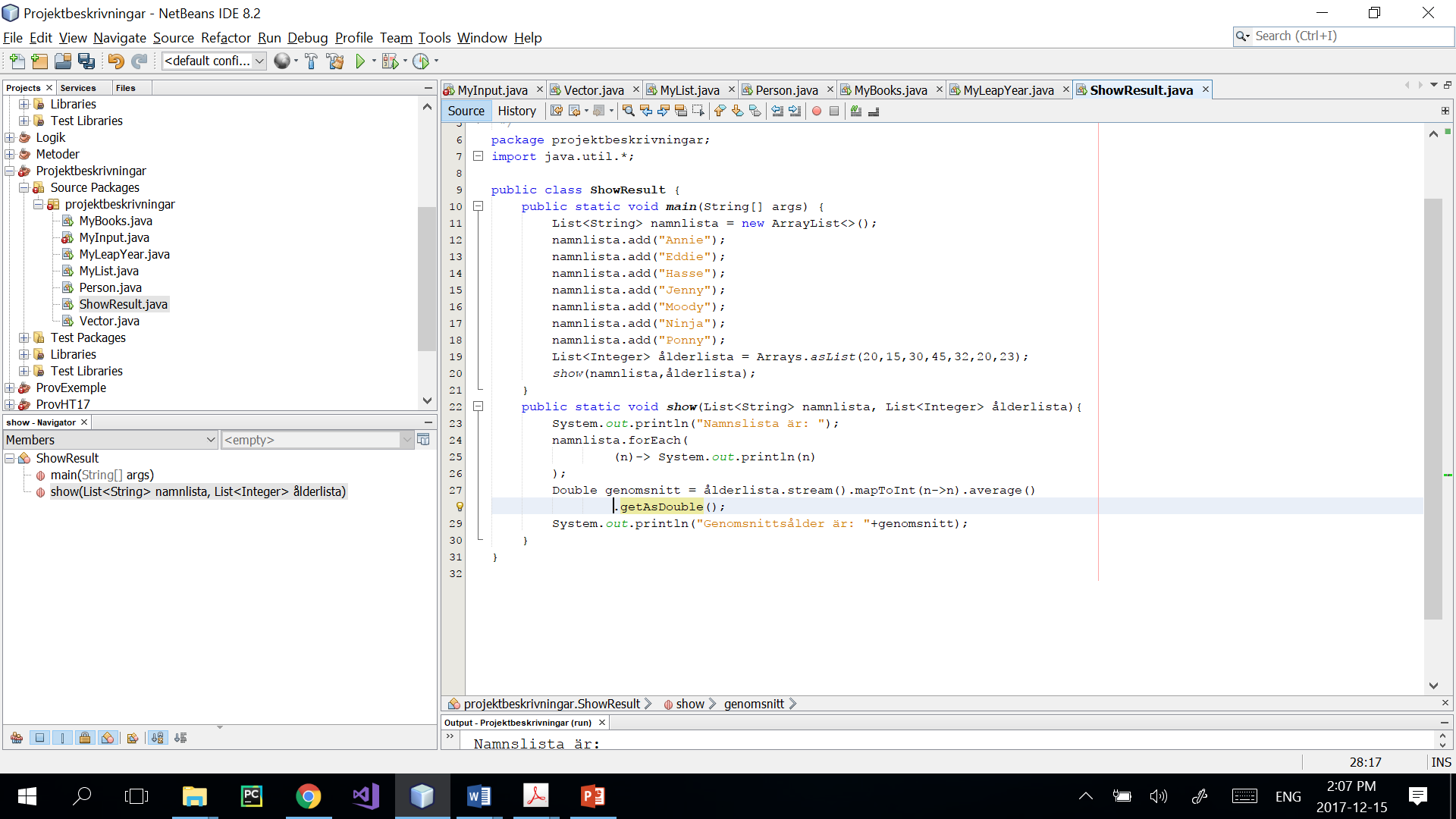


Gränssnittet leapYear har en abstrakt metod meddelande() som är tomt utan implementera någonting.

Klassen MyLeapYear hämtar heltal som matas in från standardinmatningsenheten med metoder nextInt() som finns i klassen Scanner och lagras i en variabel. Sedan anropa metoden leapYear med parametrar variabler som har redan sparats.

Metoden leapYear med parametrar heltal som hämtas från klassen MyLeapYear implementerar lambda funktion för att bestämma detta heltal är skottår eller inte med if påstående och sedan utskriver metoden resultatet.

