Systems Programming
Final exam (extra call)

Leganés, June 28, 2018 — Final exam (extra call): test
Duration: 40 minutes — Score: 3 points out of 10 in the exam

There is only one correct option for each question. Each correct answer adds 0.15 points. Each incorrect answer subtracts 0.05 points. Unanswered questions do not add or subtract points.

- Mark the answer to each question with an "X" in the table below.
- If there is no "X" in a question or you mark more than one option, the question is considered unanswered.
- Complete **your personal information** before starting the exam.

Name:	Group:

Signature:

NIA:

	A	B	C	D		A	B	C	D
1					11				
2					12				
3					13				
4					14				
5					15				
6					16				
7					17				
8					18				
9					19				
10					20				

1.- In the code structure provided below, what should be added in Line 03 instead of the comment marked with `/* TO DO */`?

```
01 public class Point{
02         private double x, y;
03         public Point(double x, double y) {  /* TO DO */ }
04         public Point() {  /* TO DO */ }
05         public double getX() {  /* TO DO */ }
06         public void setX(double x){  /* TO DO */ }
07         public double getY() {  /* TO DO */ }
08         public void setY(double y){  /* TO DO */ }
09         public double distance() {/* TO DO */ }
10         public double distance(Point otroPunto) {/* TO DO */ }
11  }
```

(a) *** `this.x = x; this.y = y;`

(b) `x = this.x; y = this.y;`

(c) The names of the arguments in the constructor must be different of those in the attributes of the class. Otherwise, the constructor cannot be implemented.

(d) It is not necessary to include code, the arguments already indicate the values that `x` and `y` will have.

2.- We define the classes `A` and `B` in the same package as shown below. Choose the valid option so that the program prints the following output on screen: `MethodA MethodB`

```
public class A {
    public static void methodA(){
        System.out.println("MethodA");
    }
}
public class B {
    public void methodB(){
        System.out.println("MethodB");
    }
}
```

(a) *** `B b = new B(); A.methodA(); b.methodB();`

(b) `A.methodA(); B.methodB();`

(c) `A a = new A(); a.methodA(); B.methodB();`

(d) `A a = new A(); a.methodA(); b.methodB();`

3.- The following classes are defined in different files. What is the output of this program?

```
package p;
public class Parent {
    public int public;
    private int private;
    protected int protected;
```

```
    public Parent(int a, int b, int c){
        public   = a;
        private   = b;
        protected = c;
    }
}
package p;
public class Child extends Parent{
    public Child(int a, int b, int c){
        super(a,b,c);
    }
}
package p;
public class Main {
    public static void main(String[] args) {
        Parent father = new Parent(10,20,30);
        System.out.print (father.public);
        System.out.print (" " + father.protected);
        Child daughter = new Child(1,2,3);
        System.out.print (" " + daughter.public);
        System.out.print (" " + daughter.protected);
    }
}
```

(a) *** 10 30 1 3

(b) 10 0 1 0

(c) 1 3 1 3

(d) 10 30 10 30

4.- Given the following program. What statement is correct?

```
public class Parent {
    public Parent(){}
    public void print(){
        System.out.print("printParent ");
    }
}
public class Son extends Parent{
    public Son(){}
    public void print(){
        System.out.print("printSon ");
    }
}
public class Daughter extends Parent {
    public Daughter(){}
    public void print(){
        System.out.print("printDaughter ");
    }
}
```

```
public class MainFamily {
    public static void main(String[] args) {
        Parent[] family = {new Parent(), new Son(), new Daughter()};
        for (int i=0; i<3; i++){
            family[i].print();
        }
    }
}
```

(a) *** The *for* loop takes advantage of polymorphism. The output of the program is: `printParent printSon printDaughter`.

(b) The output of the program is: `printParent printParent printParent`.

(c) The *for* loop takes advantage of overload. The output of the program is: `printParent printSon printDaughter`.

(d) The *for* loop takes advantage of overload. The output of the program is: `printParent printParent printParent`.

5.- Given the following program where all classes are in the same package but in different files, indicate which statement is correct.

```
public abstract class A {
    private int i;
    // Declare int j
    A(int i){ this.i = i;}
    public abstract void method(int i);
    public String toString(){
        return "i = " + i + " j = " + j;
    }
}
public class B extends A{
    public B(int integer){
        super(integer);
    }
}
public class Main {
    public static void main(String[] args) {
        B b = new B(10);
        System.out.println(b.toString());
    }
}
```

(a) *** The program does not compile because `B` must implement `method(int i)`.

