Question 6 solution

There can be more than one correct solution, here is one:

/\*\* Assumes that this list is sorted in decreasing order.

\* Adds the given element to this list, while maintaining the list's order.\*/

public void addDecreasingOrder(int x) {

Node newNode = new Node(x);

Node prev = first;

Node current = first.next;

while ((current.value > x) && (current != null)) {

current = current.next;

prev = prev.next;

}

newNode.next = current;

prev.next = newNode;

size++;

}

Question 8 solution

There can be more than one correct solution, here is one:

/\*\* Returns the biggest integer that can be constructed from the digits of

\* the given integer. For example, if x = 231, returns 321. \*/

public static int biggestNumber(int x) {

int num = x;

LinkedList l = new LinkedList();

while (num > 0) {

l.addDecreasingOrder(num % 10);

num /= 10;

}

num = 0;

int n = l.size();

for (int i = 0; i < n; i++) {

num = 10 \* num + l.get(i);

}

return num;

}

Question 12 solution

There can be more than one correct solution, here is one.

The comments are not required, they help explain the solution.

// Reads a number, say x, and outputs the integer part of x/2.

load one // two = 2

store two

add one

store two

load zero // count = 0

store count

read // get x

store x

LOOP:

load x

gotoz END // if (x == 0) print count

sub one // if (x == 1) print count

gotoz END

load count // count++

add one

store count

load x // x = x - 2

sub two

store x

goto LOOP

END:

load count

write

stop

Question 10 solution

There can be more than one correct solution, here is one.

/\*\* Returns true if the first array is a sub-array of the second array. \*/

public static boolean subArr (int[] arr1, int[] arr2) {

return subarr (arr1, arr2, 0, 0);

}

private static boolean subarr (int[] arr1, int[] arr2, int i, int j) {

if (i == arr1.length) return true;

if (j == arr2.length) return false;

if (arr1[i] == arr2[j]) return subarr (arr1, arr2, i + 1, j + 1);

return subarr (arr1, arr2, i, j + 1);

}