https://folk.ntnu.no/michaeng/tdt4186_22/
michael.engel@ntnu.no

Theoretical exercises
Spring 2022

Theoretical Exercises 2
Synchronization

Please submit solutions on Blackboard by Friday, 11.2.2022 12:00h

2.1 Race conditions

Consider the two parallel threads $t1$ and $t2$ that share their data (variables). Initially, the values of y and $z = 0$.

```
int t1() {
  int x;
  // initialization code
  x = y + z;
  // other code
}
```

```
int t2() {
  // initialization code
  y = 1;
  z = 2;
  // other code
}
```

a. Give all possible final values for x and the corresponding order of execution of instructions in $t1$ and $t2$ (indicate task switches).

b. Is it possible to use semaphores so that the final value of x is 2? If so, give a solution using semaphores and wait/signal operations. If not, explain why now.

2.2 Semaphores

Consider the two parallel threads $t1$ and $t2$.

```
int t1() {
  printf("w");
  printf("d");
}
```

```
int t2() {
  printf("o");
  printf("r");
  printf("l");
  printf("e");
}
```

a. Use semaphores and insert wait/signal calls into the two threads so that only “wordle” is printed.

b. Give the required initial values for the semaphores.

2.3 Even more semaphores

Consider the parallel threads $t1$, $t2$ and $t3$ using the following common semaphores:

```
semaphore s_a = 0, s_b = 0, s_c = 0;
```

```
int t1() {
  while(1) {
    printf("A");
    s_c.signal();
    s_a.wait();
  }
}
```

```
int t2() {
  while(1) {
    printf("B");
    s_c.signal();
    s_b.wait();
  }
}
```

```
int t3() {
  while(1) {
    s_c.wait();
    s_c.wait();
    printf("C");
    s_a.signal();
    s_b.signal();
  }
}
```

Which strings can be output when running the three threads in parallel?

2.4 Deadlocks

Consider the parallel threads $t1$ and $t2$ using the following common variables and semaphores:

```
int x = 0, y = 0, z = 0;
semaphore lock1 = 1, lock2 = 1;
```

```
int t1() {
  z = z + 2;
  lock1.wait();
  x = x + 2;
  lock2.wait();
  lock1.signal();
  y = y + 2;
  lock2.signal();
}
```

```
int t2() {
  lock2.wait();
  y = y + 1;
  lock1.wait();
  x = x + 1;
  lock1.signal();
  lock2.signal();
  z = z + 1;
}
```

a. Executing the threads in parallel could result in a deadlock. Why?

b. What are the possible values of x, y and z in the deadlock state?

c. What are the possible values of x, y and z if the program terminates successfully (i.e., without a deadlock)? *Hint:* Remember that an assignment `z = z + 1` consists of multiple atomic operations on x.