Practice Final

```
1  class Race{
2          public static final int MAX = 5;
3          public static final int MIN = 1;
4
5          private static int data=Race.MIN;
6          public static int getData(){
7                  return data;
8          }
9          public static void incData() {
10                 if (data < Race.MAX)
11                         data++;
12         }
13         public static void foo() throws Exception{
14                 while (data < Race.MAX)
15                         incData();
16         }
17 }
18
19 public class Car implements Runnable {
20         public void run(){
21                 Race.foo();
22         }
23         public static void main(String[] args) throws Exception{
24                 Thread t1 = new Thread(new Car());
25                 Thread t2 = new Thread(new Car());
26                 t1.start();
27                 t2.start();
28                 t1.join();
29                 t2.join();
30                 System.out.println(Race.getData());
31         }
32 }
```

1. In the above code, where might be race condition(s) occur? Specifically which line(s) of the code ?
   The race condition is on line 10 because multiple threads try to access to data and Race.MAX at the same time and try to increment it at the same time.
   Example : if data = 4, two threads may add one to data at the sale time then data = 6 but shouldn't be greater than 5.

2. Where and how should the race condition(s) be prevented?
   We can block the method by synchronized them.

3. What is a deadlock? Can one occur in the above code ever? Why or why not?
   A deadlock is a situation where a thread threadA wait a other thread threadB to give a signal, but threadB wait for threadA to give a signal (endless loop).

```
1  class Examine{
2          static int[] data = new int[10];
3          public static void dance(int value){
4                  int i=0;
5                  while(i<data.length){
6                          for(int j=0;j<100;j++);
7                                  data[i] = value;
8                  }
9          }
10         static Runnable launch(int id){
11                 return new Runnable(){
12                         public void run(){
13                                 dance(id);
14                         }
15                 };
16         }
17         public static void main(String[] args){
18                 Thread t1 = new Thread(launch(1));
19                 Thread t2 = new Thread(launch(2));
20                 t1.start();
21                 t2.start();
22                 t1.join();
23                 t2.join();
24         }
25 }
```

4. In the above code is there a race condition? If yes, what is the race condition and where exactly is it (demonstrate the race via code walk thru)?
   Yes, there is a race condition. A race condition is a situation where multiple threads try to access and modify the same shared variable at the same time. The race condition is on line 7 because t1 and t2 try to access to and modify data[i] at the same time but t1 give the value one whereas t2 give the value 2.

5. Would the code when run cause an Exception?
   Oui mais la réponse que je vais donner est pas la correction du prof mais celle de javac : il faut des blocs try/catch pour utiliser les .join() pour que les erreurs soient traitées s'il y en a

```
class Grain {
        public String toString() { return "Grain"; }
}

class Wheat extends Grain {
        public String toString() { return "Wheat"; }
}
class Mill {
        Grain process() { return new Grain(); }
}
class WheatMill extends Mill {
        Wheat process() { return new Wheat(); }
}
public class CovariantReturn {
        public static void main(String[] args) {
                Mill m = new Mill();
                Grain g = m.process();
                System.out.println(g);
                m = new WheatMill();
                g = m.process();
                System.out.println(g);
        }
}
```

6. Is the method "process()" in WheatMill in error?
   No error with recent versions of Java. Wheat is a child class of Grain so Wheat is a Grain.

```
class Egg2 {
        protected class Yolk {
                public Yolk() { System.out.println("Egg2.Yolk()"); }
                public void f() { System.out.println("Egg2.Yolk.f()");}
        }
        private Yolk y = new Yolk();
        public Egg2() { System.out.println("New Egg2()"); }
        public void insertYolk(Yolk yy) { y = yy; }
        public void g() { y.f(); }
}

public class BigEgg2 extends Egg2 {
        public class Yolk extends Egg2.Yolk {
                public Yolk() { System.out.println("BigEgg2.Yolk()"); }
                public void f() { System.out.println("BigEgg2.Yolk.f()"); }
        }
        public BigEgg2() { insertYolk(new Yolk()); }
        public static void main(String[] args) {
                Egg2 e2 = new BigEgg2();
                e2.g();
        }
}
```

7. Give the output for the above:

```
// Appel au constructeur de BigEgg2 qui appelle la classe Egg2
// defini la variables y qui appelle le constructeur de Egg2.Yolk
$ Egg2.Yolk()
// Appelle le constructeur de Egg2
$ New Egg2()
// Revient dans le constructeur de BigEgg2 et cree un nouveau Yolk
// remonte dans la classe parente de BigEgg2.Yolk
$ Egg2.Yolk()
// revient dans le constructeur de BigEgg2.Yolk
$ BigEgg2.Yolk()
// Appelle Egg2.g() qui appelle Egg2.Yolk.f()
$ BigEgg2.Yolk.f()
```

```
class Parcel4 {
        private class PContents implements Contents {
                private int i = 11;
                public int value() { return i; }
        }
        protected class PDestination implements Destination {
                private String label;
                private PDestination(String whereTo) {
                        label = "whereTo";
                }
                public String readLabel() { return label; }
        }
        public Destination destination(String s) {
                return new PDestination(s);
        }
        public Contents contents() {
                return new PContents();
        }
}
public class TestParcel {
        public static void main(String[] args) {
                Parcel4 p = new Parcel4();
                Contents c = p.contents();
                Destination d = p.destination("Tasmania");
                Parcel4.PContents pc = p.new PContents();//ERROR
        }
}
```