(b) The program prints `i = 10 j = 10`.

(c) The program does not compile, the keyword `abstract` must be removed from `method` signature in class A.

(d) The program prints `i = 10 j = 0`.

6.- The following classes are defined in the same package, but in different files. Which statement is correct?

```
 1 public class Person {
 2     protected String name;
 3     Person(String s){ name = s;}
 4 }
 5 public class Student extends Person {
 6     private String id;
 7     Strudent(String s, String id) {
 8         super(s);
 9         this.id = id;
10     }
11 }
12 public class Main {
13     public static void main(String[] args) {
14         Person p1 = new Person("Lucia");
15         Person p2 = new Student("Carla", "123");
16         Student  a1 = new Student("Antonio", "456");
17         Student  a2 = (Student) new Person("Carolina");
18     }
19 }
```

(a) *** Line 15 has a valid upcasting while line 17 has an invalid downcasting.

(b) Line 15 has a valid upcasting while line 17 has another valid downcasting.

(c) Line 15 has an invalid upcasting while line 17 has a valid downcasting.

(d) Line 15 has an invalid upcasting while line 17 has a invalid downcasting.

7.- The class `Date` has the method `inRange()` as follows. The class `TestInRange` has the following structure. What statement should be placed in line 14?

```
public boolean inRange(int value, int min, int max){
    return (value >= min && value <= max);
}
```

```
 1 import static org.junit.Assert.*;
 2 import org.junit.BeforeClass;
 3 import org.junit.Test;
 4 public class TestInRange  {
 5     public static Date date;
 6     @BeforeClass
 7     public static void initialization(){
 8          date = new Date();
 9     }
10     @Test public void testBefore() {...}
11     @Test public void testLowerLimit() {...}
12     @Test
13     public void testBetweenLimits() {
14         // TO DO
15     }
16     @Test public void testUpperLimit() {...}
17     @Test public void testAfter() {...}
```

```
18 }
```

(a) *** `assertEquals(date.inRange(16, 1, 31), true);`

(b) `assert(inRange(16, 1, 30));`

(c) `assert (date.inRange(16, 1, 30), true);`

(d) `assertEquals(inRange(16, 1, 31), true);`

8.- A method returns `true` if the year received as parameter is a leap year, and `false` otherwise: `public boolean leapYear(int year)`. The following list indicates the set of leap years, and those which are not leap years. Leap years: 1604, 1608, 1612, 1616... 1684, 1688, 1692, 1696, 1704 (1700 is not leap year because it is divisible by 100 and not divisible by 400), 1708, 1712... 1792, 1796, 1804 (1800 is not leap year because it is divisible by 100 and not divisible by 400), 1808, 1812... 1892, 1896, 1904 (1900 is not leap year because it is divisible by 100 and not divisible by 400), 1908, 1912... 1992, 1996, 2000 (it is a leap year because it is divisible by 100 and 400), 2004, 2008, 2012... 2092, 2096, 2104 (2100 is not leap year because it is divisible by 100 and not divisible by 400)... 2196, 2204... 2296, 2304... 2396, 2400 (it is a leap year because it is divisible by 100 and 400), 2404... What set of tests would you do the check the validity of `public boolean leapYear(int year)`?

(a) *** 1604, 1900, 2000, 2001

(b) 1604, 2400

(c) 1604, 1900, 2000, 2400

(d) 1604, 1605, 2000

9.- The following method is intended to calculate the square of a number based on the following formula: for values of n greater than 1, *(n-1)*(n-1) = n*n – 2*n + 1*, although the implementation is not correct. Choose the correct option from the following so that the method correctly calculates the square of numbers greater than 1.

```
1 public int square (int n){
2     if (n <= 1){
3         return 1;
4     } else {
5         return (square (n-1) - 2*(n-1) + 1);
6     }
7 }
```

