# Homework 3 Searching and sorting

1. The binary search is an efficient algorithm for finding an item in a list.

(a) Under what circumstances is it not possible to carry out a binary search? [1]

(b) An algorithm for a binary search is given below.

SUB binarySearch(aList, itemSought)

found 🡨 False

index 🡨 -1

first 🡨 0

last 🡨 len(aList)-1

WHILE first <= last AND found = False

midpoint 🡨 Integer part of ((first + last)/2)

IF aList[midpoint] = itemSought THEN

found 🡨 True

index 🡨 midpoint

ELSE

IF aList[midpoint] < itemSought THEN

first 🡨 midpoint + 1

ELSE

last 🡨 midpoint - 1

ENDIF

ENDIF

ENDWHILE

RETURN index

ENDSUB

(i) What will be the value of index returned if the item is not found? [1]

(ii) What is the maximum number of items that will need to be examined   
to determine whether an item is in a list of size n? [1]

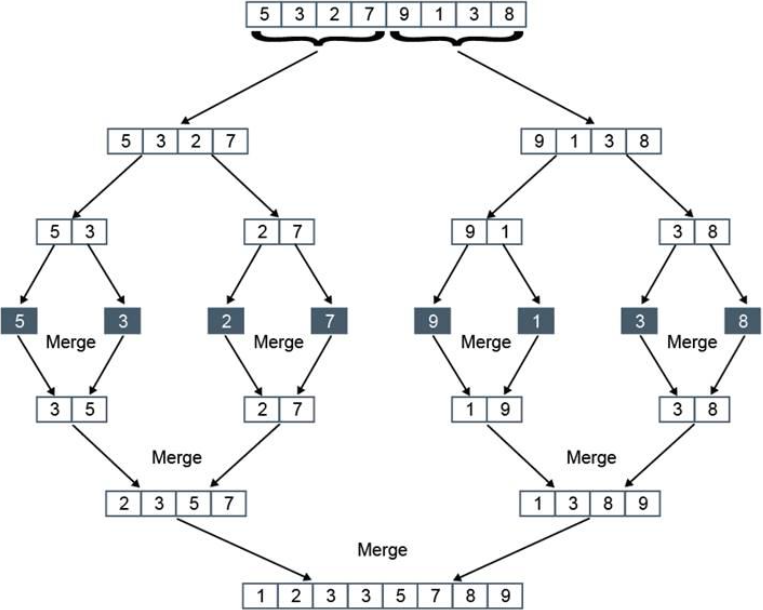
(c) A binary search is carried out on the following list of names:

|  |  |
| --- | --- |
| 1 | Ava |
| 2 | Chloe |
| 3 | Ella |
| 4 | Evie |
| 5 | Grace |
| 6 | Imogen |
| 7 | Isabella |
| 8 | Jessica |
| 9 | Lily |
| 10 | Mia |
| 11 | Ruby |
| 12 | Scarlett |

(i) List the names that will be examined when searching for Mia. [3]

(ii) What is the order of time complexity of the binary search method? [1]

2. (a) The diagram below shows an algorithm in graphical form. What is the algorithm depicted? [1]



(b) Describe briefly how the algorithm works. [5]

(c) Which of the following is the order of time complexity of the algorithm?

O(log2n) O(n2) O(n log2n) O(n) [1]

3. A binary search tree is considered generally to have an order of time complexity of O(log2n).

Under what circumstances may its order of time complexity be O(n)? [1]

Total 15 marks