CSCM12 – Software concepts and efficiency April 7th 2025
Manjiri Joshi, Cécilia Pradic & Deshan Sumanathilaka

Lab 8: hashing

This is the las lab of the term. The goal of this lab will be to play with hash tables and hashing. It will be more about using them than implementing datastructures.

You will need to review some of the documentation for the class `Hashtable` `https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/util/Hashtable.html`, and at some point you will be asked to give an implementation of `hashCode`, which is documented at `https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/Object.html#hashCode()` .

For the challenge task, you should edit the file `Countdown.java`

1. **Inverting an array** Write a function

   ```
   public static Hashtable<String, Integer> invert(int[] arr)
   ```

   returning a hashtable `h` such that, for any string `s`, `h.get(s)` returns the first index `i` of `arr` such that `arr[i].equals(s)` holds.

2. **Count sort**

   (a) Write a function

   ```
   public static void countSortInterval(int[] arr)
   ```

   which sorts arrays A of size n containing positives integers that can range from, say 0 to K, in time $\mathcal{O}(nK)$ (in particular, for the class of inputs where K is a constant, this is linear in the size of the array).

   (b) Write a function

   ```
   public static void countSort(int[] arr)
   ```

   which sorts arrays A of size n containing at most K distinct values in time $\mathcal{O}(nK)$ on average using a hashtable.

3. **Java's hashCode of lists of lists** The method `hashCode` of `LinkedList<T>` is implemented in a way which is equivalent to the following:

   ```
   public int hashCode()
   {
     int res = 1;
     for(T e : self)
       res = res * 31 + e.hashCode();
     return res;
   }
   ```

On integers, `hashCode` is the identity, i.e., it just returns the value of the integer to be hashed.

(a) What is `hashCode` of the list `[0, 1]`? Of the list of lists `[[0],[]]`?

(b) **Challenge:** Exhibit a collision for the `hashCode` method of the class `LinkedList<LinkedList<Integer>>`.

4. **Ultimate challenge task: Countdown** *Countdown* is a British[1] TV game show that is still airing nowadays. In one of the games, the contestants are given a list of numbers, as well as a target number, and attempt to reach the target number by applying the basic arithmetic operations `+`, `*`, `-` and `/` using the numbers they are given. There are some natural restrictions in how they can use those numbers:

   - Negative or fractional numbers are not allowed at any stage of a computation. So for instance, $(3 - 5) + 10$ and $5/2 + 3/2$ are never valid answers.
   - All numbers given to reach the target must be used at most once in the computation. For instance, if the list $5, 5, 4, 3, 2$ is given, $5 + 5 - 4$ and $3 * 2$ are valid answers to reach 6, but $5 - 4 + 3 + 4 - 2$ is not.

   Write the body of the method

   ```
   public static ArithmeticExpression countdown(LinkedList<Integer> pbm)
   ```

   that returns an arithmetic expression which is a solution of the countdown problem for the list `pbm` if it exists, and `null` otherwise. Feel free to write your own auxiliary methods! In particular, you may use the method

   ```
   public static LinkedList<LinkedList<Integer>>[]
                       allSplits(LinkedList<Integer> xs)
   ```

   which is provided. It enumerates all possible ways of splitting a list of elements `xs` in two sublists `ys` and `zs` different from `[]` that enumerate the same elements as `xs` and, say, in the same order in both lists as in `xs`; it does so by enumerating all of the `ys` and `zs` side-by-side in even and odd positions respectively. For instance, `allSplits` applied to `[3,2,1,0]` returns

   ```
   [[3, 2, 0], [1], [2, 0], [3, 1], [3, 0], [2, 1], [0], [3, 2, 1]]
   ```

   For the function `countdown` to be efficent enough to tackle realistic examples, you should use a hashtable whose keys are input lists of integers and values are outputs to memoize intermediate results.

[1] Actually a French export; not sure where else TV shows with the same conceit air.