CST 363

Lab: In-memory search

1. An integer array x is defined as follows:

index:	1	2	3	4	5	6	7	8	9	10	11	12
value:	10	22	23	39	42	45	54	60	62	74	87	89

   In other words, x[1] is 10, x[2] is 22, etc., and the length of array x is 12.

   Manually simulate a binary search for value 87 in array b. Take index value 6 as your starting point. Write down the index value whenever it changes.

1. Below is a simple implementation of a binary search algorithm. Fill in the blanks.

```
# return the index of a in sorted array x
# (return -1 if a does not appear in x)
basic_binary_search(x, a) {
 lo = 1
 hi = length(x)
 # start searching in the middle of the array x
 i = floor((lo + hi)/2)
 while (lo <= hi && x[i]!=a) {
  if (a < x[i]) {

   _________
  } else {

   _________
  }
  i = floor((lo + hi)/2)
 }
 # if a appears in x, it must now be x[lo] or x[hi]
 if (x[i] == a)
  return(lo)
 else
  return(-1)
}
```

2. Simulate the algorithm on the following array x of 50 values. The red numbers are the index values, the blue numbers are the array contents. Pick some value for argument ‘a’ from among the blue values, and simulate execution of the algorithm. Try again with some value ‘a’ not in array x.

   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

2 3 4 9 10 11 13 20 22 23 24 26 30 32 34 35 36 37 38 39 42 43 45 48 49 51

27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
54 55 56 58 61 62 64 65 66 69 70 72 73 76 77 80 81 84 88 92 93 94 95 97