Cognome e nome:_____ Matricola: _____ Posto: _____

Università degli Studi di Padova – Dipartimento di Matematica.
Corso di Laurea in Informatica

Regole dell'esame

Il presente esame deve essere svolto in forma **individuale** in un tempo massimo di **60 minuti** dalla sua presentazione. **Non** è consentita la consultazione di libri o appunti in forma cartacea o elettronica, **né** l'uso di telefoni o altri device. Per la convalida e registrazione del voto finale il docente si riserva di proporre al candidato una **prova orale**.

Q1. Considerando la realtà sotto modellata

```
class Student {
      private int age;
      private boolean female;

      /** get/set methods **/
      public boolean isFemale() {return female;}
      public int getAge() {return age;}
}

class Course {
      private String className;
      private List<Student> students;

      /** get/set methods **/
      public String getClassName() {return className;}
      public List<Student> getStudents() {return students;}
}

public class CorsoStudiStats {

      public static ... entryPoint(String args[]) {

            List<Course> studyPath = new ArrayList<Course>();
            ...
      }
}
```

calcolare con l'ausilio delle lambda e Stream API (approccio funzionale):

1. il numero totale delle studentesse (F) iscritte nel percorso di laurea
2. il numero totale dei studenti (M e F) iscritti al corso con *className* PCD1920
3. l'eta dello studente (M o F) più anziano iscritto al corso con *className* PCD1920.
4. (**bonus**) una partizione (mappa) delle studentesse ("F") e studenti ("M") scritti al corso PCD1920.

Q2. La capacita di adattamento a un carico di lavoro sempre in crescita di un sistema e' conosciuta come?

A. scalabilità
B. tolleranza ad errori
C. capacita di risposta alle richieste in tempi rapidi
D. nessuna delle alternative sopra elencate

Cognome e nome:_____ Matricola: _____ Posto: _____

Q3. Quale affermazione è vera per il seguente programma:

```
public class Exam {
      public . . . void entryPoint(String[] args) {
            String name = "PCD1920";
            Runnable r = () -> System.out.println(name);
            name = name.toLowerCase();
            r.run();
            System.out.println(“3 appello”);
      }
}
```

A. Il programma non compila [*motivare brevemente la risposta]
B. Il programma stampa pcd1920 3 appello
C. Il programma stampa 3 appello pcd1920
D. Il programma stampa PCD1920 3 appello

Q4. Quale affermazione è vera per il seguente programma:

```
public class Exam {
    public static ... entryPoint(String [] args) {
        System.out.print("1 ");

        synchronized(args) {
            System.out.print("2 ");
            try {
                    args.wait();
            }
            catch(InterruptedException e){ }
        }
        System.out.print("3 ");
    }
}
```

A. Il programma non compila [motiva la risposta brevemente*]
B. Stampa 1 2 3
C. Stampa 1 2
D. Stampa 1 3
E. Nessuna delle alternative elencate e’ valida

Q5. Quale affermazione è vera per il seguente programma

```
@FunctionalInterface
interface A {public default int m() {return 1;}}

class B {public int m() {return 2;}}

public class Exam extends B implements A {

      public static ... entryPoint(String[] args) {
            System.out.println(new Exam().hashCode());
      }
}
```

A. Il programma non compila [motivare la risposta]
B. Il programma compila e stampa 1
C. Il programma non produce alcun input
D. Il programma compila e stampa 2

Cognome e nome:_____ Matricola: _____ Posto: _____

Q6. Quale delle affermazioni è vera per il seguente programma (più di una):

```
class Exam extends Thread
{
    final static Object obj1 = new Object();
    final static Object obj2 = new Object();

    public static ... entryPoint(String args[]) {
        final Exam e = new Exam();

        new Thread() {
            public void run() {
                synchronized(obj1)
                {
                    System.out.println(" 1 ");
                    synchronized(obj2)
                    {
                        System.out.println(" 2 ");
                    }
                }
            }
        }.start();

        new Thread() {
            public void run() {
                synchronized(obj2)
                {
                    System.out.println(" 3 ");
                    synchronized(obj1)
                    {
                        System.out.println(" 4 ");
                    }
                }
            }
        }.start();
    }
}
```

A. Il programma può andare in deadlock
B. Il programma può stampare 1 2 3 4
B. Il programma può stampare 3 4 1 2
D. Il programma non compila.
E. Il programma può stampare 1 3 2 4

Q7. Selezionare <u>solo</u> i quesiti falsi [nessuna spiegazione e' necessaria]:

A. Un thread non può accedere ad oggetti creati da altri thread distinti.
B. Per garantire la thread-safety dobbiamo occuparci di sincronizzare l'accesso alle variabili locali dei metodi.
C. Un programma Java termina quando il thread che esegue il metodo main() termina.
D. Se eseguiamo un programma Java in un computer con 2 processori/core, possiamo creare al più due thread.
E. Un thread Java e' mandato in esecuzione chiamando il metodo proprio run().

