This exercise sheet covers different topics of the course to prepare you for the exam. Throughout this exercise sheet, let m be your Matrikelnr. having 8 digits $m_1m_2m_3m_4m_5m_6m_7m_8$ with $0 \leq m_i \leq 9$. **Start each document handed in with (writing down) your name and m.**

Note: For this solution, we use various different Matrikelnrs.

1) a) For $k = m_7 + 2$, consider k-mergesort, the variation on mergesort that sorts a list ℓ of length greater than 1 by splitting it into k lists $\ell_1, \ldots, \ell_k$ of equal lengths, recursively k-mergesorts $\ell_1, \ldots, \ell_k$ to yield $s_1, \ldots, s_k$, and then does a k-way merge of the $s_1, \ldots, s_k$ to yield a sorted list s.

   Analyse the (time) complexity $T(n)$ of k-mergesort, for n the length of the input-list. You may restrict your analysis to lists whose length is a power of k (so that in each recursive call all k parts indeed do have equal lengths), and you may assume that a k-way merge takes time linear in the length of the merged lists.

   b) Does there exist a natural number x such that $x \equiv m_2 \pmod{m_3 + 2}$ and $x \equiv m_4 \pmod{m_3 + 3}$? If so, compute such an x by applying Bézout's Lemma/the Chinese Remainder Theorem. If not, argue why not.

   c) Show that the set $LP = \{M\#x \mid \text{TM } M \text{ loops on input } x\}$ is not recursive, where you may assume that (the code of) M and x are bit-strings.

Solution: In this exercise, we use the Matrikelnr. 14658571.

a) Given our Matrikelnr. $k = 9$, so we consider 9-mergesort. By the specification, the time-complexity expressed in term of the length n of the list, is then given by the recurrence $T(n) = 9 \cdot T(\frac{n}{9}) + n$ if n is a positive power of 9, and 1 otherwise. Since this recurrence is of the shape required by the master theorem for $a = b = 9$ and $s = 1$, and since we are in its second case since $a = b^s$, we obtain $T(n) \in \Theta(n \log n)$.

Note that in fact the result here is independent of k, i.e. of the Matrikelnr. (with the $\ldots + 2$ in the definition of k guaranteeing that $a, b > 1$ even if the digit m_7 would be 0 or 1).

b) The moduli $m_3 + 2$ and $m_3 + 3$ are coprime. This can be seen e.g. by running Euclid's algorithm, i.e. $\gcd(m_3 + 2, m_3 + 3) = \gcd(m_3 + 2, 1) = 1$. The CRT then already tells us that there exists a solution. Now let us compute the solution.

For our example Matrikelnr. the question becomes whether there exists a natural number x such that $x \equiv 4 \pmod 8$ and $x \equiv 5 \pmod 9$.

To use the CRT, note that since the difference between 8 and 9 is 1, we obtain a particularly simple instance of Bézout's lemma: $1 = -1 \cdot 8 + 1 \cdot 9$ and by the Chinese Remainder Theorem, we get the solution $x = -1 \cdot 8 \cdot 5 + 1 \cdot 9 \cdot 4 = -4$.

Of course, this means that *every* number x with $x = -4 \pmod{(8 \cdot 9)}$ is also a solution (e.g. 68).

c) LP is essentially the complement of HP except that the complement of HP also contains "garbage strings" that either do not contain the symbol "#" at all or contain it multiple times.

This means that if *LP were* recursive, we could use it to decide *HP* for a word w by simply checking if w contains exactly one "#" symbol, then checking whether $w \in LP$, and returning "True" if and only if the former holds and the latter does not.

2) a) Let k be the number m_3m_4 and k' be m_5m_6 in decimal notation.

Compute $\text{lcm}(k, k')$ by first computing $\gcd(k, k')$ by means of the Euclidean algorithm, giving the intermediate steps. (You may choose either the subtraction-based or the division-based version; indicate which version you use.)

b) Let k be the number $2 + m_2$ in decimal notation.

