# 70094 ExerciseTypes.CW2

## Contact Manager

## Submitters

| | |
|---|---|
| sf23 | Shihan Fu |

```
Final Tests: Summary for sf23 of v5
-----------------------------------

  Public Tests:
    Compiles:      1 / 1
    Tests Pass:    1 / 1
    Style Checks:  0 / 1

Git Repo: git@gitlab.doc.ic.ac.uk:lab2324_spring/SE_Design_Ex2_sf23.git
Commit ID: 4de6f
```

```java
package contacts;

import static org.hamcrest.Matchers.containsString;
import static org.junit.Assert.assertThat;

import java.io.ByteArrayOutputStream;
import org.junit.Test;

public class PhoneTest {

  @Test
  public void sendMessage() {
    ByteArrayOutputStream outstream = ↗
TestSuiteHelper.replaceSystemOutStreamForTesting();

    String phoneNumber = "+501 323 33";
    String msg = "Hello";

    Phone p1 = new Phone(phoneNumber);
    p1.sendMessage(msg);

    String actualOutput = outstream.toString();

    assertThat(actualOutput, containsString(msg));
    assertThat(actualOutput, containsString(phoneNumber));
    assertThat(actualOutput, containsString("Audio"));
  }
}
```

```java
package contacts;

import static org.hamcrest.Matchers.containsString;
import static org.hamcrest.Matchers.not;
import static org.junit.Assert.assertThat;

import java.io.ByteArrayOutputStream;
import org.junit.Test;

public class MobilePhoneTest {

  @Test
  public void sendMessage() {
    ByteArrayOutputStream outstream = ↗
TestSuiteHelper.replaceSystemOutStreamForTesting();

    String phoneNumber = "+501 323 33";
    String msg = "Hello";

    MobilePhone p1 = new MobilePhone(phoneNumber);
    p1.sendMessage(msg);

    String actualOutput = outstream.toString();

    assertThat(actualOutput, containsString(msg));
    assertThat(actualOutput, containsString(phoneNumber));
    assertThat(actualOutput, not(containsString("Audio")));
  }
}
```

```java
package contacts;

import static org.hamcrest.Matchers.containsString;
import static org.hamcrest.Matchers.not;
import static org.junit.Assert.assertThat;

import java.io.ByteArrayOutputStream;
import org.junit.Test;

public class EmailTest {

  @Test
  public void sendMessage() {
    ByteArrayOutputStream outstream = ⤦
TestSuiteHelper.replaceSystemOutStreamForTesting();

    String address = "myEmail@domain.com";
    String msg = "Hello";

    Email email = new Email(address);
    email.sendMessage(msg);

    String actualOutput = outstream.toString();

    assertThat(actualOutput, containsString(msg));
    assertThat(actualOutput, containsString(address));
    assertThat(actualOutput, not(containsString("Audio")));
  }
}
```

```java
package contacts;

import static org.hamcrest.CoreMatchers.*;
import static org.hamcrest.MatcherAssert.assertThat;
import static org.junit.Assert.assertEquals;

import java.io.ByteArrayOutputStream;
import java.util.List;
import org.junit.Test;

public class ContactManagerTest {

  @Test
  public void listsContactInfo() {
    ContactManager cm = new ContactManager();
    Person person1 = new Person("M. P. J.");
    Person person2 = new Person("M. P. Junior");
    Phone p1 = new Phone("23245242");
    Email m1 = new Email("sdf@dfodfs.asdf");
    Phone mp1 = new MobilePhone("sdfasdf");
    Email p2 = new Email("aslkdfa");

    cm.add(person1, p1);
    cm.add(person1, m1);
    cm.add(person1, mp1);
    cm.add(person1, p2);
    cm.add(person2, p2);

    List<ContactInfo> l = cm.contactDetails(person1);
    assertEquals(4, l.size());
    assertThat(l, hasItem(equalTo(p1)));
    assertThat(l, hasItem(equalTo(m1)));
    assertThat(l, hasItem(equalTo(mp1)));
    assertThat(l, hasItem(equalTo(p2)));

    l = cm.contactDetails(person2);
    assertEquals(1, l.size());
    assertThat(l, hasItem(equalTo(p2)));
    assertThat(l, not(hasItem(equalTo(p1))));

    assertEquals(0, cm.contactDetails(new Person("new person")).size());
  }

  @Test
  public void spamContacts() {
    ContactManager cm = new ContactManager();
    Person person1 = new Person("P1");
    Person person2 = new Person("P2");
    Phone p1 = new Phone("+114 53 132");
    Email m1 = new Email("emailP1@address.com");
    Phone mp1 = new MobilePhone("+434 9434 132");
    Email m2 = new Email("email2@address.com");

    ByteArrayOutputStream outstream = ⤦
TestSuiteHelper.replaceSystemOutStreamForTesting();

    cm.add(person1, p1);
    cm.add(person1, m1);
    cm.add(person1, mp1);
    cm.add(person2, m2);

    cm.spam("Hello");

    String actualOutput = outstream.toString();

    assertThat(actualOutput, containsString(m1.contactInfo()));
    assertThat(actualOutput, containsString(mp1.contactInfo()));
    assertThat(actualOutput, containsString(m2.contactInfo()));
    assertThat(actualOutput, containsString(p1.contactInfo()));
    assertThat(actualOutput, containsString("Hello"));
  }
}
```

```java
package contacts;

/**
 * @time 25/01/2024 09:58
 * @version 1.0
 * @author fsh
 * @description:
 */
interface TextMessageEnabled {
  void sendTextMessage(String msg);
}
```

```java
package contacts;

/**
 * @time 25/01/2024 10:08
 * @version 1.0
 * @author fsh
 * @description:
 */
public class Phone extends ContactInfo implements AudioMessageEnabled {

  Phone(String phone_number) {
    super(phone_number);
  }

  String contactInfo() {
    return info;
  }

  String contactInfoType() {
    return "Phone";
  }

  void sendMessage(String msg) {
    System.out.println("Audio message: " + msg);
    System.out.println(("Phone number: " + this.info));
  }

  public void sendAudioMessage(Audio msg) {
    System.out.println("Audio message: " + msg.toString());
    System.out.println(("Phone number: " + this.info));
  }
}
```

