# BIRKBECK

(University of London)

MSc EXAMINATION FOR INTERNAL STUDENTS

*MSc Computer Science*

Department of Computer Science and Information Systems

## Programming in Java

BUCI033S7

**DATE OF EXAMINATION:** Tuesday, 9th June 2015
**TIME OF EXAMINATION:** 10:00–13:00
**DURATION OF PAPER:** Three hours

RUBRIC:

1. Candidates should attempt ALL 12 questions on this paper.
2. You are advised to look through the entire examination paper before getting started, in order to plan your strategy.
3. Simplicity and clarity of expression in your answers is important.
4. All programming questions should be answered using the Java programming language unless stated to the contrary.
5. Electronic calculators are NOT allowed.
6. Start each question on a new page.

| Question: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Marks: | 4 | 4 | 3 | 6 | 3 | 8 | 5 | 8 | 9 | 22 | 18 | 10 | 100 |

 

Question 1 ..... Total: 4 marks

What does the following Java program print if it is executed with `63` as its argument? What is the function that it computes? Explain your answer.

```
1  public class Thing {
2      public static void main(String[] args) {
3          int n = Integer.parseInt(args[0]);
4          unknown(2, n);
5          System.out.println();
6      }
7
8      public static void unknown(int a, int b) {
9          if (a <= b) {
10             if (b % a == 0) {
11                 System.out.print(" " + a);
12                 unknown(a, b / a);
13             } else {
14                 a = a + 1;
15                 unknown(a, b);
16             }
17         }
18     }
19 }
```

Question 2 ..... Total: 4 marks

List the access modifiers and explain what each means for the instance variable or method that it modifies.

Question 3 ..... Total: 3 marks

Consider the following scenario.

Yasmine and Poppy are working a software system and are using git, a distributed version control system to manage their code. Poppy pulls from the master repository and updates her working copy. She realises there is a bug in the code and works to fix it. When she is finished, she commits her code. The next day, Yasmine begins working on the code by pulling from the master repository and updating her working copy. She too realises the code has a bug (the same one Poppy saw) and spends the whole day fixing it.

What went wrong? Why didnt Yasmine benefit from Poppys work?

Question 4 ..... Total: 6 marks

Briefly explain each of the following terms:

- dynamic binding,
- cast expression or casting,
- overriding,
- `this`.

You should provide appropriate examples to illustrate your answer.

Question 5 ..... Total: 3 marks

In Java, `Integer` is a subtype of `Number` but `List<Integer>` is not a subtype of `List<Number>`. Why? (Dont give just a Java-specific answer; explain why Java shouldnt allow `List<Integer>` to be a subtype of `List<Number>`.)

Question 6 ..... Total: 8 marks

What does the following Java program print if it is executed with `63` as its argument? Explain your answer.

```
1  public class CharList {
2      public static Node myList = new Node((char) 0, null);
3
4      public static void main(String[] args) {
5          for (String s : args) {
6              for (int i = 0; i < s.length(); i++) {
7                  int c = Integer.parseInt(s.substring(i, i + 1));
8                  for (int n = c; n > 0; n--) {
9                      myList.add(s.charAt(i));
10                 }
11             }
12         }
13         for (Node nd = myList.getNext(); nd != null; nd = nd.getNext()) {
14             System.out.print(nd.getContent() + " ");
15         }
16         System.out.println();
17     }
18 }
19
20 class Node {
21     private char content;
22     private Node next;
23
24     public Node(char c, Node tail) {
25         content = c;
26         next = tail;
27     }
28
29     public void add(char c) {
30         next = new Node(c, getNext());
31     }
32
33     char getContent() {
34         return content;
35     }
36
37     Node getNext() {
38         return next;
39     }
40 }
```

Question 7 ..... Total: 5 marks

Suppose that you are modifying a program that contains a `LinkedList` class implementing:

- a constructor (creates an empty list),
- `add(T value)` (adds a node), and
- `contains(T value)` (returns true or false if the list contains or is lacking the value).

where `T` is the parameterised type of the element held in the list.

You may asssume that the list contains a head node that has a null value and is used just so that even an empty list has at least one node in it. (Thus, if `headNode` is a list, and the list is not empty, the first node containing a value will be `headNode.next()`.

Write a method

```
public LinkedList<T> deDup()
```

(called as `x = y.deDup()`) which, when given a list `y` that might contain duplicated nodes (with the same values), returns a new list of nodes in the same order but with duplicates removed.

You should not assume that list `y` is sorted.

Question 8 ..... Total: 8 marks

Write a method to implement simple letter substitution codes:

```
public static String encode(String source, String from, String to)
```

that works as follows.

- The first argument is a message to encode.
- The second and third arguments are both strings of $k$ characters:
- the second being a list of characters to encode and,
- the third, their coded substitute.

If a character isn't in the `from` list, it should be copied unchanged. So, for our example, if `source = "Amy loved her dog"`, `from = "dl"`, and `to = "gd"`, then the input string becomes

```
Amy doveg her gog
```

Question 9 ..... Total: 9 marks

What would be the result of executing the following code?

