Lecture 9 — Sequences

1. Assuming that $s, t, u : \mathrm{seq}\, Char$ are defined as follows:

$$s == \langle a, c, k \rangle$$
$$t == \langle b, l \rangle$$
$$u == \langle j \rangle$$

   evaluate the following expressions:

   (a) $(t \frown s) \frown (u \frown s)$
   (b) $\mathrm{head}(\mathrm{front}(t \frown s))$
   (c) $\mathrm{last}(\mathrm{front}(t \frown s))$
   (d) $\mathrm{tail}(\mathrm{front}(t \frown s))$
   (e) $s \frown t$
   (f) $\mathrm{rev}(s \frown t)$
   (g) $\mathrm{dom}(s \frown t)$
   (h) $\mathrm{ran}(s \frown t)$
   (i) $(s \frown t) \rhd \{a, e, i, o, u\}$
   (j) $(t \frown s) \oplus \{3 \mapsto o\}$

   HINT: Remember that a sequence

$$\langle a, b, c \rangle$$

   is really just a function

$$\{1 \mapsto a, 2 \mapsto b, 3 \mapsto c\}.$$

2. For each of the sequences resulting from expressions in question 1, write down the corresponding function.

3. Using sequences, develop a specification for a *stack of natural numbers.* The stack should be of limited size (say 1000 elements), and should have the following operations defined:

   - *push* — push an item onto the top of the stack;
   - *pop* — remove the top item from the stack;
   - *top* — obtain the top item from the stack, without changing the stack;
   - $full$ — returns a variable $isfull!$ which contains $true$ if the stack is full, $false$ otherwise;
   - *empty* — as for $full$, but for finding out whether the stack is empty.

Your specification should be *robust*, in that it should capture error situations (such as trying to push an item onto a full stack).

4. Using sequences, develop a specification for a *queue of natural numbers*. The queue should have a limited size (say 1000 elements), and should have the following operations defined:

   - *enqueue* — puts an item on the end of the queue;
   - *dequeue* — takes an item off the front of the queue (leaves the queue unaltered);
   - *head* — takes the first item of the front of the queue;
   - $full$ — as for the stack;
   - *empty* — as for the stack.