HW 8

Due Mar 8 by 11:59pm **Points** 35 **Submitting** a file upload

1. (5 points)Write a function in Scala that takes in two lists of Ints, and returns a new list of Ints with the elements of the input lists alternating. So if the input lists were 1, 2, 3, 4 and 7, 8, 9, 10, 11, 12 the output list would be 1, 7, 2, 8, 3, 9, 4, 10, 11, 12. Note that if one list is longer than the other, the extra elements are added at the end. For this function, you MUST use pattern matching, and you may NOT use any built-in list functions (isEmpty, head, tail, map, reduce, etc). You may NOT use a helper function.

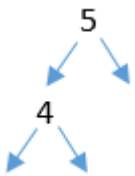
Please submit this problem with problem 2.

2. (5 points) Write a generic version of your original, uncurried reduce function that works for lists of any type that inherits from AnyRef.

Please name your file for this problem and problem 1 HW8P12Lastname.scala

3. (15 points) Add the following functions to the BSTree class we discussed

def depth:Int This function should return the number of values traversed between the root and the deepest leaf (inclusive). That is, the tree below has depth 2. You may NOT use pattern matching (the match/case syntax).



def max:Int This function should return the largest number in the BSTree, and -1 if the BSTree is empty.

def exactsubtree(that:BSTree):Boolean This function should return true if this appears in the exact same structure as a subtree of that (down to the Leaf references). You MUST use pattern matching; you may not use isLeaf.

Please name your file for this problem HW8P3Lastname.scala

4. (10 points) This problem will use the notion of representing a set using its characteristic function that we discussed in class Friday 3/3 (all the code is posted there). You will build on the contains, union, intersect, and diff functions that we defined in class.

- Define a "factory" function def setlist(List[Int]):Int=>Boolean which takes in a list of Ints and returns a set consisting of all Ints in the list. You may not define a helper function with extra parameters.

- Define the function def filter(s: Int=>Boolean, p:Int=>Boolean): Int=>Boolean which takes in a set s and a predicate p, and returns a new set that consists of the elements of s that satisfy the predicate. An example of a predicate is: P(x) = x%2==0. Predicates are often used with quantifiers in logic. You MUST define your inner function with an anonymous function, and you may not define a helper function with extra parameters.
- Define a function def forall(s:Int=>Boolean, p:Int=>Boolean):Boolean that tests whether a predicate p is true for all elements of a set s. In order to make it possible to implement this function, we will consider a predicate true for all integers if it is true for integers from -1000 to 1000. Another way to think of this: This function answers the question: Is the predicate true for all elements of the subset of s that consists only of integers between -1000 and 1000
  You may create a helper function with one parameter.

Please name your file for this problem HW8P4Lastname.scala.