### Circle the lecture section in which you are enrolled

| L0101 (WF12) | L0201 (WF1) | L5101 (W6) |

### Please indicate your answers in the table below

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td>(F)</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td>(F)</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>(A)</td>
<td>(B)</td>
<td>(C)</td>
<td>(D)</td>
<td>(E)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Nothing, does not run</td>
<td>true</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Nothing, does not run</td>
<td>true</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Nothing, does not run</td>
<td>true</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Yes</td>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 13. | x searchUser(acc1);  
    x searchUser(acc2);  
    x searchUser(acc3);  
    x searchUser((UserAccount) acc3); |
| 14. | □ searchUser(acc1);  
    □ searchUser(acc2);  
    x searchUser(acc3);  
    □ searchUser((TemporaryUserAccount) acc1); |
| 15. | (a) | (b) | (c) |
| 16. | (a) | (b) | (c) | (d) |
Consider a main method in a class that is included in the same package as class UserAccount and class TemporaryUserAccount from the Supplementary Code at the back of this booklet. Assume that the main method contains the following code:

UserAccount acc1 = new UserAccount("Anika", "acc1@domain.com");
UserAccount acc2 = new TemporaryUserAccount("Bojan", "acc2@domain.com");
TemporaryUserAccount acc3 = new TemporaryUserAccount("Chen", "acc3@domain.com");
UserAccount acc4 = acc1;
TemporaryUserAccount acc5 = (TemporaryUserAccount) acc2;
UserAccount acc6 = new UserAccount("Anika", "acc1@domain.com");

Part I.

1. What does System.out.println(acc2.userName) print?
   
   (a) Anika  
   (b) Bojan  
   (c) Chen  
   (d) null  
   (e) none of the above

2. What does System.out.println(acc1.getUserName()) print?
   
   (a) Anika  
   (b) Bojan  
   (c) Chen  
   (d) null  
   (e) none of the above

3. What does System.out.println(acc2.getUserName()); print?
   
   (a) Anika  
   (b) Bojan  
   (c) Chen  
   (d) null  
   (e) none of the above

4. What does System.out.println(acc3.toString()) print?
   
   (a) Temporary Account 0 belongs to: Chen  
   (b) Temporary Account 1 belongs to: Chen  
   (c) Temporary Account 3 belongs to: Chen  
   (d) Temporary Account 0 belongs to: null  
   (e) Temporary Account 1 belongs to: null  
   (f) Temporary Account 3 belongs to: null
5. What does `System.out.println(acc2.toString())` print?

(a) User Name: Bojan, Account Number: 1
(b) User Name: Bojan, Account Number: 0
(c) User Name: null, Account Number: 2
(d) Temporary Account 0 belongs to: Bojan
(e) Temporary Account 2 belongs to: Bojan
(f) Temporary Account 1 belongs to: null

6. What does `acc3.getEmail()` return?

(a) nothing because `acc3` does not have a method called `getEmail`
(b) "acc3@domain.com"
(c) nothing because `acc3` does not have a variable called `email`
(d) null
(e) More than one of the above answers

Part II.

7. What does `System.out.println(acc1 == acc4)` print?
Nothing, because the code does not run / true / false

8. What does `System.out.println(acc5.equals(acc2))` print?
Nothing, because the code does not run / true / false

9. What does `System.out.println(acc1.equals(acc6))` print?
Nothing, because the code does not run / true / false

In the following questions, o1 and o2 are different if `o1.equals(o2)` returns false.

10. Is it possible for the same user name to be attached to two different UserAccounts?
Yes / No

11. Is it possible for the same user name to be attached to two different TemporaryUserAccounts?
Yes / No

12. Is it possible for the same account number to be attached to two different TemporaryUserAccounts?
Yes / No
Part III.

For questions 13, 14, and 15, let us assume that we have defined the `searchUser()` and `searchTempUser()` methods in a class other than those in the Supplementary Code. Here are their signatures:

```java
public boolean searchUser(UserAccount ua);

public boolean searchTempUser(TemporaryUserAccount tua);
```

13. Check the box(es) of the function call(s) that will compile:

- searchUser(acc1)
- searchUser(acc2)
- searchUser(acc3)
- searchUser((UserAccount) acc3)

14. Check the box(es) of the function call(s) that will compile:

- searchTempUser(acc1)
- searchTempUser(acc2)
- searchTempUser(acc3)
- searchTempUser((TemporaryUserAccount) acc1)

15. Which of the following does not happen upon execution of:

   `UserAccount newAccount = new UserAccount("Ada", "ada@domain.com");`

   (a) `nextAccountNum` in class `UserAccount` is incremented by one.
   (b) `nextAccountNum` in class `TemporaryUserAccount` is incremented by one.
   (c) the value of `nextAccountNum` is stored in an instance variable before being incremented.

16. Which of the following has lookup rules that are different from the others?

   (a) instance variables
   (b) instance methods
   (c) static variables
   (d) static methods
class UserAccount {
    public String userName; // notice the access modifier
    private String email;
    private int accountNum;
    public static int nextAccountNum = 1;

    public UserAccount(String userName, String email) {
        this.userName = userName;
        this.email = email;
        accountNum = nextAccountNum;
        nextAccountNum++;
    }

    protected UserAccount(String email) {
        this.email = email;
        this.accountNum = 0;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String newName) {
        userName = newName;
    }

    public int getAccountNum() {
        return accountNum;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public static int getNextAccountNum() {
        return nextAccountNum;
    }

    public boolean equals(UserAccount ua) {
        return this.accountNum == ua.accountNum;
    }

    public String toString() {
        return "User Name: " + userName + ", Account Number: " + accountNum;
    }
}

public class TemporaryUserAccount extends UserAccount {

    private String userName;
    private int accountNum;
    public static int nextAccountNum = 1;

    public TemporaryUserAccount(String userName, String email) {
        super(email);
        this.userName = userName;
    }

    public boolean equals(UserAccount ua) {
        if (!(ua instanceof TemporaryUserAccount)) {
            return false;
        } else {
            if (((UserAccount) ua).equals(
                ((UserAccount) this).getUserName()) &&
                (((UserAccount) ua).getAccountNum()) ==
                (((UserAccount) this).getAccountNum())) {
                return true;
            }
        }
        return false;
    }

    public String toString() {
        return "Temporary Account " + accountNum + " belongs to: " + userName;
    }
}