Cognome e nome:_____ Matricola: _____ Posto: _____

Q8. Quale affermazione è vera per il seguente programma:

```
abstract class I {
	public static void write() {
		System.out.print(" Interface...");
	}
}
class A extends I {
	A() {super.write();}
	public static void write() {
		System.out.print(" A...");
	}
}
public class Exam extends A {
	public static void write() {
		System.out.print(" Exam...");
	}
	public static ... entryPoint(String[] args) {
		A a = new Exam();
		a.write();
	}
}
```

A. Il programma non compila
B. Il programma stampa Interface... A...
C. Il programma stampa Exam... A...
D. Il programma stampa A... Interface…

Q9. Quale affermazione è vera per il seguente programma:

```
class Bird {
	static { System.out.print("static1 "); }
	{ System.out.print("init1 "); }

	public Bird() { System.out.print("init2 "); }
	static { System.out.print("static2 "); }
}

public class Exam extends Bird {
	Exam(){
		System.out.print("PCD1920 ");
	}
	public static ... entryPoint(String[] args) {
		new Exam();
	}
}
```

A. Stampa PCD1920 static1 init1 init2 static2
B. Stampa static1 init1 static2 PCD1920
C. Stampa PCD1920 init2 init1 static1 static2
D. Stampa static1 static2 init1 init2 PCD1920

Q10. A cosa si riferisce il fenomeno di hash-*collision* nel contesto delle strutture a indice?

A. Quando due chiavi identiche producono un valore hash diverso.
B. Quando due chiavi diverse producono lo stesso valore hash.
C. Quando due chiavi identiche producono un valore hash uguale.
D. Nessuna delle alternative sopra elencate.

Cognome e nome:_____ Matricola: _____ Posto: _____

Q11. Quale affermazione è vera per il seguente programma:

```
public class StringWrapper {
      private String s;
      public StringWrapper(String s) {this.s = s;}
      public static ... entryPoint(String args[]) {
            HashSet<Object> sw = new HashSet<>();
            StringWrapper sw1 = new StringWrapper("ab");
            StringWrapper sw2 = new StringWrapper("cd");
            String s1 = new String("ab");
            String s2 = new String("cd");
            sw.add(sw1); sw.add(sw2);
            sw.add(s1); sw.add(s2);

            System.out.println(sw.size());
      }
}
```

A. Stampa 1
B. Stampa 2
C. Stampa 4
D. Non compila [motivare brevemente la risposta]

Q12. Perché è importante mantenere molto breve il "ciclo di servizio" di socket e datagrams (ovvero il tempo che intercorre fra due chiamate successive di ascolto di nuovi dati in arrivo):

A. per evitare che il client remoto vada in timeout
B. per evitare che i buffer del socket si riempiano perdendo dati
C. per mantenere elevato il throughput dell'interfaccia di rete
D. per mantenere alto il blocking factor

Q13. Considerando il frammento di codice:

```
try (
      ServerSocket serverSocket = new ServerSocket(8081);
      Socket socket = serverSocket.accept();
      PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
      BufferedReader in = new BufferedReader(new
                  InputStreamReader(socket.getInputStream()))) {
            String inputLine;
            while ((inputLine = in.readLine()) != null) {
                  System.out.println("Received: " + inputLine);
                  out.println("Hello " + inputLine);
            }
      }
```

Quali di queste responsabilità sono prese in carico della classe BufferedReader?

A. stabilire un protocollo orientato a righe di testo
B. prevenire il riempimento del buffer del socket in caso di traffico elevato
C. aumentare l'affidabilità del socket in caso di interruzione di rete
D. trasformare i byte ricevuti in caratteri all'interno di una String

Cognome e nome:_____ Matricola: _____ Posto: _____

Q14. Quale affermazione e' vera per il seguente programma:

```
public class Test {
      public static ... entryPoint(String args[]){
            MyGenerics<Integer> m = new MyGenerics<Integer>();
            m.set("PCD1920");
            System.out.println(m.get());
      }
}
class MyGenerics<T> {
      T var;
      void set(T var) {
            this.var = var;
      }
      T get() {
            return var;
      }
}
```

A. Stampa PCD1920
B. Il programma non compila
C. Il programma lancia un eccezione a tempo d'esecuzione
D. Nessuna delle alternative sopra elencate