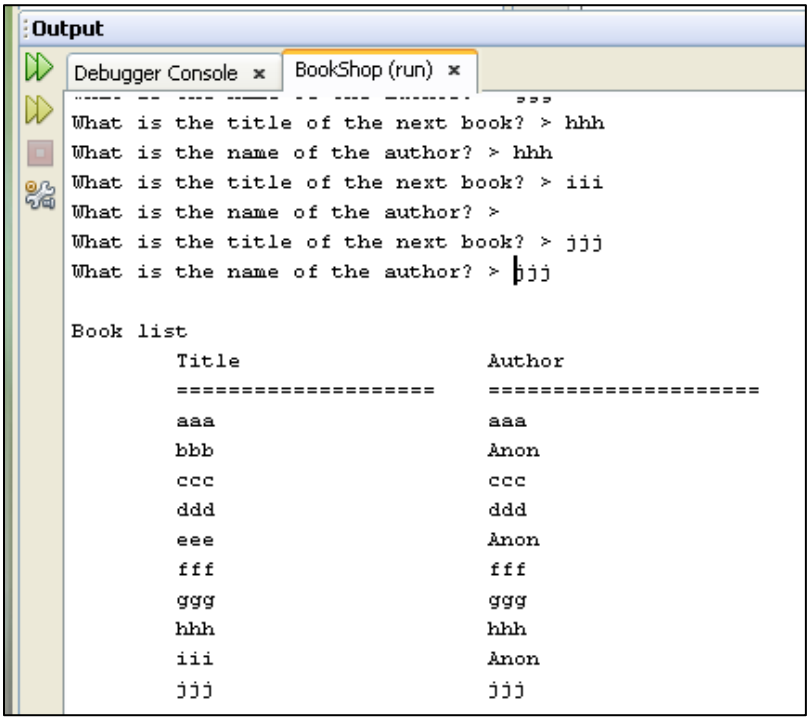
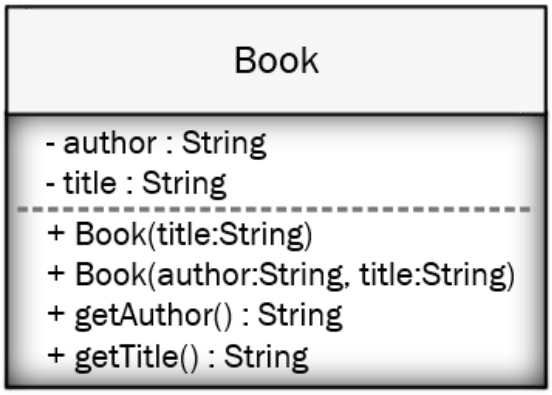
Practical session 1

This work should be completed before the next lecture.

Task 1: Book shop

Write an application that inputs and stores the titles and authors of ten books, and then outputs the book details to the console window. Use the UML class diagrams and accompanying pseudo-code given below. Follow them precisely. Your output should have a format similar to this screen shot.

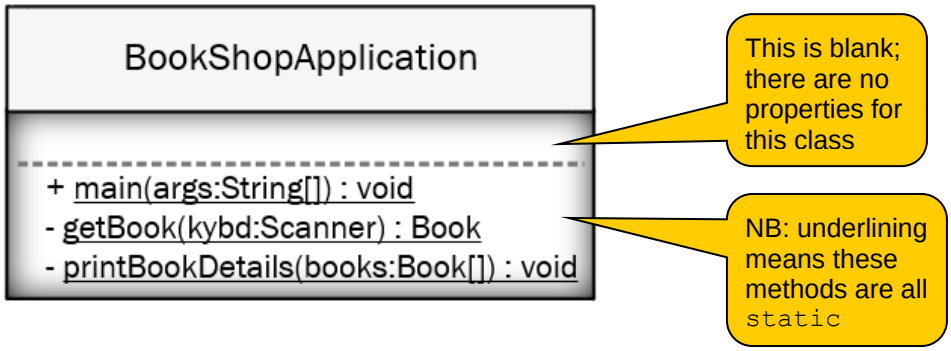




Pseudo-code for Book

```
Book(title:String)
      this.title = title
      author = "Anon"
```

