INFO1113 Week 8 Tutorial

Testing and Enums

An `enum` is a data type which specifies predefined instances. An enumerated type can be a simple label that only has its label name and order number or can include its own fields, methods and constructor for the type.

```
public enum TimeOfDay {
    Dawn,
    Morning,
    Noon,
    Afternoon,
    Evening,
    Night,
    Midnight;
}
```

Question 1: Weekdays

Write an enum to hold the days of the week. Use this to write a program that takes one command-line argument that is a number representing the day of the week, and print out what day that number represents, as well as if that day is a weekend or not. Note: If you have an enum called Weekdays, you can get an array of all possible values for the enum with the method Weekdays.values().

```
> java WhatDayIs 3
Day 3 of the week is: WEDNESDAY. It is not a weekend.
> java WhatDayIs 0
Day 0 of the week is: SUNDAY. It is a weekend.
```

Question 2: N Days

Change the program to take in two arguments: the current day of the week and a number of days after that. The program should then output what day of the week it will be that number of days from the starting day.

```
> java DaysFromNow 1 9
The current day is MONDAY, and in 9 days time it will be WEDNESDAY.
```

JUnit

JUnit is a testing framework for java, it allows you construct test cases and utilise test methods to confirm the results from your methods and classes. You can get access to JUnit and Hamcrest files from `Ed`'s resources section (under the 'Libraries' category).

To compile your code with junit, you will need to link your code to the junit.jar file and hamcrest.jar file. Place them in the folder in the same folder as your test class. You can then compile your code with the following command:

(Substitute MyTestClass with your own test class file)

```
javac -cp .:junit-4.12.jar:hamcrest-core-1.3.jar MyTestClass.java
```

And execute it with the following (Substitute MyTestClass with your own test class file):

```
java -cp .:junit-4.12.jar:hamcrest-core-1.3.jar
 org.junit.runner.JUnitCore MyTestClass
```

Question 3: Testing your collections

For the collections you created in the previous week. Design a set of test cases that will ensure the classes will behave as expected. Make sure that each of the constructors and methods in each class work as expected. If you haven't already, you should now be able to use these test to write the JUnit Test cases.

```java
public class CollectionTest {

    @Test
    public void testConstruction() {
        //Your test case
    }

    @Test
    public void testAdd() {
        //Your test case
    }

    @Test
    public void testRemove() {
        //Your test case
    }

    @Test
    public void testSet() {
        //Your test case
    }
}
```

Question 4: Contacts

You are going to write a program that manages a list of contacts (like the one in your phone). You will need to write the following classes:

- `ContactList` - A class that holds a collection of `Contact` objects. This class should be able to save its contents to a file, and also load them back again.
- `Contact` - a class representing a contact. Every `Contact` has a name, an `Address`, an array of `EmailAddresses` and an array of `PhoneNumbers`.
- `Address` - an address has two street address lines, a postcode, a state and a country.
- `EmailAddress` - an email address contains a string to represent the email address, as well as whether the email is personal, work or other.
- `PhoneNumber` - a phone number contains a string to represent the phone number, as well as whether the phone number is personal, work, or other.

The amount of detail that you put into this is up to you. Consider where you can use enums: Tagging an email/phone number is work, personal with enum type; as well as the addresses of the contact, if you wish to limit the list.

Consider what are valid entries for email addresses, phone numbers and addresses. Add as much or as little validation code (bad input checking) as you feel is necessary to make sure the data is consistent. For example what kinds of characters should you be able to put in a phone number?

Make sure that you create a set of tests and perform regression testing. Every time that you update your program, run the tests again to make sure that you haven't broken any existing functionality.

Question 5: Assessed Task - Online Task 4 and Assignment 2

Remember you are required to complete a Online Task and the final assignment within the due date. Go to EdStem for this unit and click on Assessment to find out the tasks and the due date. This are a marked tasks.

Extension

Question 6: Coins

Write an `enum` class that represents valid coin denominations. The enum should have a constructor so that you can give each denomination a value. What kind of data would this store? What kind of methods would you want to be able to perform on a coin? Try writing the payment part of a vending machine using this `enum`, and see if you can make the vending machine only accept valid coin denominations this way.

Question 7: Photos for your contacts

Add an extra property to your `Contact` that will allow a photo of the contact to be stored. Look into how you can store and manipulate images in Java. In particular, the `BufferedImage` class will be a good place to look. Modify your contact list program to also store an optional contact picture for each Contact . How would you store this when saving the file?