1 What is the return type of getNameOfMonth?

```
public class Date {
   private int month;
   public int getMonth() { return month; }

   public          getNameOfMonth() {
   if (getMonth() == 0) {
     return "January";
   } else if (getMonth() == 1) {
     return "February";
   }
   // ...
 }}
```

a. Date
b. Month
c. String
d. int

2 What is the type of this expression?

```
int x = 0;
boolean b = false;
double d = 2.0;

(x > 24) || (!b && d + 23 == 15.0)
```

a. int
b. float
c. double
d. boolean

3 What sort of variable is *day*

```
public class Date {
  private int day;
  private int month;
  private int year;
  // ...
  public int getDay()   { return day; }
  public int getMonth() { return month; }
  public int getYear()  { return year; }
  // …
}
```

a. Local variable
b. Parameter
c. Instance variable
d. Constant

4 What is the return value of *foo*?

```
public int foo() {
    int x = 23;
    int y = 34;
    return (x++) + (++y);
}
```

a. 57
b. 56
c. 58
d. 59

5 What will be printed on the screen?

```
public static void main(String [] args) {
   String obj1 = "xyz";
   String obj2 = "x" + "y" + "z";
   if(obj1.equals(obj2)) {
      System.out.println("obj1 == obj2 is TRUE");
   } else {
      System.out.println("obj1 == obj2 is FALSE");
   }
}
```

a. Compilation error
b. obj1 == obj2 is TRUE
c. Runtime error
d. obj1 == obj2 is FALSE

6 What will be printed on the screen?

```
public static void printResult(int x, int y) {
    if (y / x < 1 && x != 0) {
        System.out.println("Expression is true");
    } else {
        System.out.println("Expression is false");
    }
}

public static void main(String[] args) {
    printResult(0,  1);
}
```

a. "Expression is true”
b. An error message
c. "Expression is false”
d. Nothing

7 What is a correct specification for *power*?

```
public int power(int e1, int e2) {
    int res = 1;
    int count = 0;
    if (e1 > 0) {
        while (count < e2) {
            res = res * e1;
            count = count + 1;
        }
        return res;
    } else {
        return 1;
    }}
```

a. ensures \result >= 1;

b. ensures \result == e1 + e2;

c. invariant res >= 0;

d. requires e1 == e2;

8 What is wrong?

```
/*@  requires n > 0;
     ensures n > 0;
*/
public void compute(int n) {
   // … method body
}
```

a. Nothing
b. Postcondition
c. n > 0 should be invariant
d. Precondition

9 What will happen?

```
public class Counter {
    public static final int MAX;
    private int count;

    public void incCount() {
        count = (count + 1) % MAX;
    }
    public void setMax(int n) {
        MAX = n;
    }
}
```

a. Run-time error in setMax
b. Run-time error in incCount
c. Compiler error
d. No problem

10 Which implementation does not respect this specification?

```
public int x;

/*@  requires n >= 0;
     ensures x >= \old(x);
*/
public void inc(n) {
   // body goes here
}
```

a. x = n;

b. x = x + n;

c. x = x + 1;

d. x = x + 2*n;

11 What will happen?

```
public class Counter {
    private int counter;
    public void incCounter() { // .. }
}
public class Foo {
    public static void main (String [] args) {
        Counter c = new Counter();
        c.incCounter();
        c.incCounter();
        c.incCounter();
        System.out.println(c.counter);
    }}
```

a. 4
b. 3
c. Run-time error
d. Compiler error

12 Which invariant is wrong?

```
public class Lamp {
    public static final int OFF = 0;
    public static final int ON = 1;

    private int state = OFF;

    public static final int MAX = 2;
    //@ private invariant <invariant>

    public int getState() { return state; }
    public void switchState() { state = (state + 1) % MAX; }
}
```

a. OFF <= state
b. state < 7
c. true
d. state < MAX – 1;

13 What will be printed on the screen?

```
public class Room {
    private int number;
    private Guest guest;

    public Room (int n, Guest g) { number = n; guest = g;}

    // more methods in Room

    public static void main (String [] args) {
        Room r1 = new Room(101, null);
        Room r2 = new Room(101, null);
        System.out.println(r1.equals(r2)); }}
```

a. True
b. Depends on equals body
c. False
d. Error message

14 Which postcondition is not possible for *bla*?

```
//@ ensures <postcondition>;
public int bla(int n) {
    // body of bla
}
```

a. \result;

b. \result > n;

c. \result == 3;

d. (\forall int i; 0 <= i && i < \result;
f(i));