Consider all functions from the set $N = \{n \mid 0 \leq n < k \cdot (k+1)\}$ of natural numbers to the set $P = \{(x, y) \mid 0 \leq x < k,\ 0 \leq y < k+1\}$ of pairs of natural numbers.

i) Show that $|N| = |P|$ by giving some bijection between N and P.

ii) How many injective functions from N to P are there? Since the number is typically large, it suffices to explain how it is computed.

iii) Is the function $f(n) = (n \bmod k,\ n \bmod (k+1))$ a bijection from N to P? If so, explain why. If not, show how bijectivity fails.

c) Let k be your Matrikelnr. m interpreted as a number in decimal notation. Does there then exist an inverse of k modulo 15? If so, compute the inverse. If not, explain why not. (You may use a calculator for intermediate steps, say for modulo computations, but you have to explain your method.)

Solution:

a) We give sample computations for $m_3 = 4$, $m_4 = 4$, $m_5 = 1$, $m_6 = 8$, so $k = 44$ and $k' = 18$. Using the subtraction-based version of Euclid we find $\gcd(44, 18) = \gcd(26, 18) = \gcd(8, 18) = \gcd(8, 10) = \gcd(8, 2) = \gcd(6, 2) = \gcd(4, 2) = \gcd(2, 2) = 2$. Thus (see the lecture) $\text{lcm}(k, k') = \frac{k \cdot k'}{\gcd(k,k')} = \frac{44 \cdot 18}{2} = 44 \cdot 9 = 396$.

b) We give sample calculations for $m_2 = 4$, so $k = 6$.

- Both N and P have $k \cdot (k+1) = 42$ elements, and (as seen in the lecture) the function $n \mapsto (n/(k+1), n \bmod k+1)$ is a bijection with inverse $(x, y) \mapsto x \cdot (k+1) + y$ since $n = (n/(k+1)) \cdot k + 1 + (n \bmod k+1)$ and $(x, y) = ((x \cdot (k+1) + y)/(k+1), (x \cdot (k+1) + y) \bmod k+1)$, where / is integer division (`div` in Haskell).
  But note that also the function in the third item is a bijection.
- The number of injections is computed via the falling factorial, which since $\#N = \#P$, here is just the factorial $\#N!$ (see the lecture). (Already in our case this number is huge; it has 52 digits.)
- Yes, f is a bijection as follows from the CRT (see the lecture), since for any $k > 1$, k and $k+1$ are coprime.

  Alternatively, we may check that for each input in N the output is in P and all such outputs are pairwise distinct. E.g. the latter may be checked in ghci by first computing the list of outputs by evaluating

  ```
  map (\n -> (n `mod` 6,n `mod` 7)) [0..41]
  ```

  and then see there are no duplicates by checking `it == Data.List.nub it` evaluates to `True`.

c) The answer depends on the value of k.

If k is divisible by 3 or 5 (e.g. $k = 84713562$ is divisible by 3 since the sum 36 of its digits is), it does not have an inverse since (see the lecture) a number k is invertible modulo 15 iff $\gcd(k, 15) = 1$ iff neither 3 nor 5 divides k.

If k is divisible by neither 3 nor 5 (e.g. evaluating `84713561 ‘mod‘ 15` in Haskell yields 11 which isn't divisible by either), then we may proceed by Bézout's lemma. E.g. starting from 11 as above it finds u, v such that $11 \cdot u + 15 \cdot v = 1 = \gcd(11, 15)$: starting from $11 \cdot 1 + 15 \cdot 0 = 11$ and $k \cdot 0 + 15 \cdot 1 = 15$, we successively find first $11 \cdot (-1) + 15 \cdot 1 = 4$, then $11 \cdot 3 + 15 \cdot (-2) = 3$, and finally $11 \cdot (-4) + 15 \cdot 3 = 1$, so $u = -4$ and $v = 3$. Hence $11 \cdot (-4) \equiv 1 \pmod{15}$. We conclude by choosing the natural number 11 as representative of -4 modulo 15.

