

70094 ExerciseTypes.CW2

Contact Manager

Submitters

sf23

Shihan Fu

Emarking

Final Tests**TestSummary.txt: 1/1****Shihan Fu - sf23:v5**

```
1: Final Tests: Summary for sf23 of v5
2: -----
3:
4:   Public Tests:
5:     Compiles:      1 / 1
6:     Tests Pass:    1 / 1
7:     Style Checks:  0 / 1
8:
9: Git Repo: git@gitlab.doc.ic.ac.uk:lab2324_spring/SE_Design_Ex2_sf23.git
10: Commit ID: 4de6f
```

Final Tests

PhoneTest.java: 1/1

Shihan Fu - sf23:v5

```
1: package contacts;
2:
3: import static org.hamcrest.Matchers.containsString;
4: import static org.junit.Assert.assertThat;
5:
6: import java.io.ByteArrayOutputStream;
7: import org.junit.Test;
8:
9: public class PhoneTest {
10:
11:     @Test
12:     public void sendMessage() {
13:         ByteArrayOutputStream outstream =
TestSuiteHelper.replaceSystemOutStreamForTesting();
14:
15:         String phoneNumber = "+501 323 33";
16:         String msg = "Hello";
17:
18:         Phone p1 = new Phone(phoneNumber);
19:         p1.sendMessage(msg);
20:
21:         String actualOutput = outstream.toString();
22:
23:         assertThat(actualOutput, containsString(msg));
24:         assertThat(actualOutput, containsString(phoneNumber));
25:         assertThat(actualOutput, containsString("Audio"));
26:     }
27: }
```

Final Tests

MobilePhoneTest.java: 1/1

Shihan Fu - sf23:v5

```
1: package contacts;
2:
3: import static org.hamcrest.Matchers.containsString;
4: import static org.hamcrest.Matchers.not;
5: import static org.junit.Assert.assertThat;
6:
7: import java.io.ByteArrayOutputStream;
8: import org.junit.Test;
9:
10: public class MobilePhoneTest {
11:
12:     @Test
13:     public void sendMessage() {
14:         ByteArrayOutputStream outstream =
TestSuiteHelper.replaceSystemOutStreamForTesting();
15:
16:         String phoneNumber = "+501 323 33";
17:         String msg = "Hello";
18:
19:         MobilePhone p1 = new MobilePhone(phoneNumber);
20:         p1.sendMessage(msg);
21:
22:         String actualOutput = outstream.toString();
23:
24:         assertThat(actualOutput, containsString(msg));
25:         assertThat(actualOutput, containsString(phoneNumber));
26:         assertThat(actualOutput, not(containsString("Audio")));
27:     }
28: }
```

Final Tests

EmailTest.java: 1/1

Shihan Fu - sf23:v5

```

1: package contacts;
2:
3: import static org.hamcrest.Matchers.containsString;
4: import static org.hamcrest.Matchers.not;
5: import static org.junit.Assert.assertThat;
6:
7: import java.io.ByteArrayOutputStream;
8: import org.junit.Test;
9:
10: public class EmailTest {
11:
12:     @Test
13:     public void sendMessage() {
14:         ByteArrayOutputStream outstream =
TestSuiteHelper.replaceSystemOutStreamForTesting();
15:
16:         String address = "myEmail@domain.com";
17:         String msg = "Hello";
18:
19:         Email email = new Email(address);
20:         email.sendMessage(msg);
21:
22:         String actualOutput = outstream.toString();
23:
24:         assertThat(actualOutput, containsString(msg));
25:         assertThat(actualOutput, containsString(address));
26:         assertThat(actualOutput, not(containsString("Audio")));
27:     }
28: }