15

What is not a possible result value of *getRoom(4).getGuest().getName();*

```
public class Guest {
    private String name;
    public String getName() { return name;}
}
public class Room {
    private int number;
    private Guest guest;
    public Guest getGuest() { return guest;}
}
public class Hotel {
    public Room getRoom(int number) { // ..}
}
```

a. Null
b. Run-time error
c. “Marieke”
d. ‘x’

16 What will happen?

```
public class Item {
    public static int created = 0;

    public Item() { created = created + 1; }

    public static void main (String [] args) {
        Item i1 = new Item();
        Item i2 = new Item();
        System.out.println(Item.created);
    }
}
```

a. Print 2
b. Error: created should be private
c. Error: created belongs to object
d. Error: integer cannot be printed

17 What is a correct invariant for *counter?*

```
public class Counter {
   //@ public invariant <invariant>
   public int count;
   public static final int max = 40;

   public int getCount() {
      return count;
   }

   public void incCount() {
      count = count + 1;
   }}
```

a. 0 <= count && count < 40;

b. true;

c. 0 <= count && count <= 39;

d. 0 <= count && count < 41;

18 What is the final value of *state?*

```
public enum State {On, Off, Error}

    State state = Off;
    boolean flag = true;
    switch (state) {
        case : Off : state = On;
        case : On : state = flag? Error : Off;
        case default:
    }
```

a. Run-time error
b. On
c. Error
d. Off

19 What is the return value of this *foo*?

```
public String foo() {
    boolean x = false;
    if (x = false) {
        return “bla”;
    } else {
        return “blob”;
    }
}
```

a. “bla”
b. “blob”
c. Compiler error
d. Run-time error

20 How many test cases for method *open*?

```
public enum State {Open, Closed}
public class Lock {
    private State state;
    private int code;
    //@ private invariant 0 <= this.code && this.code <= 99;

    //@ requires 0 <= input && input <= 99;
    public void open(int input) {
        // ..
    }
```

a. For all possible integers
b. 2 x 100 x 100
c. 100 x 100
d. 2 x 100

21 What is the result type of *getRoom(4).getGuest();*

```
public class Guest {
    private String name;
    public String getName() { return name;}
}
public class Room {
    private int number;
    public Guest getGuest() { return guest;}
}
public class Hotel {
    public Room getRoom(int number) { // body of getRoom}
}
```

a. String
b. Guest
c. Room
d. int

22 What will happen?

```
public class Counter {
    public static int MAX;
    private int count;

    public void incCount() {
        count = (count + 1) % MAX;
    }
    public void setMax(int n) {
        MAX = n;
    }
}
```

a. Run-time error in setMax
b. Run-time error in incCount
c. Compiler error
d. No problem

23 What will be printed on the screen?

```
public static void main(String [] args) {
   String obj1 = new String("xyz");
   String obj2 = new String("xyz");
   if(obj1.equals(obj2)) {
      System.out.println("obj1 == obj2 is TRUE");
   } else {
      System.out.println("obj1 == obj2 is FALSE");
   }
}
```

a. Compilation error
b. obj1 == obj2 is TRUE
c. Runtime error
d. obj1 == obj2 is FALSE

24 What is the return value of *foo*?

```
public int foo(int n) {
    int x = 568;
    x = (x++) + n;
    return x;
}
```

a. 568
b. 569 + n
c. 568 + n
d. n

25 What will happen?

```
public class Counter {
    private int Counter;
    public int getCounter() {return counter;}
    public void incCounter() { // .. }

    public static void main (String [] args) {
        Counter c = new Counter();
        System.out.println(Counter.getCounter());
    }}
```

a. 0
b. 1
c. Compiler error
d. Run-time error

26 What will happen?

```
public class Box {
    private int height;
    private int length;

    public Box (int h, int l) {
        height = h;
        length = l;
    }

    public static void main (String [] args) {
        Box b = new Box(“42”, “3”);
    }}
```

a. Compiler error
b. Run-time error
c. Box with height = 42, length = 3
d. Box with height = 2, length = 1

27 What will happen?

```
public class Point {
    private int x;
    private int y;
    public void move(int n) { x = x + n; }
    public void move(int n, int m) { x = x + n; y = y + m; }
    public int getY() {return y;}

    public static void main (String [] args) {
        Point p = new Point();
        p.move(3); p.move(4, 5);
        System.out.println(p.getY());
    }}
```

a. 3
b. 7
c. 5
d. Compiler error

28 What will be printed on the screen?

```
public class Room {
    private int number;
    private Guest guest;

    public Room (int n, Guest g) { number = n; guest = g;}

    // more methods in Room

    public static void main (String [] args) {
        Room r1 = new Room(101, null);
        Room r2 = new Room(101, null);
        System.out.println(r1 == r2); }}
```

a. True
b. Depends on equal body
c. False
d. Error message

29 What will happen?

```
public class Print {
  public static int a;

  public static void main(String[] args) {
    int b;
    System.out.println("a : "+a);
    System.out.println("b : "+b);
    }
}
```

a. Compiler error
b. Run-time error
c. Nothing
d. “a : 0” and “b : 0” will be printed