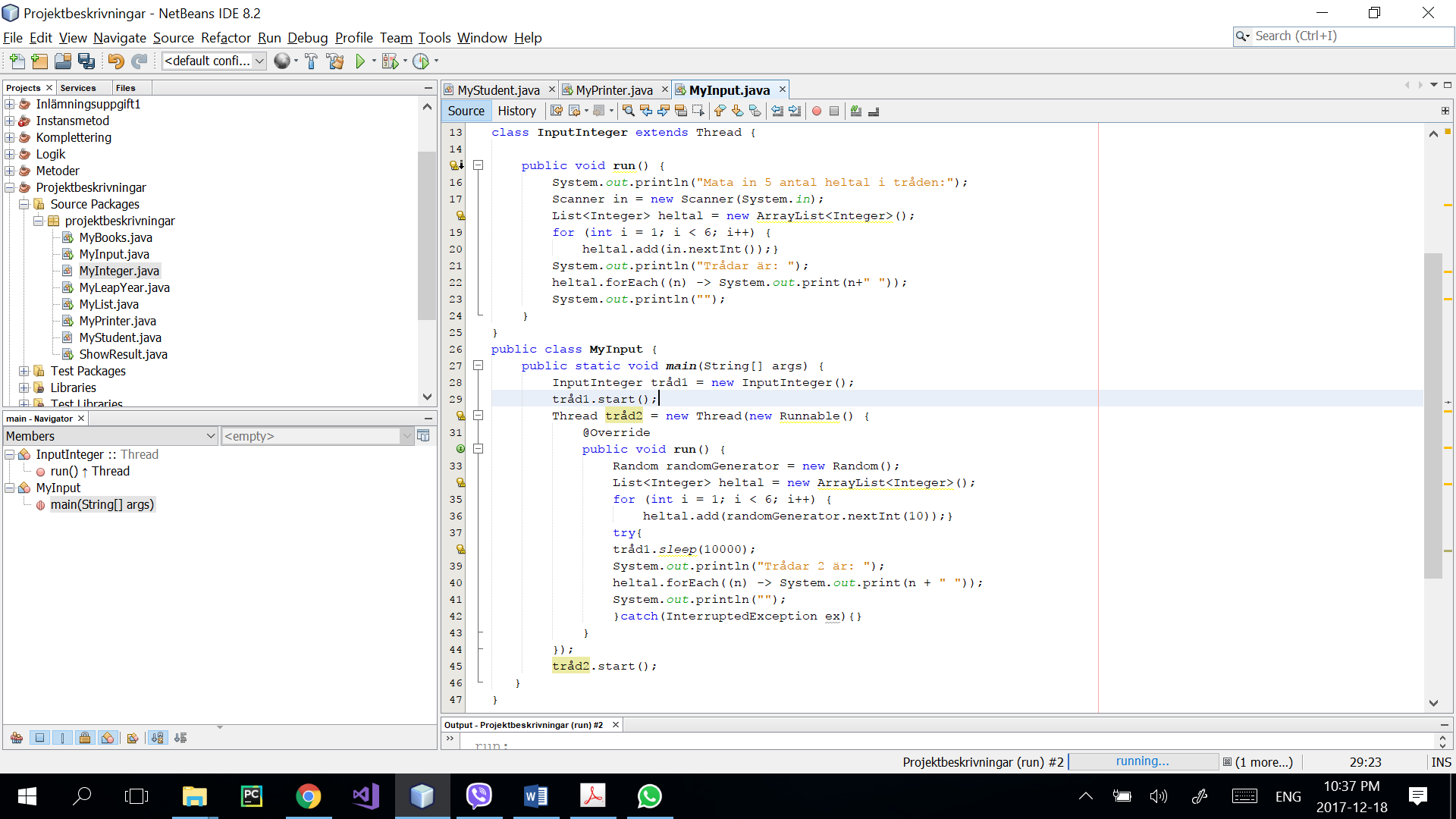
# Uppgift 6: show och MyResult



Klassen ShowResult skapar två listan av typ ArrayList. En list innehåller namnlista som är Sträng och andra list innehåller ålderlista. Sedan anropar metoden show med parametrar ålderlista och namnlista.

Metoden show implementerar lambda funktion för att utskriva alla element i namnlistan och få genomsnitt från ålderlista som också utskriver på skärmen.

