Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Practice Sheet #11 with Solutions

Topic: Searching Techniques

Date: 23-03-2017

1. (a) Consider the following C program that attempts to locate an element x in an array Y[] using binary search. The program is erroneous.

```
1.  f(int Y[10], int x) {
2.    int i, j, k;
3.    i = 0; j = 9;
4.    do {
5.          k =  (i + j) /2;
6.          if( Y[k] < x)  i = k; else j = k;
7.       } while(Y[k] != x && i < j);
8.    if(Y[k] == x) printf ("x is in the array ") ;
9.    else printf (" x is not in the array ") ;
10. }
```

On which of the following contents of Y and x does the program fail?
(a) Y is [1 2 3 4 5 6 7 8 9 10] and x < 10
(b) Y is [1 3 5 7 9 11 13 15 17 19] and x < 1
(c) Y is [2 2 2 2 2 2 2 2 2 2] and x > 2
(d) Y is [2 4 6 8 10 12 14 16 18 20] and 2 < x < 20 and x is even
Ans. c

(b) Consider the data given in above question, the correction needed in the program to make it work properly is
(a) Change line 6 to: if (Y[k] < x) i = k + 1; else j = k-1;
(b) Change line 6 to: if (Y[k] < x) i = k - 1; else j = k+1;
(c) Change line 6 to: if (Y[k] <= x) i = k; else j = k;
(d) Change line 7 to: } while ((Y[k] == x) && (i < j));
Ans. a

2. Which one of the following is the tightest upper bound that represents the time complexity of inserting an object into a binary search tree of n nodes?
(a) O(1)
(b) O(Logn)
(c) O(n)
(d) O(nLogn)
Ans. c

3. Consider the following C program segment.

```
while (first <= last)
{
  if (array [middle] < search)
     first = middle +1;
  else if (array [middle] == search)
     found = True;
  else last = middle – 1;
  middle = (first + last)/2;
}
if (first < last) not Present = True;
```

Run on IDE

The cyclomatic complexity of the program segment is _____.
(a) 3
(b) 4

(c) 5
(d) 6
Ans. c

4. Suppose you are provided with the following function declaration in the C programming language.

int partition (int a[], int n);

The function treats the first element of a[] as a pivot, and rearranges the array so that all elements less than or equal to the pivot is in the left part of the array, and all elements greater than the pivot is in the right part. In addition, it moves the pivot so that the pivot is the last element of the left part. The return value is the number of elements in the left part. The following partially given function in the C programming language is used to find the kth smallest element in an array a[ ] of size n using the partition function. We assume $k \leq n$

```
int kth_smallest (int a[], int n, int k)
{
  int left_end = partition (a, n);
  if (left_end+1==k)
  {
     return a [left_end];
  }
  if (left_end+1 > k)
  {
    return kth_smallest (_____);
  }
  else
  {
    return kth_smallest (_____);
   }
}
```

The missing argument lists are respectively:
(a) (a, left_end, k) and (a+left_end+1, n–left_end–1, k–left_end–1)
(b) (a, left_end, k) and (a, n–left_end–1, k–left_end–1)
(c) (a, left_end+1, N–left_end–1, K–left_end–1) and(a, left_end, k)
(d) (a, n–left_end–1, k–left_end–1) and (a, left_end, k)
Ans. a

--*--