Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik, Anja Rey

Exercise Sheet 3

for the lecture on

Advanced Programming and Algorithms

Submission until **Monday, 6th November, 12:30 pm**.

Discussion in the exercise classes on 13th, 16th and 17th November, 2023.

Problem 1 to hand in: *Loop Invariant*

The following algorithm computes the symmetric difference $A \Delta B = (A \setminus B) \cup (B \setminus A)$, given two input sets A and B.

`get_symmetric_difference(`A, B`)`:

```
1 C ← B
2 for a ∈ A do
3 |  if a ∈ C then
4 |  |  C ← C \ {a}
5 |  if a ∉ B then
6 |  |  C ← C ∪ {a}
7 return C
```

a) State a loop invariant that holds at the beginning of each iteration of the for loop (lines 2 to 4).

b) Proof this loop invariant.

c) Use this loop invariant to show that indeed the algorithm returns the symmetric difference.

Problem 2 for discussion: *Basic Algorithm Design*

Design an algorithm for the following basic problem:

Given a matrix of non-negative integers, compute the number of occurences of positive numbers for each row of the matrix, and compute the mean for each row of the matrix.

a) Which data structure can you use to use to represent the input matrix?

b) Describe an algorithm that solves this problem intuitively.

c) Formulate the algorithm in Pseudocode.

d) Analyse the asymptotic worst-case running time of your algorithm.

e) Prove that the algorithm is correct.

Problem 3 as a programming exercise: *Unit Tests*

Consider the following problems. For each problem,

- write tests that can help you verify representative examples and properties. How many tests are reasonable?
- write a function that passes your tests. Can you improve the structure of your function, e.g. by extracting subfunctions?

a) Given a positive integer a, return a Boolean value that decides whether a is a prime number.

b) Given a positive rational number $b > 0$ and a positive integer n, compute the partial sum $\sum_{i=0}^{n} b^i$ of the geometric series.

c) Given a non-negative integer c, multiply c with a random integer m, $1 \leq m \leq 100$.

d) Given a non-negative integer d, return the number of `1` entries in the corresponding binary representation of d.

Problem 4 as a programming exercise: *Basic Code Quality Habits*

For the next code you write — here or in another context — try out the following habits:

- For each function you write, first think of a representative unit test; even better: write that test (e.g., simulate with `assert` in jupyter notebook). Afterwards write and test your function.
- Refactor your code: consider expressive naming, spacing conventions. In particular, break functions down into subfunctions, extract functions with a single purpose.
- Make sure your tests still pass.

Problem 5 for discussion: *Queue*

A queue is a data structure that follows the *first-in-first-out* principle: The first element that has entered the queue (waits the longest), is attended to first.

A queue needs the following methods:

initialise(): Create a new queue,

enqueue(x): insert a new element with key x into the queue in $\mathcal{O}(1)$,

dequeue: delete and return the oldest element from the queue in $\mathcal{O}(1)$.

a) Describe a way to realise a queue with the help of a linked list. Describe each of the three methods `initialise()`, `enqueue(x)`, and `dequeue()`. Explain why all operations work in constant running time.

b) Discuss: Can queues also be realised with arrays? If so, how? Which conditions do we need? What are the running times of the three operations?