Assignment 1

. Define a function isPerfect :: Integer -> Bool that checks if the given input (a positive integer) is a *perfect*

*number*. A positive integer is perfect if it is the sum of all its proper divisors.

. Define a function nextPerfect :: Integer -> Integer such that for each positive integer *n*, nextPerfect n

returns the least perfect number *m* > *n*.

. Define a function partitioned :: [Int] -> Bool that returns True if there is an element *n* of the list such

that:

• for each element *m* occurring before *n* in the list, *m* B *n*, and

• for each element *m* occurring after *n* in the list, *m* > *n*.

Sample cases:

partitioned [] = False

partitioned [22] = True

partitioned [19,17,18,7] = False

partitioned [7,18,17,19] = True

partitioned [19,13,16,15,19,25,22] = True

partitioned [19,13,16,15,25,19,22] = False

. Define a function connected :: [String] -> Bool that checks whether the input list of strings is *connected*.

A list of strings is connected iff:

• each string in the list (other than the first) is obtained from the previous one by changing the character

in *exactly one* position, and

• no string occurs twice in the list.

Sample cases:

connected [] = True

connected [”aa”, ”ab”, ”ba”] = False

connected [”aa”,”ab”,”bb”,”ba”] = True

connected [”aa”,”ab”,”bb”,”ba”,”aa”] = False