# Concurrent Programming

## Exercise Booklet 1: Transition Systems

Note: For this booklet you must assume that assignment is atomic and that the scheduler is fair. Solutions to selected exercises (♢) are provided at the end of this document. Important: You should first try solving them before looking at the solutions. You will otherwise learn **nothing**.

**Exercise 1. (♢)** Assume that the **print** command is atomic. Build the transition system and then exhibit all possible paths of execution of the following program:

```
Thread.start { //P
  print("Hi")
  print("Alice")
}
```

```
Thread.start { // Q
  print("Hey")
  print("Bob")
}
```

**Exercise 2.** Draw the transition system for the following programs. Use "?" for the value of uninitialized variables. What values can $x$ take at the end of the execution?

1.

```
int x = 0
Thread.start { //P
  int local = x
  local = local + 1
  x = local
}
```

```
Thread.start { //Q
  int local = x
  local = local + 1
  x = local
}
```

2.

```
int x = 0
Thread.start { //P
  x = x + 1
}
```

```
Thread.start { //Q
  x = x + 1
}
```

**Exercise 3. (♢)** Given the following program:

```
int x = 0
int y = 0
Thread.start { //P
  y = x + 1
}
```

```
Thread.start { //Q
  x = y + 1
}
```

1. Show an execution path whose endstate holds $x = 2$ and $y = 1$.
2. Is there a path with an endstate holding s.t. $x = y = 1$? Justify your answer. What would happen if the assumption on atomicity of assignment is dropped?

**Exercise 4.** Draw the transition system for the following program:

```
int x = 0
Thread.start { //P
  2.times {
        x = x+1
    }
}
```

```
Thread.start { //Q
  x = 7
}
```

**Exercise 5.** Given the following program:

```
int n = 0
Thread.start { //P
  int local
  5.times {
    local = n
    n = local + 1
} }
```

```
Thread.start { //Q
  int local
  5.times {
    local = n
    n = local + 1
} }
```

Show an execution path whose endstate holds the value 5 for $n$.

**Exercise 6.** Assume that $f$ has an integer root, i.e., $f(x) = 0$ for some integer. We now propose two different programs for finding this root. We consider a program to be correct if, in the case that $f$ does have a root, both threads terminate and $x$ holds the root. For each program indicate whether it is correct or not, justifying your answer by exhibiting appropriate paths.

- Program A:

```
boolean found = false
Thread.start { //P
  int i = 0
  while (!found) {
    i = i + 1
    found = (f(i) == 0)
} }
```

```
Thread.start { //Q
  int j = 1
  while (!found) {
    j = j - 1
    found = (f(j) == 0)
} }
```

- Program B:

```
boolean found = false
Thread.start { //P
  int i = 0
  while (!found) {
    i = i + 1
    if (f(i) == 0)
      found = true
} }
```

```
Thread.start { //Q
  int j = 1
  while (!found) {
    j = j - 1
    if (f(j) == 0)
      found = true
} }
```

**Exercise 7.** Consider the program:

```
int n = 0
Thread.start { //P
  while (n < 2)
    print(n)
}
```

```
Thread.start { //Q
  n = n + 1
  n = n + 1
}
```

1. Supply the execution paths that print the following sequences: 012, 002, 02.
2. Should 2 necessarily appear in the output?
3. How many times can 2 appear in the output?
4. How many times can 1 appear in the output?
5. How many times can 0 appear in the output?
6. What is the length of the shortest sequence that can be exhibited?

**Exercise 8.** Consider the program:

```
int n = 0
Thread.start { //P      Thread.start { //Q
  while (n < 1)          while (n >= 0)
    n = n + 1              n = n - 1
}                      }
```

1. Provide an execution path in which the loop in the thread on the left is executed exactly once.
2. Provide a path in which the loop in the thread on the left is executed exactly three times.
3. Describe a path in which the loop in the thread on the left does not terminate.

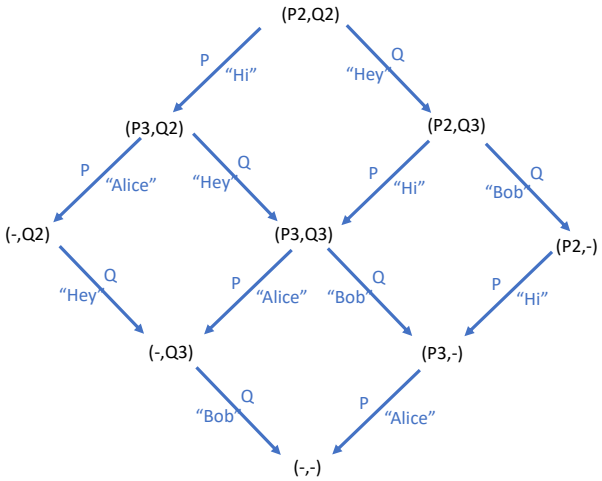
**Exercise 9.** Consider the program:

```
int n = 0
boolean flag = false
Thread.start { //P        Thread.start { //Q
  while (!flag) {          while (!flag) {
    n = 1 - n                if (n == 0)
  }                             flag = true
}                          } }
```

1. Provide an execution path in which the program terminates.
2. What are the possible values of $n$ when the program terminates.
3. Can the program not terminate?

# 1 Solutions to Selected Exercises

**Answer to exercise 1**




The execution paths are all the paths in the graph that start at the startstate $(P2, Q2)$ and end in the endstate $(-, -)$.

**Answer to exercise 3**

1. State format $(IP\_P, IP\_Q, x, y)$. Path: $(P2, Q2, 0, 0) \xrightarrow{P} (-, Q2, 0, 1) \xrightarrow{Q} (-, -, 2, 1)$
2. There are no paths that result in x and y both holding one. If the assumption on atomicity of assignment is dropped, then there would be such a path.