30 What is the final value of *state?*

```
public enum State {On, Off, Error}

    State state = Off;
    boolean flag = true;
    switch (state) {
        case : Off : state = On; break;
        case : On : state = flag? Error : Off; break;
        case default:
    }
```

a. Run-time error
b. On
c. Error
d. Off

31 What will happen?

```
public class Counter {
    private static int counter;
    public static int getCounter() {return counter;}
    public void incCounter() { // .. }

    public static void main (String [] args) {
        Counter c = new Counter();
        System.out.println(Counter.getCounter());
    }}
```

a. 0
b. 1
c. Compiler error
d. Run-time error

32 What specification is checked by these asserts?

```
public int compute (int m, int n) {
    assert m >= 0;
    assert n >= 0;
    return <complicated expression using m and n>
}
```

a. requires m >= 0 && n >= 0;

b. ensures m >= 0 && n >= 0;

c. requires m >= 0 || n >= 0;

d. ensures m >= 0 || n >= 0;

33 What will be printed on the screen?

```
public static void main(String [] args) {
   String obj1 = new String("xyz");
   String obj2 = new String("xyz");
   if(obj1 == obj2) {
      System.out.println("obj1==obj2 is TRUE");
   } else {
      System.out.println("obj1==obj2 is FALSE");
   }
```

a. Compilation error
b. obj1 == obj2 is TRUE
c. Runtime error
d. obj1 == obj2 is FALSE

34 What will happen?

```
int x = 29/3;

System.out.println(x);
```

a. Compiler error
b. 9 printed
c. 9.66666666… printed
d. Run-time erro

35 What will happen?

```
public class Lamp {
    public static final int OFF = 0;
    public static final int ON = 1;

    private int state = OFF;

    public int getState() { return state; }
    public void switchState() { state = state + 1; }
    public static void main (String [] args) {
        Lamp l = new Lamp();
        l.switchState(); l.switchState();
        System.out.println(l.getState());   }}
```

a. Compiler error
b. Run-time error
c. 0
d. 2

36 What will happen?

```
public class Room {
    private int number;
    private String name;
    public String getGuest() { return name;}
    public void setGuest(String name) {this.name = name;}

    public static void main (String[] args) {
        Room r1 = new Room();
        Room r2 = r1;
        r1.setGuest("Klaas");
        System.out.println(r2.getGuest());
}}
```

a. Run-time error
b. Null will be printed
c. "Klaas" will be printed
d. Compiler error

37 What will be printed on the screen?

```
public static void printResult(int x, int y) {
    if (y != 0 && x/ y < 1) {
        System.out.println("Expression is true");
    } else {
        System.out.println("Expression is false");
    }
}

public static void main(String[] args) {
    printResult(1,  2);
}
```

a. Nothing
b. “Expression is true”
c. “Expression is false”
d. An error message

38 What is the return value of *foo*?

```
public int foo() {
    int x = 18;
    return (x++) + (++x);
}
```

a. 36
b. 37
c. 38
d. 39

39 What will happen?

```
public class Room {
    private int number;
    private String name;
    public String getGuest() { return name;}
    public void setGuest(String name) {this.name = name;}

    public static void main (String[] args) {
        Room r = new Room();
        System.out.println(r.getGuest());
}}
```

a. Run-time error
b. Null will be printed
c. “” will be printed
d. Compiler error

40 What will be printed on the screen?

```
public class Counter {
    private int count;
    public Counter (int count) {
        count = count;
    }
    public int getCount() {
        return count;
    }
    public static void main (String [] args) {
        Counter c = new Counter(4);
        System.out.println(c.getCount());
    }}
```

a. Undefined
b. 2
c. 4
d. 0

41 How many test cases for method *open*?

```
public enum State {Open, Closed}
public class Lock {
    private State state;
    private int code;
    //@ private invariant 0 <= this.code && this.code <= 99;

    //@ requires 0 <= input;
    public void open(int input) {
        // ..
    }
```

a. 2 x 100 x all positive integers
b. 2 x 100 x 100
c. 100 x 100
d. 2

42 What specification is captured by the assert in *foo*?

```
public enum State {Normal, Error}

public class Bla {
   private State state = Normal;

   public void foo(boolean flag) {
      if (flag) {
         state = Error;
      } else {
         // do some computation
      }
      assert !flag || state == Error; }}
```

a. requires \old(state) == state
b. ensures flag ==> state == Error
c. ensures state == Error
d. ensures flag