```

Final Tests

ContactManagerTest.java: 1/2

Shihan Fu - sf23:v5

```

1: package contacts;
2:
3: import static org.hamcrest.CoreMatchers.*;
4: import static org.hamcrest.MatcherAssert.assertThat;
5: import static org.junit.Assert.assertEquals;
6:
7: import java.io.ByteArrayOutputStream;
8: import java.util.List;
9: import org.junit.Test;
10:
11: public class ContactManagerTest {
12:
13:     @Test
14:     public void listsContactInfo() {
15:         ContactManager cm = new ContactManager();
16:         Person person1 = new Person("M. P. J.");
17:         Person person2 = new Person("M. P. Junior");
18:         Phone p1 = new Phone("23245242");
19:         Email m1 = new Email("sdf@dfodfs.asdf");
20:         Phone mp1 = new MobilePhone("sdfasdf");
21:         Email p2 = new Email("aslkdfa");
22:
23:         cm.add(person1, p1);
24:         cm.add(person1, m1);
25:         cm.add(person1, mp1);
26:         cm.add(person1, p2);
27:         cm.add(person2, p2);
28:
29:         List<ContactInfo> l = cm.contactDetails(person1);
30:         assertEquals(4, l.size());
31:         assertThat(l, hasItem(equalTo(p1)));
32:         assertThat(l, hasItem(equalTo(m1)));
33:         assertThat(l, hasItem(equalTo(mp1)));
34:         assertThat(l, hasItem(equalTo(p2)));
35:
36:         l = cm.contactDetails(person2);
37:         assertEquals(1, l.size());
38:         assertThat(l, hasItem(equalTo(p2)));
39:         assertThat(l, not(hasItem(equalTo(p1))));
40:
41:         assertEquals(0, cm.contactDetails(new Person("new person")).size());
42:     }
43:
44:     @Test
45:     public void spamContacts() {
46:         ContactManager cm = new ContactManager();
47:         Person person1 = new Person("P1");
48:         Person person2 = new Person("P2");
49:         Phone p1 = new Phone("+114 53 132");
50:         Email m1 = new Email("emailP1@address.com");
51:         Phone mp1 = new MobilePhone("+434 9434 132");
52:         Email m2 = new Email("email2@address.com");
53:
54:         ByteArrayOutputStream outstream =
TestSuiteHelper.replaceSystemOutStreamForTesting();
55:
56:         cm.add(person1, p1);
57:         cm.add(person1, m1);
58:         cm.add(person1, mp1);
59:         cm.add(person2, m2);
60:
61:         cm.spam("Hello");
62:
63:         String actualOutput = outstream.toString();
64:
65:         assertThat(actualOutput, containsString(m1.contactInfo()));

```

Final Tests **ContactManagerTest.java: 2/2** **Shihan Fu - sf23:v5**

```
66:     assertThat(actualOutput, containsString(mp1.contactInfo()));
67:     assertThat(actualOutput, containsString(m2.contactInfo()));
68:     assertThat(actualOutput, containsString(p1.contactInfo()));
69:     assertThat(actualOutput, containsString("Hello"));
70: }
71: }
```

Final Tests **TextMessageEnabled.java: 1/1** **Shihan Fu - sf23:v5**

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 09:58
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: interface TextMessageEnabled {
10:     void sendTextMessage(String msg);
11: }
```

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 10:08
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: public class Phone extends ContactInfo implements AudioMessageEnabled {
10:
11:     Phone(String phone_number) {
12:         super(phone_number);
13:     }
14:
15:     String contactInfo() {
16:         return info;
17:     }
18:
19:     String contactInfoType() {
20:         return "Phone";
21:     }
22:
23:     void sendMessage(String msg) {
24:         System.out.println("Audio message: " + msg);
25:         System.out.println(("Phone number: " + this.info));
26:     }
27:
28:     public void sendAudioMessage(Audio msg) {
29:         System.out.println("Audio message: " + msg.toString());
30:         System.out.println(("Phone number: " + this.info));
31:     }
32: }
```

You should add access modifier

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 10:13
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: public class MobilePhone extends Phone implements AudioMessageEnabled,
TextMessageEnabled {
10:     MobilePhone(String phone_number) {
11:         super(phone_number);
12:     }
13:
14:     void sendMessage(String msg) {
15:         System.out.println("message: " + msg);
16:         System.out.println(("Phone number: " + this.info));
17:     }
18:
19:     public void sendTextMessage(String msg) {
20:         System.out.println("msg: " + msg);
21:         System.out.println("email: " + this.info);
22:     }
23: }
```

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 09:59
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: public class Email extends ContactInfo implements TextMessageEnabled {
10:
11:     Email(String email) {
12:         super(email);
13:     }
14:
15:     String contactInfo() {
16:         return this.info;
17:     }
18:
19:     String contactInfoType() {
20:         return "Email";
21:     }
22:
23:     void sendMessage(String msg) {
24:         System.out.println("msg: " + msg);
25:         System.out.println("email: " + this.info);
26:     }
27:
28:     public void sendTextMessage(String msg) {
29:         System.out.println("msg: " + msg);
30:         System.out.println("email: " + this.info);
31:     }
32: }
```

```
1: package contacts;
2:
3: import java.util.ArrayList;
4: import java.util.List;
5:
6: /**
7:  * @time 25/01/2024 10:17
8:  * @version 1.0
9:  * @author fsh
10:  * @description:
11:  */
12: public class ContactManager {
13:     List<Contact> list = new ArrayList<Contact>();
14:
15:     void add(Person p, ContactInfo info) {
16:         list.add(new Contact(p, info));
17:     }
18:
19:     List<ContactInfo> contactDetails(Person p) {
20:         List<ContactInfo> ret = new ArrayList<ContactInfo>();
21:         for (Contact c : list) {
22:             if (c.p.name().equals(p.name())) {
23:                 ret.add(c.info);
24:             }
25:         }
26:         return ret;
27:     }
28:
29:     void spam(String msg) {
30:         for (Contact c : list) {
31:             c.info.sendMessage(msg);
32:         }
33:     }
34: }
```

