Systems Programming
(English group)

Leganés, June 27, 2019 — Final exam resit (extraordinary call) (test)
Duration: 40 min — 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.- Given the following recursive algorithm with input values a non-empty array of integers `a`, and an integer value `x` to be searched in the array, select the correct option...

```
public static int binarySearchRecursive(int[] a, int x){
    return binarySearchRecursive(a, 0, a.length-1, x);
}
public static int binarySearchRecursive(int[] a, int first, int last, int x){
    int half;
    if(first <= last){
        half = (first + last) / 2;
        if (a[half] == x){
            return half;
        }
        else if (a[half]<x){
            return binarySearchRecursive(a, half+1, last, x);
        }
        else if (a[half]>x){
            return binarySearchRecursive(a, first, half-1, x);
        }
    }
}
```

(a) *** It is a tail recursion.
(b) It is a non-tail recursion.
(c) The base case cannot be reached.
(d) It is a nested recursion.

2.- Select the correct option.

(a) *** White box tests does not ensure that the program properly works.
(b) JUnit shows the percentage of code coverage for black box tests.
(c) With 100 % code coverage, we can ensure the correct performance of the program.
(d) White box tests are also called input/output tests.

3.- The resolution of the method call in case of overloading depends on...

(a) *** The parameters of the method.
(b) The return type of the method.
(c) The visibility of the method.
(d) Whether the method is static or not.

4.- If you try to compile the following code and the compiler says: *error: ClassB is not abstract and does not override abstract method method1() in ClassA*, what can you say about `ClassA` and `ClassB`?

```
public class Main{
    public static void main(String args[]){
        ClassB b  = new ClassB();
    }
}
```

(a) *** `ClassB` extends `ClassA`. `ClassA` is abstract with `method1()` also abstract.

(b) `ClassB` does not extend `ClassA` but needs to implement `method1()`.

(c) `ClassB` extends `ClassA` and needs to implement interface `Abstractable`.

(d) `ClassB` does not extend `ClassA` and `ClassB` has to be abstract.

5.- Given the following code, is the program correct?

```
public class Main{
    public static void main(String args[]){
        int[] elements = {1,2,3,4,5,6};
        int i = 0;
        int len = elements.length;
        while (i<=len){
            System.out.println(elements[i]);
            i++;
        }
    }
}
```

(a) *** It is not correct; it throws an `ArrayIndexOutOfBoundsException`.

(b) It is correct; it prints on screen the elements of the array and ends correctly.

(c) It is not correct; it does not compile indicating that value resulting from calling `elements.length` cannot be assigned to a variable of type int.

(d) It is not correct; it does not compile indicating that the array has not been initialized properly.

6.- Given the following program, select the correct option.

```
public class Main{
    protected int i;
    public static void main(String args[]){
        Main m = new Main();
        m.i = 0;
        System.out.println(m.i);
    }
}
```

(a) *** The program compiles and runs correctly printing 0 on screen.

(b) The program does not compile because the attribute `i` is protected and cannot be accessed by the object `m`.

(c) The program does not compile because the object `m` cannot be created in the `main` method of class `Main`.

(d) The program compiles and runs correctly printing `null` on screen.

7.- Given the following code, select the correct option.

```
public class ClassA extends ClassB, ClassC implements InterfaceD, InterfaceE{
    public static void main(String args[]){ }
  }
}
```

(a) *** The program does not compile because Java does not support inheritance of more than one class at the same time.

(b) The program does not compile because Java does not support the implementation of more than one interface at the same time.

(c) The program does not compile because we should have explicitly implemented a constructor in `ClassA`.

(d) The program does not compile because in `ClassA` the `main` method cannot be implemented.

8.- Given the following code, select the correct option.

```
class ClassA{
    public ClassA() { }
}
class ClassB extends ClassA {
    private int a;
    public ClassB(int a) {super(); this.a = a;}
}
class ClassC extends ClassB{
    private int b;
    public ClassC(int a, int b) {super(); this.b = b;}
}
public class Main{
    public static void main(String args[]){
        ClassC c = new ClassC(10,10);
    }
}
```

(a) *** The calls to the constructors in the inheritance hierarchy are not correct for the object `c`.

(b) The object `c` is correctly created.

(c) An object cannot be created in a `static` method.

(d) `super()` must be replaced by `this()` in the code of `ClassC`.

9.- Given the following code, select the correct option.

```
public class ClassA{
    public void m(int a) { ; }
    public void m(float b) { ; }
    public void m(int c) { ; }
    public void m(double d) { ; }
}
```

(a) *** The overload for `public void m(int c)` is not correct.

(b) The overload for `public void m(float b)` is not correct.

(c) The overload for `public void m(double d)` is not correct.

(d) Method `m` is correctly overloaded.