8. The above class has an error on the line indicated. Explain why there is an error.
   We try to access to a constructor of an object from a private classe without using an object from this private class.

```
public class Wrapping {
        private int i;
        public Wrapping(int x) { i = x; }
        public int value() { return i; }
}

public class Parcel {
        public Wrapping wrapping(int x) {
        //return an anonymous inner class object of Wrapping type
        //with overloaded method "public int value (){ return 47*i;}"
        }
        public static void main(String[] args) {
                Parcel p = new Parcel();
                Wrapping w = p.wrapping(10);
        }
}
```

9. In the above classes provide the missing code.

```
// A verifier (pas tres bien compris ce que le prof a dit)
public Wrapping wrapping(int x){
        return new Wrapping(x){
                public int value(x){
                        return 47*x;
                }
        };
}
```

10. Given an ArryList is-a List and a List is-a Collection is the following true? ArrayList<String> is-a Collection<String>

    Yes, because ArrayList<String> is a List<String> is a Collection<String>.

11. Given interface FooBar<X,Y> extends Silly<X> which of the following ARE subtypes of Silly<String>?

    (a) FooBar<String, String>
    (b) FooBar<String, Integer>
    (c) FooBar<Integer, String>
    (d) FooBar<String, Exception>
    (e) FooBar<Integer, Integer>
    (f) FooBar<Exception, Integer>

12. Given

    static <T>T pick(T a, T b){return b}

    Is the following an error or not? If an error, explain why, if not give the return type.

    Collection c = pick(new Set<String>(), new Stack<String>());

    No because they're all collections.

13. If the code below gives an error explain why, if not explain why.

```
public static void addNumbers(List<? super Integer> list){
        for (int i=1; i<10; i++){
                list.add(i);
        }
}
// in other code:
addNumbers(new ArrayList<Number>());
```

There is no error because, due to ? super Integer, the list accepts Integer, Number and Object. So we can add an int element to a list of Integer/Number/Object.

14. If the code below gives an error explain why, if not explain why.

```
void swapFirst(List<? extends Number>listA, List<? extends Number> listB){
        Number temp = listA.get(0);
        listA.set(0,listB.get(0));
        listB.set(0,temp);
}
```

There is an error because the two list may contains different types (ex : List<Integer>listA et List<Float>listB)

```
class Example{
        public void open() throws FileNotFoundException{
                System.out.println("attempting to open file");
                throw new FileNotFoundException();
        }
        public void close() throws CloseException {
                System.out.println("attempting to close file");
                throw new CloseException();
        }
        public static void main(String[] args) throws Exception{
                Example e = new Example();
                try{
                        e.open();
                        System.out.println("after opening file");
                }finally{
                        System.out.println("finally");
                        e.close();
                        System.out.println("after closing file");
                }
                System.out.println("end of program");
        }
}
```

15. Give the output. State any exception(s) that are displayed on exit.

```
$ attempting to open file
$ finally
$ attempting to close file
$ Error CloseException()
```

```
class LanguageException extends Exception{}
class JavaException extends LanguageException{}

public class Test {
        public void a() throws LanguageException{
                throw new LanguageException();
        }
        public void b() throws JavaException{
                throw new JavaException();
        }
        public static void main(String[] args){
                Test t = new Test();
                try{
                        t.a();
                        t.b();
                }
                catch(LanguageException l){}
                catch(JavaException j){}
                System.out.println("finished main");
        }
}
```

16. Give the output. State any exception(s) that are displayed on exit.
    There is a compile time error because JavaException is LanguageException which is already caught the line just before, so catch JavaException never occurs.
    But if we switch the catch lines, the output will be "finished main".

```
public class Out {
        int x;
        static class In {
                public void setX(int value){
                        x = value;
                }
        }
        public static void main(String[] args){
                //your code goes here
        }
}
```

17. What is the error in the above class? Why?
    Yes, there is an error because In.setX() try to access to the x Out attribute but In is a static class so it can directly access to x. We can create a In object without creating a Out object, and in this case, x is not define nor declared yet.