43 What problem will the compiler detect?

```
public class Item {
    public int x;

    private void update(int n, boolean flag) {
        if (flag) {
            n = x * n;
        }
        return x == n;
    }
}
```

a. Public instance variable
b. Private method
c. Result type of update
d. Method changes parameter

44 What will happen?

```
public class Puppy{
   public void pupAge(){
      int age;
      age = age + 7;
      System.out.println("Puppy age is : " + age);
   }

   public static void main(String args[]){
      Puppy test = new Puppy();
      test.pupAge();
   }
}
```

a. Compiler error
b. Print 0
c. Print 7
d. Run-time error

45 What is the result of method *foo?*

```
public int foo() {
    int x = 16;
    x= (++ x + 3) + x++;
    return x;
}
```

a. Run-time error
b. 16
c. 38
d. 37

46 What is a good specification for the constructor?

```
public class Point {
    private int x;
    private int y;

    public Point(int n, int m) {
        x = n;
        y = m;
    }
    public int getX() { return x; }
    public int getY() { return y; }
}
```

a. requires x == 0 && y == 0;

b. requires x < n && y < m;

c. ensures x == n && y == m;

d. ensures x < n && y < m;

47

What is a possible result value of *getRoom(4).getGuest().getName();*

```
public class Guest {
    //@ private invariant name.length < 13;
    private String name;
    public String getName() { return name;}
}
public class Room {
    private int number;
    public Guest getGuest() { return guest;}
}
public class Hotel {
    public Room getRoom(int number) { // ..;}
}
```

a. “Marieke”
b. “prof. dr. Marieke Huisman”
c. ‘x’
d. 13

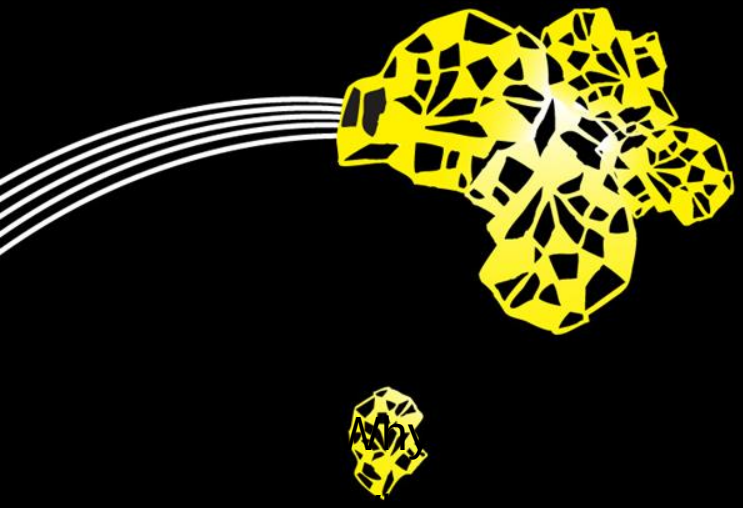
48 What will happen?

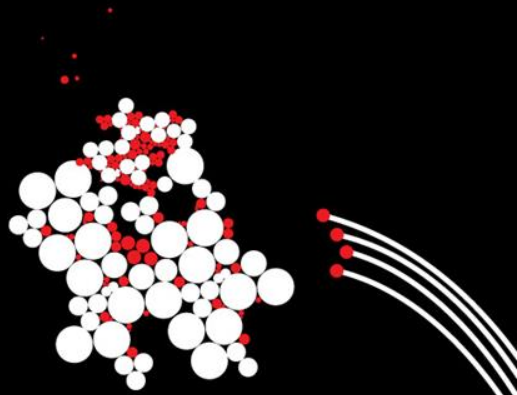
```
public class Guest {
    private String name;
    public String getName() { return name; }
}
public class Room {
    private Guest guest;
    public Guest getGuest() { return guest;}

    public static void main (String[] args) {
        Room r = new Room();
        System.out.println(r.getGuest().getName());
}}
```

a. Run-time error
b. Null will be printed
c. “” will be printed
d. Compiler error

UNIVERSITY OF TWENTE.





LAST MAN STANDING

DIAGNOSTIC TEST 1

RULES OF THE GAME

- Register yourself with your student number at http://130.89.12.249
- Initially everybody is standing
- Answer the question within 45 seconds by selecting an answer option on your screen
- If you answer the question correctly, you remain standing
- If you answer wrong, you sit down, but you keep on answering questions
- If there is only 1 person standing, we have a winner
- If we have a winner, or if all participants answered wrong, we start with the complete group again.
- There are prizes for the winners, and for the best overall scores