Object Oriented Analysis and Design

Lab 03

OOAD - Week 3
Exercises

Exercise 1 (8.10 from Book)

Exercise - using constants in and outside a class

AIM:

- use of user-defined constants in and out of your own classes

Do the following:

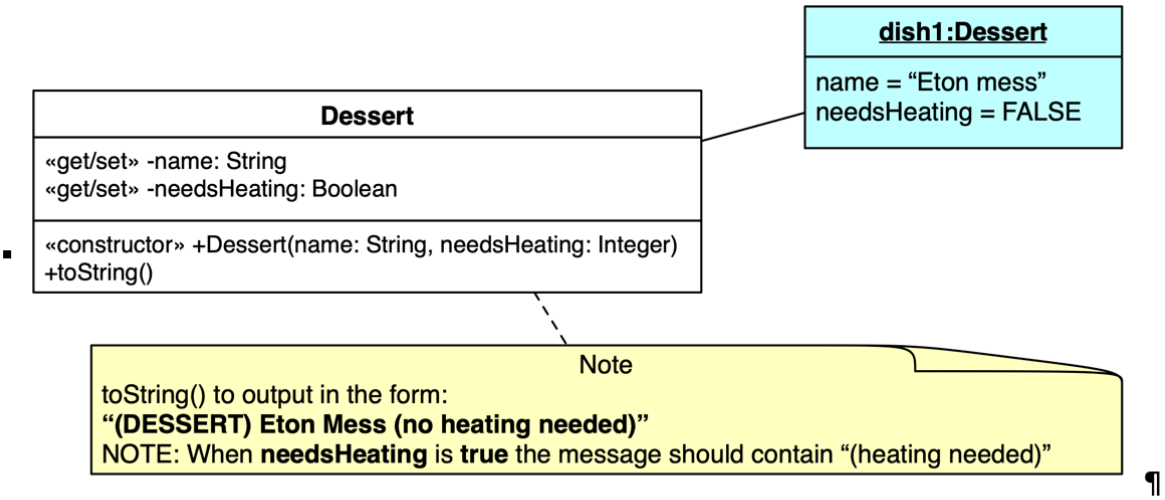
- write class `FamilyTicket` as follows:
    - make all properties private and add get/set methods for all properties
    - define a constant MAX_PEOPLE = 4
    - with private String `familyName`
    - with private Integer `numberTravelling`
    - the `setNumberTravelling()` method should use the constant to **not** allow more numbers travelling than definerd by the constant
- in the `main()` method of a class `Main` do the following:
    - create 2 object-instances of your class
    - try to set the number of travellers to 3 for family1, and to 6 for family2
    - output values from the objects to see if your setter validation worked
    - also print out the value of the constant

Output

```
max per family = 4
FamilyTicket{familyName='Smiths', numberTravelling=3}
FamilyTicket{familyName='Murphys', numberTravelling=0}
```

Exercise 2 (9.8 from Book)

Exercise - choosing a String message from a Boolean property value



Class-Object·diagram·for·Dessert·with·Boolean-to-String·output.¶

AIM:

- learn to write a "helper method", to output a useful String based on a boolean property value

ACTION:

- declare a class called `Dessert`:
    - with `private` String property `name`
    - with `private` Boolean property `needsHeating`
    - with a constructor that initialised both properties
    - with public get and set methods for both properties
    - with public `toString()` method to return a String summary of the object's state
        - in the form `(DESSERT) <name> (<no> heating needed)`
        - where the text in parentheses is `(heating needed)` if property `needsHeating` is true, and `(no heating needed)` otherwise

- in the `main()` method of a class `Main` do the following:
    - Create an instance of `Dessert` named `dish`, which is an "Eton Mess" that does NOT need heating
    - print out the object's state via its `toString()` method, i.e.

```
System.out.println(dish1);
```

- compile (`javac *.java`) and run your program
-

OUTPUT:

```
(DESSERT) Eton mess (no heating needed)
(DESSERT) Christmas pudding (heating needed)
```

HINT:

- avoid an `if`-statement in your `toString()` method by creating a "helper method":
    - `private String needsHeatingString()`
    - this returns "(heating needed)" if needsHeating is true, and "(no heating needed)" otherwise

Exercise 3 (10.8 from Book)

Exercise - loop through array of CLI arguments

```
% java Main one two three
there were 3 command line arguments
argument: one
argument: two
argument: three
```

Running loop with 3 arguments.

AIM:

- to start working with the array of strings that are the command line arguments

ACTION:

- write code inside the `main()` method of class `Main` that loops through the array `args` to output messages in the form:

  ```
  there were 2 command line arguments"
  argument: word1
  argument: word2   (and so on ...)
  ```

- compile the class

  ```
  javac *.java
  ```

- run the application passing in command line arguments `one two three`

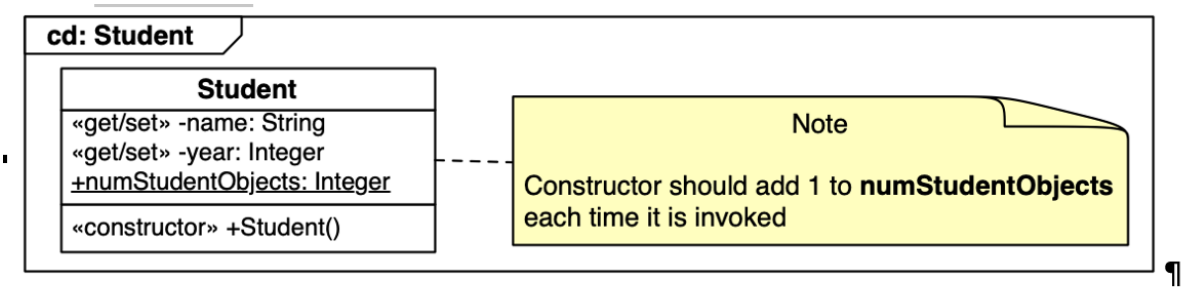
OUTPUT:

```
$ java Main one two three
there were 3 command line arguments
argument: one
argument: two
argument: three

$ java Main "one two three"
there were 1 command line arguments
argument: one two three
```

Exercise 4 (11.6 from Book)

Exercise - constructor to add to object count each time new object created



Class-Object·diagram·for·`static`·variable·`numStudentObjects`.

AIM:

- practice working with `static` variables and methods

Do the following:

- create the following class
    - class `Student` with private properties and public getters/setters:
        - `name`: String
        - `year`: Integer
        - a public **static** variable `numStudentObjects`: Integer with default value zero
        - a constructor method, that adds 1 to static variable `numStudentObjects` each time a new object is created
- create a `Main` class with a `main()` method to:
    - output the static variable `numStudentObjects` of class `Student` (should be zero)
    - create a student object `student1` & print out message `created student1`
    - output the static variable `numStudentObjects` of class `Student` (should be 1)
    - create a student object `student2` & print out message `created student2`
    - output the static variable `numStudentObjects` of class `Student` (should be 2)

OUTPUT:

```
java Main

num student objects = 0
created student1
num student objects = 1
created student2
num student objects = 2
```