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 09:54
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: abstract class ContactInfo {
10:     String info;
11:
12:     ContactInfo(String info) {
13:         this.info = info;
14:     }
15:
16:     abstract String contactInfo();
17:
18:     abstract String contactInfoType();
19:
20:     abstract void sendMessage(String msg);
21: }
```

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 09:57
5:  * @version 1.0
6:  * @author fsh
7:  * @description: contact that combines person with contactInfo
8:  */
9: public class Contact {
10:     Person p;
11:     ContactInfo info;
12:
13:     Contact(Person p, ContactInfo info) {
14:         this.p = p;
15:         this.info = info;
16:     }
17: }
```


Final Tests

AudioMessageEnabled.java: 1/1

Shihan Fu - sf23:v5

```
1: package contacts;
2:
3: /**
4:  * @time 25/01/2024 10:07
5:  * @version 1.0
6:  * @author fsh
7:  * @description:
8:  */
9: interface AudioMessageEnabled {
10:     void sendAudioMessage(Audio msg);
11: }
```

Final Tests**testResults.txt: 1/1****Shihan Fu - sf23:v5**

```
1: ----- Test Output -----
2: Running LabTS build...
3:
4: Submission summary...
5: You made 2 commits
6:   - 7cf0943 version 1 basic implementation [8 files changed, 168 insertions]
7:   - 4de6f78 version 2 fix phone and mobilephone sendMessage [8 files changed, 88 insertions, 75 deletions]
8:
9: Preparing...
10:
11: BUILD SUCCESSFUL in 641ms
12:
13: Compiling...
14: BUILD SUCCESSFUL in 2s
15:
16: Running tests...
17:
18: contacts.MobilePhoneTest > sendMessage PASSED
19:
20: contacts.EmailTest > sendMessage PASSED
21:
22: contacts.PhoneTest > sendMessage PASSED
23:
24: contacts.ContactManagerTest > spamContacts PASSED
25:
26: contacts.ContactManagerTest > listsContactInfo PASSED
27:
28: BUILD SUCCESSFUL in 1s
29:
30: Checking code style...
31:
32:
33: ----- Test Errors -----
34: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/AudioMessageEnabled.java:3: Summary javadoc is missing. [SummaryJavadoc]
35: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Contact.java:3: Summary javadoc is missing. [SummaryJavadoc]
36: [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]*$'. [MemberName]
37: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/ContactInfo.java:3: Summary javadoc is missing. [SummaryJavadoc]
38: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/ContactManager.java:6: Summary javadoc is missing. [SummaryJavadoc]
39: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Email.java:3: Summary javadoc is missing. [SummaryJavadoc]
40: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/MobilePhone.java:3: Summary javadoc is missing. [SummaryJavadoc]
41: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/MobilePhone.java:10:22: Parameter name 'phone_number' must match pattern '^[a-z]([a-z0-9][a-zA-Z0-9]*)?$'. ↗
[ParameterName]
42: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/Phone.java:3: Summary javadoc is missing. [SummaryJavadoc]
43: [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]
44: [ant:checkstyle] [WARN] /tmp/d20240125-38-ama9j5/src/contacts/TextMessageEnabled.java:3: Summary javadoc is missing. [SummaryJavadoc]
45:
46: FAILURE: Build failed with an exception.
47:
48: * What went wrong:
49: Execution failed for task ':checkstyleMain'.
50: > Checkstyle rule violations were found. See the report at: file:///tmp/d20240125-38-ama9j5/build/reports/checkstyle/main.html
51:   Checkstyle files with violations: 8
52:   Checkstyle violations by severity: [warning:11]
53:
54:
55: * Try:
56: 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.
57:
58: * Get more help at https://help.gradle.org
59:
60: 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