# Sample exam 3

The INFDEV team

# 1 Question 1

Given the following block of code, fill in the stack, heap, and PC with all the steps taken by the program at runtime.

- Points: *4 (50% of total).*
- Grading: one point per correctly filled-in execution step.
- Associated learning objective: *abstraction.*

```
1  interface MovableObject {
2  }
3  class Car : MovableObject {
4    private float direction = 3.14;
5    public Car() {
6    }
7    static public void move(Car car,float direction) {
8      car.direction = direction;
9    }
10 }
11 class Particle : MovableObject {
12   private float direction = 0;
13   public Particle() {
14   }
15 }
16 MovableObject mo = new Car();
17 Car.move(mo,1);
```

1. Stack:

| PC |
|---|
| 1 |

2. Stack:

| PC |
|---|
| 16 |

3. Stack:

| PC |
|---|
| 16 |

Heap:

| 1 |
|---|
| __type=Car<br>direction= |

4. Stack:

| PC | ... | | PC | ret | this |
|---|---|---|---|---|---|
| 16 | ... | | 6 | null | ref 1 |

Heap:

| 1 |
|---|
| __type=Car<br>direction= |

5. Stack:

| PC | ... | | PC | ret |
|---|---|---|---|---|
| 16 | ... | | 6 | ref 1 |

Heap:

| 1 |
|---|
| __type=Car<br>direction= |

6. Stack:

| PC | mo |
|---|---|
| 17 | ref 1 |

Heap:

| 1 |
|---|
| __type=Car<br>direction= |

7. Stack:

| PC | ... | | PC | ret | car | direction |
|---|---|---|---|---|---|---|
| 17 | ... | | 8 | null | ref 1 | 1 |

Heap:

| 1 |
|---|
| __type=Car<br>direction= |

8. Stack:

| PC | ... | | PC | ret |
|---|---|---|---|---|
| 17 | ... | | 8 | null |

Heap:

| 1 |
|---|
| __type=Car<br>direction=1 |

9. Stack:

| PC | mo |
|---|---|
| 18 | ref 1 |

Heap:

| 1 |
|---|
| __type=Car<br>direction=1 |



## 2 Question 2

Given the following block of code, fill in the declarations, class definitions, and PC with all steps taken by the compiler while type checking.

- Points: *4 (50% of total).*
- Grading: one point per correctly filled-in type checking step.
- Associated learning objective: *type checking.*

```
1  interface IntList {
2    bool isEmpty();
3    int getValue();
4  }
5  class IntNode : IntList {
6    private int value;
7    private IntList tail;
```

```
8   public IntNode(int value,IntList tail) {
9     this.value = value;
10    this.tail = tail;
11  }
12  public bool isEmpty() {
13    return false;
14  }
15  public int getValue() {
16    return this.value;
17  }
18 }
19 class IntEmpty : IntList {
20  public IntEmpty() {
21  }
22  public bool isEmpty() {
23    return true;
24  }
25  public int getValue() {
26    return 0;
27  }
28 }
29 ...
30 IntList list = new IntNode(5,new IntEmpty());
```



1. Declarations:

| PC |
|---|
| 1 |

2. Declarations:

| PC |
|---|
| 5 |

Classes:

| IntList |
|---|
| getValue=IntList → int<br>isEmpty=IntList → bool |

3. Declarations:

| PC |
|---|
| 18 |

Classes:

| IntList | IntNode |
|---|---|
| getValue=IntList → int<br>isEmpty=IntList → bool | IntNode=(IntNode×int×IntList) → IntNode<br>getValue=IntList → int<br>isEmpty=IntList → bool<br>tail=IntList<br>value=int |

4. Declarations:

| PC |
|---|
| 28 |

Classes:

| IntEmpty | IntList | IntNode |
|---|---|---|
| IntEmpty=IntEmpty → IntEmpty<br>getValue=IntList → int<br>isEmpty=IntList → bool | getValue=IntList → int<br>isEmpty=IntList → bool | IntNode=(IntNode×int×IntList) → IntNode<br>getValue=IntList → int<br>isEmpty=IntList → bool<br>tail=IntList<br>value=int |

5. Declarations:

| PC |
|---|
| 30 |

Classes:

| IntEmpty | IntList | IntNode |
|---|---|---|
| IntEmpty=IntEmpty → IntEmpty<br>getValue=IntList → int<br>isEmpty=IntList → bool | getValue=IntList → int<br>isEmpty=IntList → bool | IntNode=(IntNode×int×IntList) → IntNode<br>getValue=IntList → int<br>isEmpty=IntList → bool<br>tail=IntList<br>value=int |

6. Declarations:

| PC | list |
|---|---|
| 31 | IntList |

Classes:

| IntEmpty | IntList | IntNode |
|---|---|---|
| IntEmpty=IntEmpty → IntEmpty<br>getValue=IntList → int<br>isEmpty=IntList → bool | getValue=IntList → int<br>isEmpty=IntList → bool | IntNode=(IntNode×int×IntList) → IntNode<br>getValue=IntList → int<br>isEmpty=IntList → bool<br>tail=IntList<br>value=int |