CPSC 335 Spring 2023

Homework 1

Due on 03/01 by 11:59 PM on Canvas

Q1. (30): Each of the following snippets of pseudocode fails to live up to all of the clarity, correctness, and termination requirements of algorithms. In each case, describe the problem, and then rewrite the pseudocode as a proper algorithm.

(b) def long_division(num, denom):
quotient = num // denom

remainder = num % denom

Error: there is nothing being returned by this function. One "/" line

Correct pseudocode:

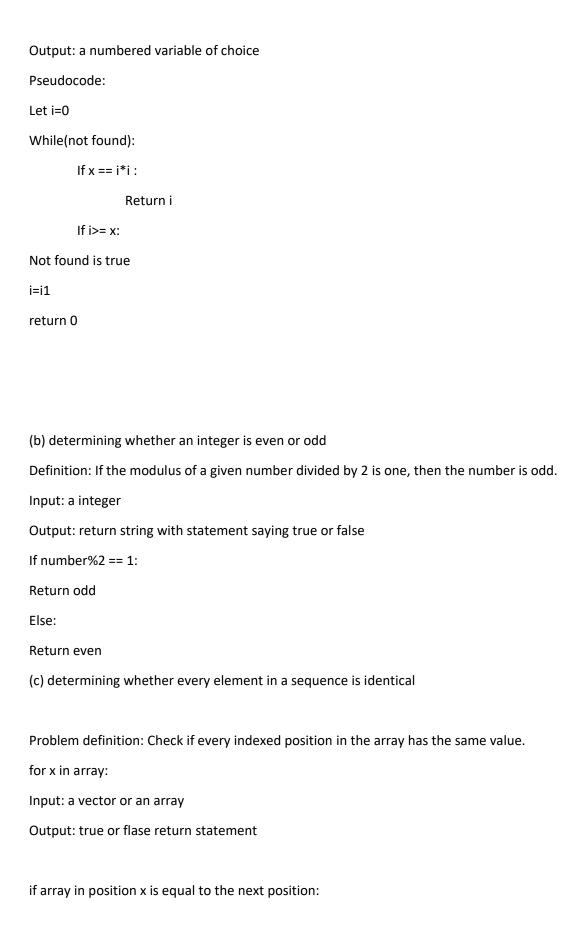
def long_division(num, denom):

quotient = num / denom

remainder = num % denom

return (quotient,remainder)

```
(c)
def keep_positives(S):
if len(S) == 0:
        return 0
else:
        result = []
for x in S:
        if x > 0:
                result.add(x)
return result
Error: the error is in defining result. The .add() function assumes for adding an integer. The append
function will then correctly add the integer to the string result.
Correct Pseudocode:
def keep_positives(S):
if len(S) == 0:
        return 0
else:
        result = []
for x in S:
        if x > 0:
                result.add(x)
return result
Q2. Exercise (40): Write a problem definition and pseudocode for each of the following problems.
(a) computing a square root
Definition: Computing a square root of a number is to find if it multiplies by itself.
Input: a number variable
```



a boolean equals true
else:
the boolean is false
(d) determining whether two strings are identical Input: two strings Output: a boolean return statement
if string1 length is equal to string2:
for i in the length of the string:
if the string1 at i is the same char as string2 at i:
bool is true
else:
bool is false
else:
bool is false
Q3 (30). Consider the following algorithm:
Algorithm Calc(a, n):
Input: two integers, a and n
Output: ?
k <- 0
b <- 1
while k < n do
k <- k + 1
b <- b * a

return b

(a) What does the algorithm calculate?

For a repetitive about of times up to the value of n. Multiply b by a and insert into b. Then return the final value of b after the loop ends.

(b) . Analyze its worst-case running time and express using Big-Oh notation.

Using step count: O(n) is the worst case running time.