You should show your working.

```
1  public class MC {
2      public static void main(String args[]) {
3          try {
4              try {
5                  C c = new C();
6                  System.out.println(c.max(11, 21));
7              } catch (ArithmeticException | IllegalArgumentException rte) {
8                  System.out.println(rte);
9              } finally {
10                 System.out.println("In finally of main " + MC.class.getSimpleName());
11                 throw new MyException("interesting");
12             }
13         } catch (Exception ex) {
14             System.out.println(ex);
15         } finally {
16             throw new RuntimeException("again");
17         }
18     }
19 }
20
21 class A {
22     int max(int x, int y) {
23         try {
24             if (x > y)
25                 x++;
26             else
27                 throw new Exception("Oh Dear!");
28             System.out.println("A::max value of x is " + x);
29         } catch (Exception ex) {
30             System.out.println("In exception " + ex.getMessage());
31             System.out.println("x = " + x + " y = " + y);
32             return y;
33         } finally {
34             System.out.println("A::max finally block");
35             throw new IllegalArgumentException("Finally of max in "
36                     + this.getClass().getSimpleName() + " with x = " + x);
37         }
38     }
39 }
40
41 class C extends A {
42     public int max(int x, int y) {
43         return super.max(x - 3, y + 10);
44     }
45 }
46
47
48 class MyException extends Exception {
49     MyException(String s) {
50         super(s);
51     }
52 }
```

Question 10 ..... Total: 22 marks

(a) Write a generic method to count the number of elements in a collection that have a specific property (for example, odd integers, prime numbers, palindromes). 5 marks

For example, the following program counts the number of odd integers in an integer list:

```
package generics;

import java.util.Arrays;
import java.util.Collection;

public class TestCount {
    public static void main(String[] args) {
        Collection<Integer> ci = Arrays.asList(1, 2, 3, 4);
        int count = Algorithm.countIf(ci, (x) -> x % 2 != 0);
        System.out.println("Number of odd integers = " + count);
    }
}
```

(b) Write a generic method to exchange the positions of two different elements in an array. 5 marks

(c) What is the following class converted to after type erasure? 4 marks

```
package generics;

public class Pair<K, V> {
    private K key;
    private V value;

    public Pair(K key, V value) {
        this.key = key;
        this.value = value;
    }

    public K getKey() {
        return key;
    }

    public V getValue() {
        return value;
    }

    public void setKey(K key) {
        this.key = key;
    }

    public void setValue(V value) {
        this.value = value;
    }
}
```

(d) Write a generic method to find the maximal element in the interval `[begin, end)` represented as a list. 8 marks

Question 11 ..... Total: 18 marks

In the following code, a developer was trying to understand how concurrency works, but got confused. His basic idea is to run two computations in parallel, but as part of his debugging tests he added the `count` variable seen below.

```
import static java.lang.Thread.sleep;

public class Thready {
    private int count = 0;

    public class Worker1 implements Runnable {
        public void run() {
            for (int i = 0; i < 10000; i++) {
                computeSomething(i);
                count = count + 1;
            }
        }
    }

    public class Worker2 implements Runnable {
        public void run() {
            for (int i = 0; i < 10000; i++) {
                computeSomethingElse(i);
                count = count - 1;
            }
        }
    }

    public void runThreadExperiment() throws InterruptedException {
        for (int k = 0; k < 10; k++) {
            Thread t1 = new Thread(new Worker1());
            Thread t2 = new Thread(new Worker2());
            t1.start();
            t2.start();

            t1.join(); // This waits for thread t1 to finish
            t2.join(); // And this waits for thread t2 to finish
            System.out.println(count);
            count = 0;
            sleep(100);
        }
    }

    public void computeSomething(int i){
        // dummy operation
    }

    public void computeSomethingElse(int i){
        // dummy operation
    }
}
```

where the member method

```
public final void join() throws InterruptedException
```

waits for the thread to die.

(a) Suppose that from the `main` method the developer executes the following code: 4 marks

```
Thready mythready = new Thready();
  mythready.runThreadExperiment();
```

The developer was hoping this program would print 0 each time through the main loop. What might it actually print? (If the value of `count` is undefined when the `println` is executed, say so and explain why).

(b) Explain why the code, as shown above, is incorrect. What basic Java rule is being violated here? Could the error have any effect on the value of `count` printed as the program loops? If you answer "yes", could this error change the number of times `computeSomething` or `computeSomethingElse` are called, or will each be called 10,000 times per execution of the main loop, as seems to be intended? 6 marks

(c) How can this code be modified to make sure that the behaviour described in (a) is what really occurs? Explain the effect of your change. If your modified version of the program is run on a two-core machine, would it really obtain any speedup? 8 marks

Hint: Assume that if two threads "contend" for a lock on some object, each thread loses about 1/10000 of a second in locking overhead, plus of course any delay until the lock is released, if the object was locked.

Question 12 ..... Total: 10 marks

Imagine that you have a job at Bobble Maps and are implementing a Java class to remember the distances between cities, so that they dont need to be recomputed every time the same question occurs. As distances are symmetric, if the distance is known from city1 to city2, the same result should be returned for queries that start at city2 and end at city1. Distances are represented as floating point numbers. Your class should have two methods:

```
public class Memory {
  // Remember the distance between two cities
  public void saveDist(String city1, String city2, float d) {... }

  // Return the saved distance, if known, else an empty optional
  public Optional<Float> lookupDist(String city1, String city2) { ... }
}
```

Write code for the two methods. You can add fields to the `Memory` class if you wish.