Alternative solution: Since 15 is a small number we can simply find the inverse of any number n by brute force, generating all $i \cdot n \mod 15$ for $0 \leq i < 15$ and testing for which of those i this results in 1. E.g. evaluating

```
[a | a <- [0..14], a * 11 ‘mod‘ 15 == 1]
```

in Haskell yields `[11]` i.e. the same answer as above.

More alternative solutions: Note that since 15 is not a prime number, Fermat's Little Theorem does not apply here. However, since 3 and 5 are prime, Euler's Theorem (which we know from RSA) says that $a^{(3-1)(5-1)} \equiv 1 \pmod{15}$ for any a coprime to 15. Thus $11^8 \equiv 1 \pmod{15}$ and 11^7 is a multiplicative inverse of 11 modulo 15. Since $11^7 \bmod 15 = 11$, this is consistent with the answer we got above.

3) a) Let G be a directed graph with nodes $\{a, b, c, d\}$ and labeled edges

$$\{(a, m_2, b), (a, m_3, c), (a, m_4, d), (b, m_5, a), (b, m_6, a), (c, m_7, d), (d, m_8, c)\}$$

where each triple describes the start of the edge, the weight, and the end of the edge. Use Floyd's algorithm to compute the distances (least weight among possible paths) between all nodes. Give the start matrix and all intermediate matrices, and give the distance from b to d.

b) Let G be the graph with nodes $\{a, b, c, d, e, f, g\}$ and edges

$$\{a, b\}, \{a, e\}, \{b, e\}, \{c, d\}, \{c, g\}, \{d, g\}, \{d, f\}, \{f, g\}$$

The weights of the edges in this order are $m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8$. Use Kruskal's algorithm to compute a spanning forest with minimal weight. Show all the intermediate steps and the final spanning forest.

c) Consider the relation R on digits $\{0, \ldots, 9\}$, defined by $R(d, e)$ if and only if no instance of the digit d appears anywhere after any instance of the digit e in the word m. Formally:

$$R = \{(d, e) \mid 0 \leq d, e \leq 9,\ \nexists u, v, w.\ m = uevdw\}$$

For example, in the word 121, we have $2R2$, $3R4$, $\neg 1R1$, $\neg 1R2$, $\neg 2R1$.

Is R an equivalence relation? If so, prove it. If not, give for **each** property (among the 3 properties equivalence relations have) that is not satisfied a counterexample.

Solution: In this solution we assume that the digits $m_1 \ldots m_8$ are 12343456.

a) Ordering the nodes as a, b, c, d we obtain matrices:

$$\begin{pmatrix} 0 & 2 & 3 & 4 \\ 3 & 0 & \infty & \infty \\ \infty & \infty & 0 & 5 \\ \infty & \infty & 6 & 0 \end{pmatrix} \to_a \begin{pmatrix} 0 & 2 & 3 & 4 \\ 3 & 0 & 6 & 7 \\ \infty & \infty & 0 & 5 \\ \infty & \infty & 6 & 0 \end{pmatrix} \to_b \begin{pmatrix} 0 & 2 & 3 & 4 \\ 3 & 0 & 6 & 7 \\ \infty & \infty & 0 & 5 \\ \infty & \infty & 6 & 0 \end{pmatrix} \to_c$$

$$\begin{pmatrix} 0 & 2 & 3 & 4 \\ 3 & 0 & 6 & 7 \\ \infty & \infty & 0 & 5 \\ \infty & \infty & 6 & 0 \end{pmatrix} \to_d \begin{pmatrix} 0 & 2 & 3 & 4 \\ 3 & 0 & 6 & 7 \\ \infty & \infty & 0 & 5 \\ \infty & \infty & 6 & 0 \end{pmatrix}$$