You should add access modifier

```java
package contacts;

/**
 * @time 25/01/2024 10:13
 * @version 1.0
 * @author fsh
 * @description:
 */
public class MobilePhone extends Phone implements AudioMessageEnabled, TextMessageEnabled {
  MobilePhone(String phone_number) {
    super(phone_number);
  }

  void sendMessage(String msg) {
    System.out.println("message: " + msg);
    System.out.println(("Phone number: " + this.info));
  }

  public void sendTextMessage(String msg) {
    System.out.println("msg: " + msg);
    System.out.println("email: " + this.info);
  }
}
```

```java
package contacts;

/**
 * @time 25/01/2024 09:59
 * @version 1.0
 * @author fsh
 * @description:
 */
public class Email extends ContactInfo implements TextMessageEnabled {

  Email(String email) {
    super(email);
  }

  String contactInfo() {
    return this.info;
  }

  String contactInfoType() {
    return "Email";
  }

  void sendMessage(String msg) {
    System.out.println("msg: " + msg);
    System.out.println("email: " + this.info);
  }

  public void sendTextMessage(String msg) {
    System.out.println("msg: " + msg);
    System.out.println("email: " + this.info);
  }
}
```

```java
package contacts;

import java.util.ArrayList;
import java.util.List;

/**
 * @time 25/01/2024 10:17
 * @version 1.0
 * @author fsh
 * @description:
 */
public class ContactManager {
  List<Contact> list = new ArrayList<Contact>();

  void add(Person p, ContactInfo info) {
    list.add(new Contact(p, info));
  }

  List<ContactInfo> contactDetails(Person p) {
    List<ContactInfo> ret = new ArrayList<ContactInfo>();
    for (Contact c : list) {
      if (c.p.name().equals(p.name())) {
        ret.add(c.info);
      }
    }
    return ret;
  }

  void spam(String msg) {
    for (Contact c : list) {
      c.info.sendMessage(msg);
    }
  }
}
```

```java
package contacts;

/**
 * @time 25/01/2024 09:54
 * @version 1.0
 * @author fsh
 * @description:
 */
abstract class ContactInfo {
  String info;

  ContactInfo(String info) {
    this.info = info;
  }

  abstract String contactInfo();

  abstract String contactInfoType();

  abstract void sendMessage(String msg);
}
```

```java
package contacts;

/**
 * @time 25/01/2024 09:57
 * @version 1.0
 * @author fsh
 * @description: contact that combines person with contactInfo
 */
public class Contact {
  Person p;
  ContactInfo info;

  Contact(Person p, ContactInfo info) {
    this.p = p;
    this.info = info;
  }
}
```

```java
package contacts;

/**
 * @time 25/01/2024 10:07
 * @version 1.0
 * @author fsh
 * @description:
 */
interface AudioMessageEnabled {
  void sendAudioMessage(Audio msg);
}
```

```
-------- Test Output --------
Running LabTS build...

Submission summary...
You made 2 commits
  - 7cf0943 version 1 basic implementation [8 files changed, 168 insertions]
  - 4de6f78 version 2 fix phone and mobilephone sendMessage [8 files changed, 88 insertions, 75 deletions]

Preparing...

BUILD SUCCESSFUL in 641ms

Compiling...
BUILD SUCCESSFUL in 2s

Running tests...

contacts.MobilePhoneTest > sendMessage PASSED

contacts.EmailTest > sendMessage PASSED

contacts.PhoneTest > sendMessage PASSED

contacts.ContactManagerTest > spamContacts PASSED

contacts.ContactManagerTest > listsContactInfo PASSED

BUILD SUCCESSFUL in 1s

Checking code style...


-------- Test Errors --------
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/AudioMessageEnabled.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Contact.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Contact.java:10:10: Member name 'p' must match pattern '^[a-z][a-z0-9][a-zA-Z0-9]*$'. [MemberName]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/ContactInfo.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/ContactManager.java:6: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Email.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/MobilePhone.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/MobilePhone.java:10:22: Parameter name 'phone_number' must match pattern '^[a-z]([a-z0-9][a-zA-Z0-9]*)?$'. [ParameterName]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Phone.java:3: Summary javadoc is missing. [SummaryJavadoc]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Phone.java:11:16: Parameter name 'phone_number' must match pattern '^[a-z]([a-z0-9][a-zA-Z0-9]*)?$'. [ParameterName]
[ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/TextMessageEnabled.java:3: Summary javadoc is missing. [SummaryJavadoc]

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':checkstyleMain'.
> Checkstyle rule violations were found. See the report at: file:///tmp/d20240125-38-ama9j5/build/reports/checkstyle/main.html
  Checkstyle files with violations: 8
  Checkstyle violations by severity: [warning:11]


* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get more log output. Run with --scan to get full insights.

* Get more help at https://help.gradle.org

BUILD FAILED in 5s
```

Mark: 16/20

Great job! You've successfully implemented the required classes and methods. However, there are a few areas for improvement. Firstly, please ensure the code is properly formatted before submission. Additionally, remember to include access modifiers when declaring variables or methods. Keep up the good work