10.- Given the following code, select the correct option.

```
interface InterfaceA{
    void m();
}
class ClassA implements InterfaceA{
    private int a = 1;
    public void m() {System.out.print(a);}
}
class ClassB extends ClassA implements InterfaceA{
    private int b = 2;
    private int c = 3;
    public void m() {System.out.print(b);}
    public void c() {System.out.print(c);}
}
public class Main{
    public static void main(String args[]){
        InterfaceA ia = new ClassA();
        ia.m();
        ia = new ClassB();
        ia.c();
    }
}
```

(a) *** The method `c()` cannot be accessed from the reference `ia`.

(b) The program displays on the screen: 1 3.

(c) The program displays on the screen: 1 2 3.

(d) The program displays on the screen: 1 2.

11.- Select the correct option about linear data structures.

(a) *** Stacks and queues can be implemented both with arrays and linked lists.

(b) Arrays can increase their capacity dynamically depending on the number of elements to be stored.

(c) Linked lists require contiguous memory space to store the data.

(d) It is necessary to indicate the size of a linked list before its creation.

12.- What is the following method doing on a double-linked list which contains two dummy nodes `top` and `tail`?

```
public E m(){
    E result = null;
    if(tail.getPrev()!=top){
      result = tail.getPrev().getInfo();
```

```
        tail.setPrev(tail.getPrev().getPrev());
        tail.getPrev().getPrev().setNext(tail);
        size--;
    }
    return result;
}
```

(a) *** It removes an element on the side of `tail` (`removeLast`)

(b) It removes an element on the side of `top` (`removeFirst`)

(c) It inserts an element on the side of `tail` (`insertLast`)

(d) It inserts an element on the side of `top` (`insertFirst`)

13.- Given the following code, select the correct option.

```
public E m() {
    E result = null;
    if(top!=-1){
        result = data[top];
        data[top] = null;
        top = top -1;
    }
    return result;
}
```

(a) *** It is equivalent to the method `pop()` in an `ArrayStack`

(b) It is equivalent to the method `dequeue()` in a `LinkedQueue`

(c) It is equivalent to the method `push()` in an `ArrayStack`

(d) It is equivalent to the method `removeFirst()` in a `LinkedList`

14.- Given the following method, which receives as parameter the array {1, 2, 3}, and considering that class `LStack` implements the expected behavior of a stack, and class `LQueue` implements the expected behavior of a queue, select the correct option.

```
public void m(int array[]){
    LStack s = new LStack();
    LQueue q = new LQueue();
    for(int i=0; i<array.length; i++){
        s.push(array[i]);
    }
    for(int i=0; i<array.length; i++){
        q.enqueue(s.pop());
    }
    for(int i=0; i<array.length; i++){
        System.out.println(q.dequeue());
    }
}
```

(a) *** It prints the numbers 3, 2, 1.

(b) It prints the numbers 1, 2, 3.

(c) It throws an `ArrayIndexOutOfBoundException`.

(d) A stack and a queue cannot coexist in the same method.

15.- Select the correct option.

(a) *** In a *min-heap* the key of the children is always higher than that of the parent.

(b) In a binary tree, and given a node, the keys of the nodes of its left subtree are always higher than those of its right subtree.

(c) A binary tree is always a binary search tree.

(d) A tree cannot be represented by an array because a tree is a hierarchical structure.

16.- What does the following method calculate in a binary tree with `getLeft()` returning the left subtree and `getRight()` returning the right subtree?

```
public int m() {
    int result = 0;
    try {
        result = 1 + getLeft().m() + getRight().m();
    } catch (BTreeException e) {
        result = 0;
    }
    return result;
}
```

(a) *** The size of the tree.

(b) The depth of the tree.

(c) The height of the tree.

(d) It checks if the tree is empty and returns zero if it is empty.

17.- Given the unsorted array {3, 1, 7, 2, 5, 9}, what would be the content of the array after three iterations of the external loop of the Selection Sort method, if we use the implementation that sorts from lowest to highest selecting the minimum value of the unsorted part and swapping it for the first unsorted element?

(a) *** {1, 2, 3, 7, 5, 9}

(b) {1, 3, 2, 5, 7, 9}

(c) {1, 3, 7, 2, 5, 9}

(d) {1, 3, 2, 5, 9, 7}

18.- Given the binary search tree resulting from inserting one by one the following sequence of nodes {3, 1, 5, 2, 7, 9}, what is the result of traversing the tree in post-order after extracting the node whose value is 7?

(a) *** 2, 1, 9, 5, 3

(b) 1, 2, 3, 5, 9

(c) 3, 1, 5, 2, 9

(d) 3, 1, 2, 5, 9

19.- Given the *min-heap* resulting from inserting one by one the sequence of nodes {5, 1, 3, 2, 7}, which of the following *arrays* corresponds to the resulting heap after calling the method `extract()`?

(a) *** {2, 5, 3, 7}
(b) {3, 2, 5, 7}
(c) {2, 3, 5, 7}
(d) {2, 7, 3, 5}

20.- Which sorting algorithms uses recursive partitions, chooses an element as pivot, and swaps elements so that those on the left side are lower than the pivot, and those on the right side are higher than the pivot (assuming that the sorting is done in ascending order)?

(a) *** Quick Sort.
(b) Bubble Sort.
(c) Insertion Sort.
(d) Selection Sort.