From the final matrix we read off that the distance from b to d is 7.

b) We list the edges in the order of increasing weights and proceed with merging the connected components:

k_i	$b(k_i)$	connected components
		$\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}$
$\{a, b\}$	1	$\{a, b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}$
$\{a, e\}$	2	$\{a, b, e\}, \{c\}, \{d\}, \{f\}, \{g\}$
$\{b, e\}$	3	
$\{c, g\}$	3	$\{a, b, e\}, \{c, g\}, \{d\}, \{f\}$
$\{c, d\}$	4	$\{a, b, e\}, \{c, d, g\}, \{f\}$
$\{d, g\}$	4	
$\{d, f\}$	5	$\{a, b, e\}, \{c, d, f, g\}$
$\{f, g\}$	6	

The final computed spanning forest is: $\{a, b\}$, $\{a, e\}$, $\{c, g\}$, $\{c, d\}$, $\{d, f\}$.

c) For the Mat.Nr. 15658571: The relation is not reflexive, since $\neg R(3, 3)$.
The relation is not symmetric, since $R(6, 1)$ but $\neg R(1, 6)$.
The relation is not transitive, since $R(1, 7)$ and $R(7, 6)$, but $\neg R(1, 6)$.

4*) a) Prove the following statement if it is true or give a counterexample if it is false: Let M, N be countably infinite sets such that $M \subseteq N$. Then $N \setminus M$ is finite.

b) For languages L_1, L_2 over Σ, L_1 is said to be *reducible* to L_2, denoted by $L_1 \leq L_2$, if there exists a computable $f : \Sigma^* \to \Sigma^*$ such that $x \in L_1 \Leftrightarrow f(x) \in L_2$.

Explain how the notion of reducibility is typically used to show languages are not recursive, and why the condition that f be *computable* cannot be omitted from the definition (of reducibility), without rendering it useless for that usage.

c) Show that the language $L = \{M\#x \mid \text{TM } M \text{ never moves to the left on input } x\}$ is recursive.

Solution

a) Counterexample: $M = \{n \mid n \in \mathbb{N},\ n \text{ even}\}$ (the set of even natural numbers) and $N = \mathbb{N}$ are countably infinite. But $N \setminus M$ is the set of odd natural numbers, which is clearly not finite.

b) If $L_1 \leq L_2$ and L_1 is not recursive then L_2 is not recursive either. So typically to show that a language L_2 is not recursive, one tries to come up with a language L_1 that is

- already known to be non-recursive, e.g. the HP or the MP;
- is easily/conveniently reducible to L_2.

The condition that the reduction f be computable cannot be omitted, since otherwise we could show HP to be recursive by defining f to map $x \in \text{HP}$ to 0 and $x \notin \text{HP}$ to 1 and thus reduce HP to the set $\{1\}$.

In general, if $L_1 \leq L_2$ and L_2 is decided by a Turing Machine M_2 but we do not know whether f is computable, it need not be possible to construct a Turing Machine M_1 deciding L_1 (if f is computable, say by TM F, then such an M_1 *can* be constructed by composing F and M_2).

c) Consider a TM U that given an input first checks that it is of shape $M\#x$ and if not rejects, and if it is of that shape, simulates running M on x as follows. Throughout the simulation, U rejects immediately if M moves to the left. Otherwise, if M halts, U accepts.

The remaining case is that M runs forever without ever moving to the left. In this case, M will eventually reach the empty part of the tape. From that point on, M will only ever see the symbol $\sqcup$. Since there are only finitely many states, M will visit some state q twice after at most $n+1$ steps (where n is the number of states of M) by the pigeonhole principle. Once that happens, M has clearly entered an infinite loop since it will always visit q again.

This means that once we see a $\sqcup$ symbol, we can simulate M for $n+1$ more steps and if it has not terminated by then anyway, we can stop the simulation and accept.

From this description it is then clear that $L(U) = L$.