(5).

Dați un exemplu de situație în Java când folosiți synchronized, și altul când folosiți wait/notify. Justificați necesitatea folosirii unei construcții în detrimentul celeilalte.

```
class MyThread implements Runnable {
    @Override
    public void run() {
        synchronized (Main.lock) {
            Main.a = Main.a + 3;
        }
    }
}
class Main {
    public Object lock = new Object();
    public int a = 5;

    public static void main() {
        Thread t1 = new Thread(new MyThread());
        Thread t2 = new Thread(new MyThread());
        t1.start();
        t2.start();
    }
}
```

Grigore Lucian Liviu.

```
class MyThread implements Runnable {
    @Override
    public void run() {
        synchronized(Main.lock) {
            this.wait();
            int step = 0;
            while (step < 1000) {
                // intense computation
                // verifying SpaceTravel theorem
                // print("Smart stuff");
                ++step;
            }
            this.notify();
        }
    }
}
class Main {
    /* same as before; */
}
```

Cele două variante menționate nu se exclud una pe cealaltă. Folosirea doar a synchronized va asigura că un singur thread intră în acea bucată de cod (thread safe). Additional, wait() poate fi folosit (doar) după o instrucțiune synchronized) pentru a opri thread-ul

Grigore Lucian Liviu.

curent până când alt thread va apela notify. Dacă thread-urile pot aștepta mult timp înainte a obține cum acest, adăugarea wait()/notify() poate fi utilă.

Grigore Lucian