18. Give the code to create an "In" object in main()

```
Out.In inObject = new Out.In();
```

```
class Cat {
        Kitten k = new Kitten();
        public Cat(){
                System.out.println("cat");
        }
        class Kitten{
                public Kitten(){
                        System.out.println("kitten");
                }
        }
}
public class Lion extends Cat {
        public Lion(){
                System.out.println("Lion");
        }
        class Kitten {
                public Kitten(){
                        System.out.println("young Lion");
                }
        }
        public static void main(String[] args){
                new Lion();
        }
}
```

19. Give the output

```
$ kitten
$ cat
$ Lion
```

```
class Cat {
        Kitten k;
        public Cat(){
                System.out.println("cat");
        }
        class Kitten{
                public Kitten(){
                        System.out.println("kitten");
                }
        }
        public void produce(Kitten kk){
                k = kk;
        }
}
public class Lion extends Cat {
        public Lion(){
                System.out.println("Lion");
                produce(new Kitten());
        }
        class Kitten {
                public Kitten(){
                        System.out.println("young Lion");
                }
        }
        public static void main(String[] args){
                new Lion();
        }
```

```
}
```

20. Give the output or if there is an error, fix it and give the output.
(On a pas corrigé celle là mais j'ai essayé de la faire sur mon ordi) There is an error because we use a Lion.Kitten in Lion.Kitten.Kitten() in produce() method but produce method accepts Cat.Kitten only. To fix it, we can put Cat.Kitten in static and do produce(new Cat.Kitten()) in Lion constructor. So the output will be

```
$ cat
$ Lion
$ kitten
```

```
class A {
        private int x;
        public void setZ(int zz){
                z = zz;
        }
        class B{
                private int y;
                class C{
                        private int z;
                        public void setX(int xx){
                                x = xx;
                        }
                }
        }
}
```

21. Examine the code above. Is there an error? If so what is the error and why? If not, explain.
(On a pas corrigé celle là non plus mais je pense que c'est ça)
There is an error because setZ() try to modify z value which does not exist yet.

```
class A{
        class B{
                class C{}
        }
        public static void main(String[] args){
                //code
        }
}
```

22. Give the code necessary to create a C object in main().

```
public static void main(String[] args){
        A a = new A();
        B b = a.new B();
        B.C c = b.new C();
}
```

```
class X {
        int z = 5;
        static class Y{
                public int getZ(){return z;}
        }
        public static void main(String[] args){
                //code
        }
}
```

23. What is the an error in the code above?
    Same answer as the question 20.

24. If the error was removed in the above code, give the code to create a Y object.

```
public static void main(String[] args){
        Y y = new Y();
}
```

```
class X{
        int y;
        public void foo(){
                for(int i=0; i<10; i++){
                        if (y < 5)
                                y++;
                }
        }
}
```

25. For the above code give the places where a race condition exists.
    The race condition exist on the lines with if(y<5) et y++ because if multiple threads access to y==4 at the same time and each one increment y by one, y will be 6 but y can't be more than 5.

```
class A implements Runnable {
        ReentrantLock lock = new ReentrantLock();
        List s ;
        public A(List store){
                s = store;
        }
        public void run(){
                for(int i=0;i<10; i++){
                        if(lock.trylock())
                                s.add(i);
                }
        }
}

class B implements Runnable{
        ReentrantLock lock = new ReentrantLock();
        List s;
        public B(List store){
                s = store;
        }
        public void run(){
                for(int x = 20; x<30; x++){
                        if(lock.tryLock())
                                s.add(x);
                }
        }
}
public class Test {
        public static void main(String[] args){
                List store = new LinkedList();
                A a = new A(store);
                B b = new B(store);
                new Thread(a).start();
                new Thread(b).start();
                Thread.sleep(2000);
        }
}
```

26. Does the above code protect the shared List? Why or why not?
    (Celui là on l'a pas fait et j'ai la flemme de le voir ce soir)

```
public class LTest {
        public static void main(String[] args){
                new Thread(new Runnable(){
                        public void run(){
                                System.out.println("hello world!");
                        }
                }).start();

        }
}
```

27. Write the above using a Lambda expression.

```
new Thread( () -> System.out.println("Hello world!"));
```

```
public class Table{
        enum TYPE {MULT, ADD}
        public void display(int start, int end, TYPE t){
                for (int i = start; i<end; i++){
                        for (int j = start; j <end; j++){
                                if (t == TYPE.MULT)
                                        System.out.printf("%3d ",(i*j));
                                else
                                        System.out.printf("%3d ",(i+j));
                        }
                        System.out.println("");
                }
        }
        public static void main(String[] args){
                Table t = new Table();
                t.display(1,10,TYPE.MULT);
        }
}
```

28. Modify to use a Lambda expression and give the expression to get the same output as the code above with your new display() method.

```
interface BiFunction<X,Y,Z>{
        Z apply(X x, Y y);
}


public void display(int start, int end, Function f){
        for (int i = start; i<end; i++){
                for (int j = start; j <end; j++){
                        System.out.println(f.apply(i*j));
                }
        }
}

public static void main(String[] args){
        Table t = new Table();
        t.display(1,10,(a,b) -> a*b);
 }
```