EECS 1022 3.0 Programming for Mobile Computing

Solution of Midterm - Version A

18:30–19:30 on July 10, 2017

1 (2 marks)

(a) The process of software development consists of multiple phases. Which artefact is produced in the design phase?

Answer: plan or algorithm.

Marking scheme: 1 mark for either plan or algorithm.

(b) Which phase follows the design phase?

Answer: implementation.

Marking scheme: 1 mark for implementation.

2 (2 marks)

The design pattern MVC separates the code into a model, a view, and a controller. Assume you develop a mobile app that converts the temperature in Fahrenheit to the corresponding temperature in Celcius. For each of the following items, indicate whether it belongs to the model, the view or the controller.

(a) The temperature in Fahrenheit.

Answer: model.

Marking scheme: 0.5 mark for model.

(b) The constant 32 that is used to convert Fahrenheit to Celcius.

Answer: model.

Marking scheme: 0.5 mark for model.

(c) The method `buttonClicked`.

Answer: controller or activity.

Marking scheme: 0.5 mark for controller or activity.

(d) The submit button.

Answer: view.

Marking scheme: 0.5 mark for view.

3 (2 marks)

Consider the API of the class `Person` which is provided at the end of this test.

(a) How many (public) constructors does the class have?

Answer: 2

Marking scheme: 1 mark for 2.

(b) What is the return type of the `getAge` method?

Answer: int.

Marking scheme: 1 mark for int.

4 (2 marks)

Can two objects have the same state but different identities? **Explain your answer.** You only get marks for your explanation.

Answer: Yes. For example, consider

```
Person one = new Person("First", "Last", 1);
Person another = new Person("First", "Last", 1);
```

The objects created above, referenced by `one` and `another`, have that same state since the attributes have the same values, but have different identities as the objects reside at different memory addresses.

Marking scheme: 1 mark for mentioning that there can be two objects whose *attributes* have the same *values*. 1 mark for mentioning that these objects reside on different *memory addresses*.

5 (2 marks)

(a) Consider the following code snippet.

```
Person john = new Person("John", "Doe", 32);
Person jane = new Person("Jane", "Doe", 34);
Person copy = jane;
```

Draw the corresponding memory diagram. Make sure that the attributes `firstName`, `lastName` and `age` of the `Person` class and the variables `john`, `jane` and `copy` are reflected in your diagram.

Answer:

0		
⋮		
100	*300*	john
	400	jane
	400	copy
⋮		
200	Person class	
⋮		
300	Person object	
	”John”	firstName
	”Doe”	lastName
	32	age
⋮		
400	Person object	
	”Jane”	firstName
	”Doe”	lastName
	34	age
⋮		

”John”, ”Jane” and ”Doe” are String objects and, hence, should be represented by object blocks as well, but for simplicity we have not done that here.

Marking scheme:

- 0.25 mark for two Person object blocks.
- 0.25 mark for the values of john and jane are the addresses of the two objects.
- 0.25 mark for the values of jane and copy be the same.
- 0.25 mark for the correct values of the attributes of the Person objects.

(b) The method `hasSameAge` is implemented as follows.

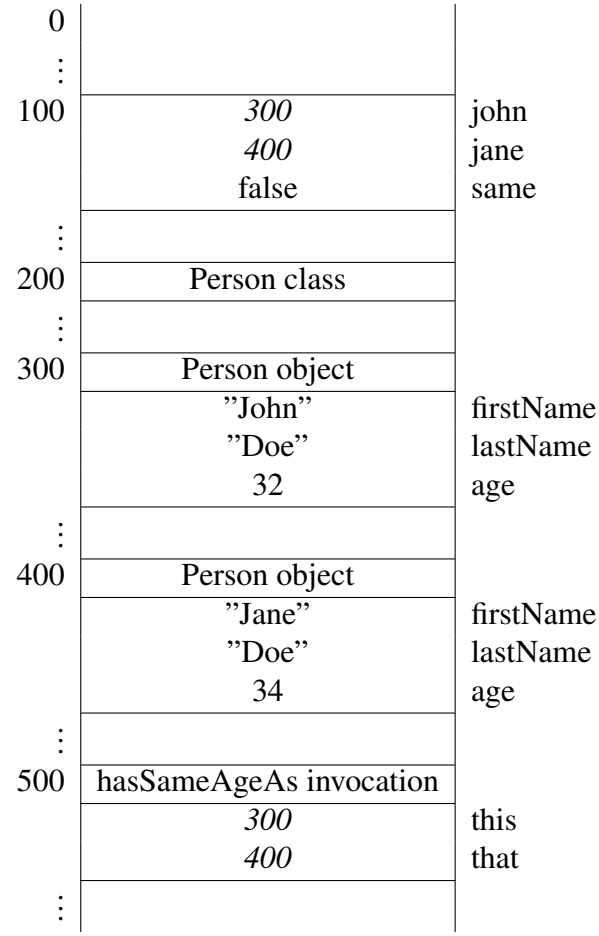
```
public boolean hasSameAgeAs(Person that)
{
  return this.age == that.age;
}
```

Consider the following code snippet.

```
Person john = new Person("John", "Doe", 32);
Person jane = new Person("Jane", "Doe", 34);
boolean same = jonh.hasSameAgeAs(jane);
```

Draw the corresponding memory diagram. Make sure that the attributes `firstName`, `lastName` and `age` and the variables `john`, `jane` and `same` are reflected in your diagram. Make sure to include the invocation block for the call of the `hasSameAgeAs` method.

Answer:



”John”, ”Jane” and ”Doe” are String objects and, hence, should be represented by object blocks as well, but for simplicity we have not done that here.

Marking scheme:

- 0.2 mark for this in the invocation block.
- 0.2 mark for that in the invocation block.
- 0.2 mark for the value of this (address of the Person object referred to by john) in the invocation block

- 0.2 mark for the value of that (address of the Person object referred to by jane) in the invocation block
- 0.2 mark for the value of same.