Week 4 – Adapting the Binary search algorithm

Here is my Pseduo-code:

BINARY\_SEARCH(alist, low, high) (1)

First <- 0 (1)

last <- length(alist) – 1 (1)

found <- False (1)

while first<= last and not found: (N)

midpoint <- (first + last)//2 (N)

if alist[midpoint] > low and alist[midpoint] < high: (N)

found <- True (N)

else: (N)

if high < alist[midpoint]: (N)

last <- midpoint-1 (N)

else: (N)

first <- midpoint+1 (N)

return found (N)

low <- 33 (1)

high <- 35 (1)

testlist <- [0, 1, 2, 8, 13, 17, 19, 32, 42,] (1)

The bigO time complexity for this solution is 10N + 7. Once we remove the constant and the multiplier, we have a run time of O(n).

Here is my adapted code:

def binarySearch(alist, low, high):

first = 0

last = len(alist)-1

found = False

while first<=last and not found:

midpoint = (first + last)//2

if alist[midpoint] > low and alist[midpoint] < high:

found = True

else:

if high < alist[midpoint]:

last = midpoint-1

else:

first = midpoint+1

return found

low = 33

high = 35

testlist = [0, 1, 2, 8, 13, 17, 19, 32, 42,]

print(binarySearch(testlist, low, high))