FBA

HW solutions

QUESTION 1

Question 1 :

Choose the correct frequency for every line as well as the total O of the following code:

```
1 int sum = 0;
2 for(int i = 0; i < n * n; i++)
3   for(int j = n - 1; j >= n - 1 - i; j--) {
4     sum = i + j;
5     System.out.println(sum); }
```

Line 1 :

- ○ A) 0
- ● B) 1
- ○ C) 2
- ○ D) n
- ○ E) None

QUESTION 2

Line 2 :

- A) $2n + 2$
- B) $n^2 + 1$
- C) $\log(n+1)$
- D) $n(n+3)/2$
- E) None

QUESTION 3

Line 3 :

- A) $n^2(n^2 + 1)/2$
- B) $2n^2 (n^2 + 2)/2$
- C) $n^2 + n^2 (n^2 + 1)/2$
- D) $n(n + 1)/2$
- E) None

Question completion status:

QUESTION 4

Line 4 & 5:

- A) $n(n^2 + 1)$
- B) $n^2(n^2 + 1)/2$
- C) $2n(n^2 - 1)/2$
- D) $n(n - 1)/2$
- E) None

QUESTION 5

Total O :

- A) $n\log n$
- B) n^2
- C) n^3
- D) n^4
- E) None

Question Completion Status:

Question 2 :

Choose the most appropriate answer:

1. $log(n^n) + n^n + 2^n$ is O( ):

- A) 2^n
- B) n^n
- C) $n!$
- D) $nlogn$
- E) None

QUESTION 7

2. $n^4/2^{log(n)} + nlog(n^{100})$ is O( ):

- A) n^2logn
- B) n^2
- C) $nlogn$
- D) n^3
- E) None

QUESTION 8

3.

Suppose that an algorithm $\mathcal{A}$ iterates over a list L of integers of size n. For the elements of L at even positions, $\mathcal{A}$ executes some $O(n)$ operations. For the elements at odd positions, $\mathcal{A}$ executes some $O(n^2)$ operations. Which running time *best describes* algorithm $\mathcal{A}$ in the best case?

- ○ A) $O(n\log n)$
- ○ B) $O(n)$
- ○ C) $O(n^2)$
- ○ D) $O(n^3)$
- ○ E) None

QUESTION 9

4.

Suppose $f(n)$ is $O(g(n))$, and $g(n)$ is $O(h(n))$. What is the best big-O for $f(n) * h(n)$?

- A) $O(h(n))$
- B) $O(f(n) * g(n))$
- C) $O(f(n) * h(n))$
- D) $O(f(n) + g(n))$
- E) None

QUESTION 10

Write the method **public static<T> void insertUnique(List<T> l, T e)**, user of the ADT `List`. The method takes a list `l` and an element `e` and inserts the element at the end of the list only if it is not already there.

Example 0.2. *If* l: $A \to B \to C$, *then after calling* `insertUnique(l, "C")`, *the list does not change. Calling* `insertUnique(l, "D")` *will make* l *be:* $A \to B \to C \to D$.

Line 1:

- `l.findFirst();`
- `while(!l.last()) {`
- `if(l.head != null) {`
- `if(l.empty()) {`
- None

QUESTION 11

Line 10

- l.insert(e);
- l.update(e)
- return 1;
- l.current = l.head = n; }
- None

QUESTION 12

Line 2

- l.current = l.head;
- l.insert(e);
- while(!l.last()) {
- if(l.retrieve().equals(e))
- None

Question Completion Status:

QUESTION 13

Line 3

- `l.findNext();`
- `return; }`
- `while(l.current != null) {`
- `if(l.retrieve().equals(e))`
- None

QUESTION 14

Line 4

- `l.insert(e);`
- `l.findFirst();`
- `if(l.current.data.equals(e))`
- `return;`
- None

QUESTION 15

Line 5

- while(!l.last()) {
- l.findNext(); }
- return;
- l.update(e);
- None

QUESTION 16

Line 6

- l.current = l.current.next; }
- while(l.findNext())
- if(!l.retrieve().equals(e))
- if(l.retrieve().equals(e))
- None

QUESTION 17

Line 7

- return;
- l.current.next = new Node <T>(e); }
- l.insert(e);
- if(!l.retrieve().equals(e))
- None

QUESTION 18

Line 8

- l.insert(e);
- l.findFirst();
- else {
- l.findNext(); }
- None

QUESTION 19

Line 9

- ○ `l.update(e);`
- ○ `if(!l.retrieve().equals(e))`
- ○ `l.findFirst();`
- ○ `Node<T> n = new Node <T>(e);`
- ○ None

QUESTION 20

Line 10:

- ○ `p = q.right.next;`
- ○ `p = p.next;`
- ○ `n = q.right;`
- ○ `n = n;`
- ○ None

```
public static void main(String args[]){
        LinkedList<String> l= new LinkedList<String>();

        l.insert("A"); l.insert("B");
        l.findFirst();
        String str1= l.retrieve();
        l.insert(str1);
        l.findFirst();
        l.insert("C"); l.insert("D");
        l.findFirst();
        l.insert("E");
        l.findNext();
        l.remove();
        l.findFirst();
        while(!l.last()){
                String str2= l.retrieve();
                l.update(str1);
                str1= str2;
                l.findNext();
        }
        l.findFirst();
        l.insert(l.retrieve());
        l.findFirst();
        l.update("C");
    }
}
```

```
                    }
                    }
                    }
                    }
                    }
                    }
                    }
                    }
}
```

- ○ C A A E B D
- ○ A A E D B C
- ○ A E D B A C
- ○ C A A B D E
- None

Click Save and Submit to save and subm