(a) *** Line 5: replace by `return (square(n-1) + 2*n - 1);`

(b) Line 5: replace by `return (square(n-1) - 2*square(n-2) + 1);`

(c) Line 5: replace by `return ((n-1)*(n-1) - 2*n + 1);`

(d) Line 5: replace by `return (square(n-1) - 2*square(n-1) + 1);`

10.- To write a decimal number `n` in its corresponding binary number `b` the following steps are applied: (1) `n<2` implies that `b=n`; (2) `n>=2` implies that `b=n/2`, followed by `n%2`. What instructions should be placed in texttt`L1` and `L2` for the next method to return a string of characters representing the binary number of `n`?

```
public String d2b(int n){
    String s;
    if (n < 2){
        // L1
    } else{
        int m = n % 2;
        // L2
    }
    return s;
}
```

(a) *** `L1: s = ‘‘’’ + n; y L2: s = d2b (n/2) + m;`

(b) `L1: s = ‘‘’’ + n; y L2: s = d2b (m) + n;`

(c) `L1: s = ‘‘’’ + 0; y L2: s = d2b (m) + n;`

(d) `L1: s = ‘‘’’ + 1; y L2: s = d2b (n/2) + m;`

11.- To concatenate two existing linked lists...

(a) *** we need to traverse one of the lists until its last element and assign as the following element the first of the other list

(b) we need to traverse both lists until their last elements and assign as the following element of one the lists the last node of the other list

(c) we need to create a new list with size the sum of the elements of the two existing lists and copy all the elements of the two existing lists to the newly created list

(d) we need to assign as the first element in one list the first element in the other list

12.- In a binary search tree that represents the destinations an airline flies to, the following information is inserted sequentially, also acting as a key (in alphabetical order): “Washington”, “Toronto”, “Madrid”, “Barcelona”, “Berlin”, “Amsterdam”, “Lisbon”, what is the height of the resulting tree?

(a) *** 5

(b) 3

(c) 4

(d) 7

13.- The following elements are inserted in a heap (*max-heap*) sequentially, 4,6,8,2,1,3 and then the element with key 6 is extracted. What is the post-order traversal of the resulting heap?

(a) *** 2,1,4,3,8

(b) 1,3,2,4,8

(c) 2,3,1,4,8

(d) 1,2,3,4,8

14.- In a doubly linked list that uses dummy nodes (referenced by *top* and *tail*) and that is empty the following condition is met:

(a) *** *tail.getPrev()* is *top*

(b) *top.getNext()* is *tail.getPrev()*

(c) *top.getNext()* is *null*

(d) *tail* and *top* point to the same node

15.- How many swaps are needed to sort the following array 5,3,4,1,2 from lowest to highest with Heap Sort? Note that the node with key 5 will be the root, with nodes with keys 3 and 4 its left and right children, and nodes with keys 1 and 3 the left and right children of 3.

(a) *** 6

(b) 5

(c) 8

(d) 7

16.- In a simple (singly) linked list in which each node only knows its next node, and in which we have a reference to the first node (*top*) and to the last node (*tail*), which of the following operations is more computationally expensive.

(a) *** Extraction at the end

(b) Extraction at the beginning

(c) Insertion at the beginning

(d) Insertion at the end

17.- If we insert one by one (and in the given order) the nodes with keys 3,1,2,5,7,4,6 in a binary search tree. Which node would be the left child of the node whose key is 5 after finishing the insertion process?

(a) *** The node whose key is 4

(b) The node whose key is 2

(c) The node whose key is 6

(d) The node whose key is 5 will not have a left child

18.- Given an empty stack *s* and an empty queue *q*. What would the method *front()* return applied on the queue after executing the following sequence of statements: *s.push(3); s.push(2); s.push(1); q.enqueue(s.pop()); q.enqueue(5); s.top(); q.dequeue()*.

(a) *** 5

(b) 3

(c) 1

(d) 2

19.- Given a list linked with *first* referencing the first node, what does the following method?

```
public void method(E info) {
    Node<E> current = first;
    while (current!=null) {
        current = current.getNext();
    }
    System.out.println(current.getInfo());
}
```

(a) *** Throws a *NullPointerException*.

(b) Prints the information of the last node.

(c) Prints the information of the penultimate (second-to-last) node.

(d) Prints the information of all the nodes in the list.

20.- Given an unsorted array of a thousand elements, which of the following algorithms requires more memory to sort it?

(a) *** Merge Sort

(b) Heap Sort

(c) Quick Sort

(d) Insertion Sort