All other methods have expected behaviour



Pseudo-code for BookShopApplication

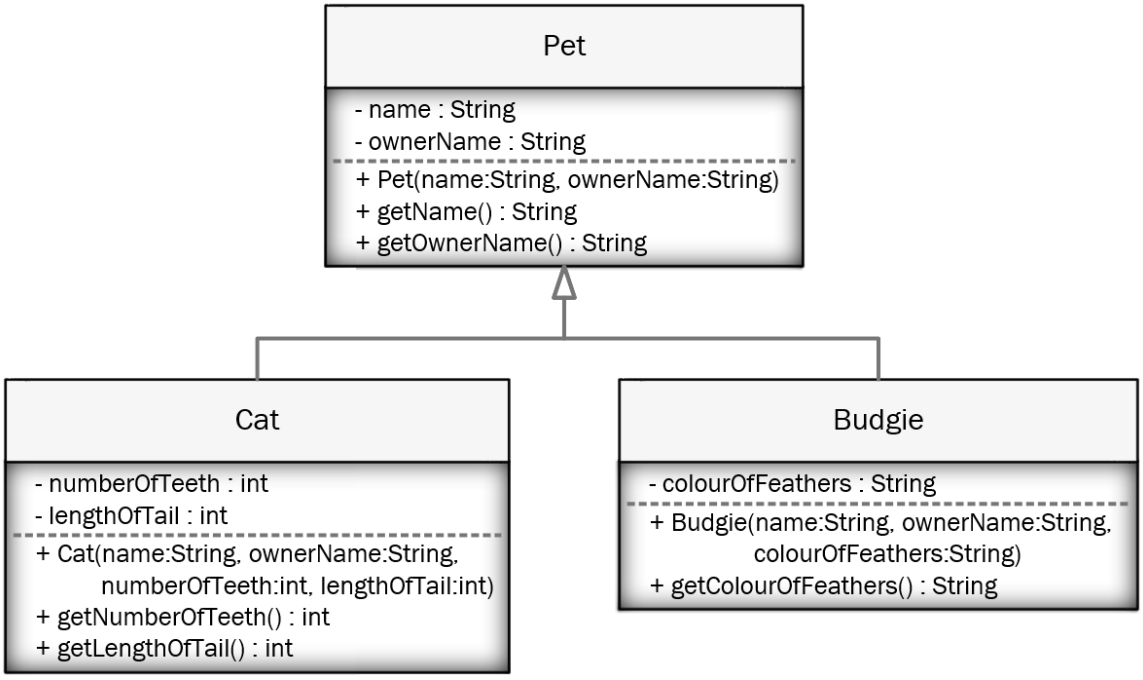
```
main(args:String[]) : void
      kybd = a scanner for reading from the keyboard;
      create an array called books to hold 10 Book objects

      for i = 0, books.length do
            books[i] = getBook(kybd)
      end for
      printBookDetails(books)


getBook(kybd:Scanner) : Book
      prompt = "What is the title of the next book? > "
      read title using kybd

      prompt = "What is the name of the author? > "
      read author using kybd

      if author is blank
            create new Book with title
      else
            create new Book with title and author
      end if

      return newly created Book


printBookDetails(books:Book[]) : void
      write "Book list"
      write "Title   Author"
      write "=====   ======"

      for i=0, books.length do
            write title and author of books[i]
      end for
```

Portfolio requirements:

- The NetBeans project for this completed task

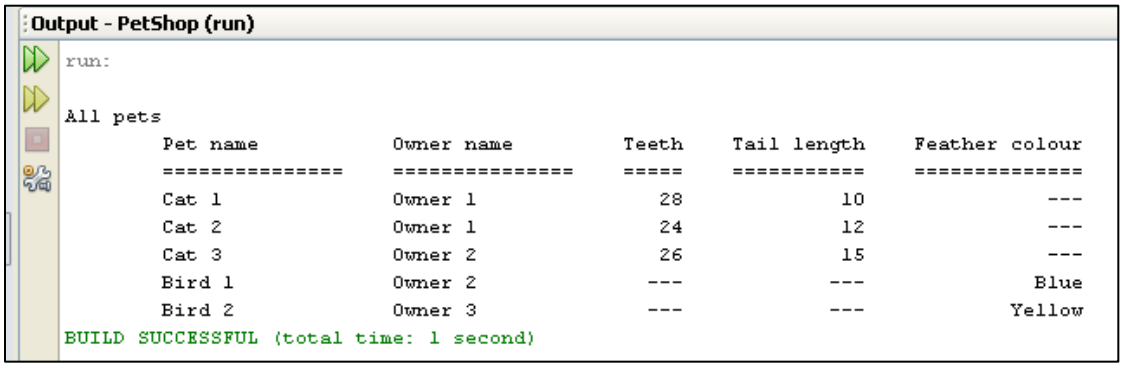
Task 2: Pet shop with inheritance

Convert the UML class diagrams shown below into Java code. Be precise.



Hint: Remember to ensure that the constructor methods in the subclasses call the superclass's constructor method.

Add to your project a Java class called `PetShopApplication`. The `main()` method should create three `Cat` objects and two `Budgie` objects. Next, the `main()` method should output the details of all pets to the console window in a tabulated format, similar to the illustration below.



Portfolio requirements:

- The NetBeans project for this completed task

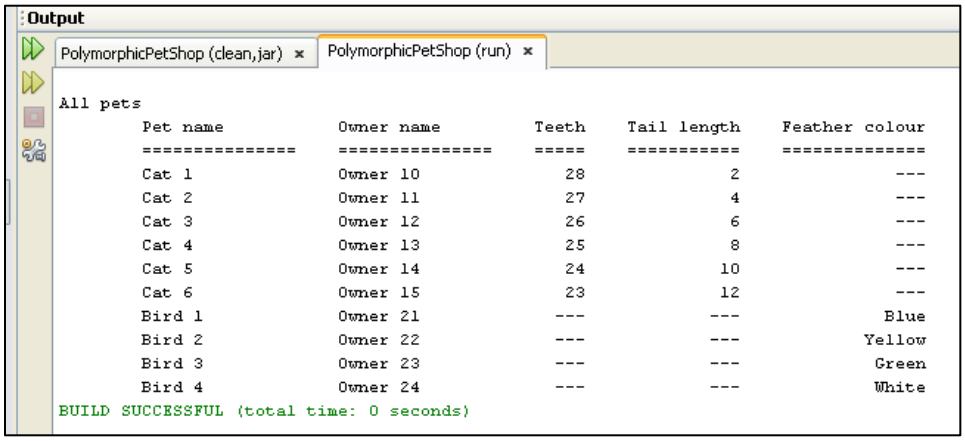
Task 3: Pet shop with polymorphism

Make a copy of your NetBeans project from Task 2 and modify it so that polymorphism is used.

Instead of storing the pets in separate variables, use an array.

In the `PetShopApplication` class, add a method called `printPetDetails()`, which outputs to the console window the contents of the `pets` array in a tabulated format, similar to the illustration below. The `pets` array must be passed to the method as a parameter. Use the `instanceof` operator as shown in the `main()` method, `InheritanceApplication` class, `PersonInheritance3` project from the lecture.

The `main()` method should create six `Cat` objects and four `Budgie` objects, storing them all in the `pets` array, and then call the `printPetDetails()` method, passing the `pets` array as the parameter.



Portfolio requirements:

- The NetBeans project for this completed task