# Uppgift 7: InputInteger och MyInput

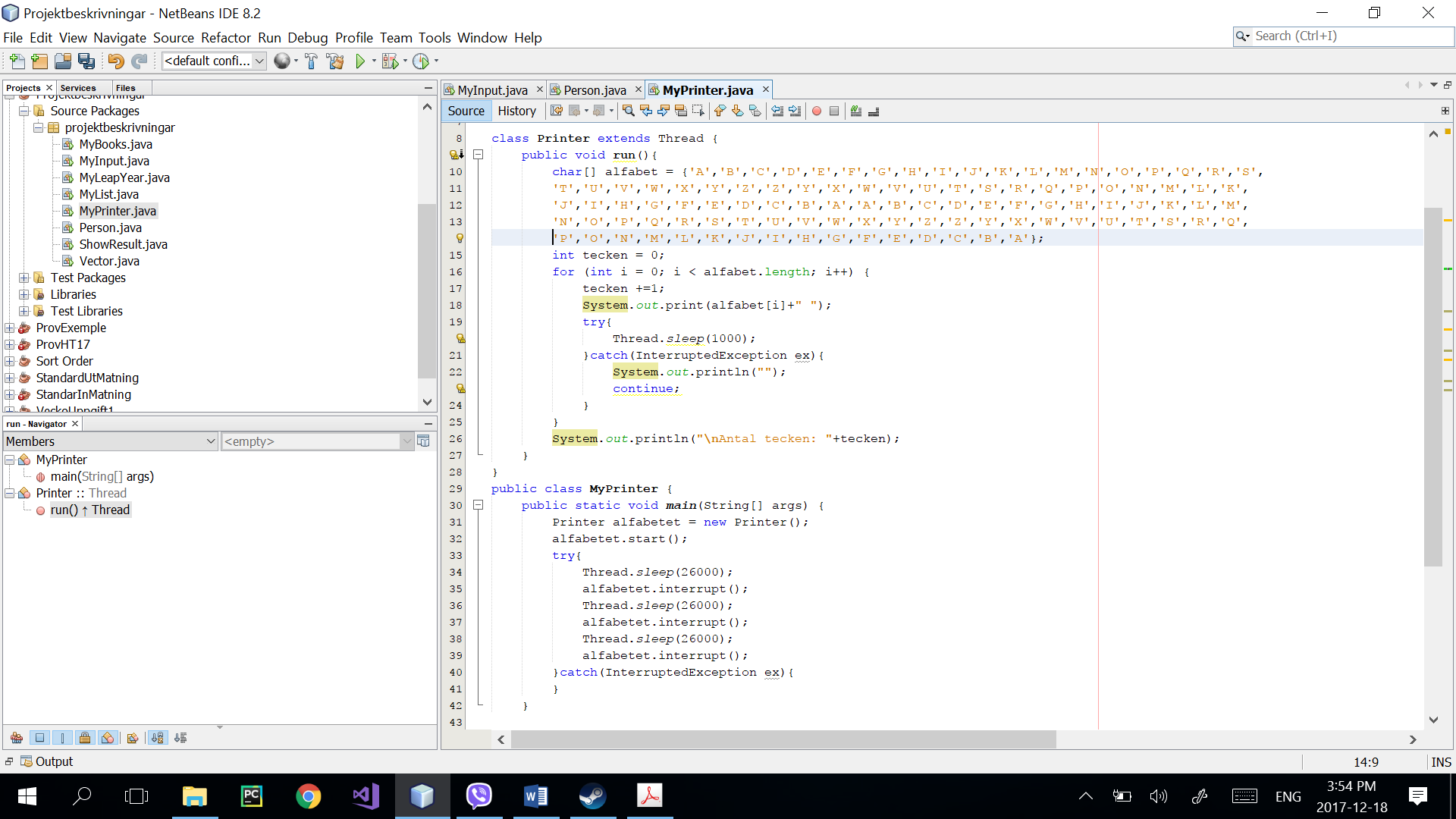


Klassen InputInteger ärver från klassen Thread. Klassen InputInteger skapar en vektor typ av ArrayList som innehåller ett antal heltal. Dessa heltal matar in från objekt Scanner med metod nextInt(). Sedan utskriver Klassen InputInteger trådar denna vektor.

Applikationen MyInput skapar två heltalvektorer och en tråd av typen InputInteger. Efter tråden startar matar första tråden in från objekt Scanner med metod nextInt() och sedan utskriver dessa antal heltal. Den andra tråden väntar 10 sekunder efter första tråden innan börjar den att fylla med slumpmässigt heltal som implementerar objekt Random och metod nextInt() och sedan utskriver dessa slumpmässigt heltal.

# Uppgift 8: StringArray och MyStringArray

# Uppgift 9: Printer och MyPrinter



Klassen Printer ärver från klassen Thread, som implementerar metod run(). Metoden run() skapar vektorn av typ char och skriver ut engelska alfabetet från A till Z och Z till A, och detta ska upprepas i två gånger. Varje tecken skriver ut en sekund efter varandra med implementerar metoden sleep(). Efter exekvering tråd skriver också ut antal tecken som skriver ut. Tråden avslutar exekvering när den får en interrupt-signal och sedan försätter igen skriver ut element i tråden.

Applikationen MyPrinter skapar och statar en tråd av typen Printer. Efter Printer skriver ut klart från A till Z eller från Z till A, skickar tråden en interrupt-signal för att avbryta tråden.