

Name: \_\_\_\_\_

UserID: \_\_\_\_\_

Lab Section: \_\_\_\_\_

Date: \_\_\_\_\_

## Section 1. Collections

1. A chain of linked nodes is a good example of which of the following terms?
  - (a) abstract data type
  - (b) collection
  - (c) data structure
  - (d) generic type
2. Let `c` be an instance of some collection that we have implemented using an array as the storage data structure. Assume that this collection uses the *dynamic resizing* strategy that we discussed that allowed us to guarantee that the array uses  $O(N)$  memory at all times, where  $N$  is the number of elements in the collection. What will the capacity of the array be after the following sequence of calls to the `add` and `remove` methods?  

```
c.add(1); c.add(2); c.add(3); c.add(4); c.add(5);  
c.remove(1); c.remove(2); c.remove(3);
```

  - (a) 10
  - (b) 8
  - (c) 4
  - (d) 2
3. Suppose we implemented a Set collection using a chain of linked nodes as the storage data structure. Suppose also that we tailored the implementation of the set to optimize the performance of the *difference* method, even at the expense of other methods of the set. Assume that in the call `s1.difference(s2)`, `s1` has size  $M$  and `s2` has size  $N$ . (Recall the definition of set difference: `s1.difference(s2)` would return a set that contains the elements in `s1` that are *not* in `s2`.) Under those assumptions and using only concepts already covered in the course, what is the best big-oh time complexity that the difference method could achieve?
  - (a)  $O(M \log N)$
  - (b)  $O(M \times N)$
  - (c)  $O(N \log N)$
  - (d)  $O(M + N)$
4. Suppose we implemented a Set collection using an array as the storage data structure. Suppose also that we tailored the implementation of the set to optimize the performance of the *difference* method, even at the expense of other methods of the set. Assume that in the call `s1.difference(s2)`, `s1` has size  $M$  and `s2` has size  $N$ . (Recall the definition of set difference: `s1.difference(s2)` would return a set that contains the elements in `s1` that are *not* in `s2`.) Under those assumptions and using only concepts already covered in the course, what is the best big-oh time complexity that the difference method could achieve?
  - (a)  $O(M \log N)$
  - (b)  $O(M \times N)$
  - (c)  $O(N \log N)$
  - (d)  $O(M + N)$

5. Select the term below that best matches the following definition: “A particular way of storing and organizing data in a computer so it can be used efficiently.”
- (a) util class
  - (b) collection
  - (c) data structure
  - (d) generic type
6. An array is a good example of which of the following terms?
- (a) abstract data type
  - (b) collection
  - (c) data structure
  - (d) generic type
7. Select the term below that best matches the following definition: “General ways of organizing data and providing controlled access to that data.”
- (a) util class
  - (b) collection
  - (c) data structure
  - (d) generic type

## Section 2. Linked Structures

Answer questions 8 – 10 in the context of the code below.

```
private class Node {  
    private Object element;  
    private Node next;  
  
    public Node(Object e) {  
        element = e;  
    }  
  
    public Node(Object e, Node n) {  
        element = e;  
        next = n;  
    }  
}
```

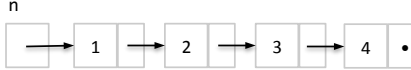

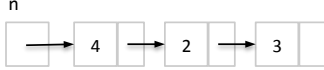
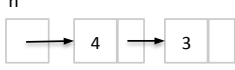
8. How many Node objects will be garbage after the following code executes?

```
Node n = new Node(1);
n.next = new Node(2);
Node m = new Node(3, n);
m.next = new Node(4, n);
Node p = m.next;
n = n.next;
p.next = n;
n = null;
p = null;
```

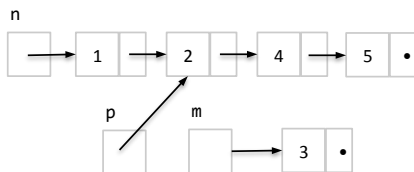
- (a) 0
- (b) 1
- (c) 2
- (d) 3

9. Which chain of nodes is created by the following code?

```
Node n = new Node(1);
n.next = new Node(2, new Node(3));
n = new Node(4, n.next.next);
```

- (a) 
- (b) 
- (c) 
- (d) 

10. Suppose n, m, and p are references to Node objects as shown in the image below. Which set of statements would modify the pointer chain so that the nodes are in the order 1, 2, 3, 4, 5?



- (a) `m = p.next; p = m;`
- (b) `m.next = p; p.next = m;`
- (c) `m.next = p.next; p.next = m;`
- (d) `p.next = m; m.next = p.next;`

11. Which of the following best defines memory that would be identified as *garbage*? (In this context, the term *stack* refers to the portion of memory referred to as “stack memory” and the term *heap* refers to the portion of memory referred to as “heap memory”)
- (a) memory that has been allocated on the stack but is inaccessible from the heap
  - (b) memory that has been allocated on the heap but is inaccessible from the stack
  - (c) memory that has been allocated on the stack but is never referenced by the program
  - (d) memory that has been allocated on the heap but is never referenced by the program
12. Which of the following is a primary disadvantage of using linked nodes?
- (a) no random access to arbitrary nodes
  - (b) a limit on the total number of nodes that can be linked together
  - (c) an inability to store more than one data item per node
  - (d) a given node can only link to one other node

### Section 3. Stacks and Queues

13. What does the stack *s* contain after the following sequence of operations? Note that the top of stack *s* is the left-most element listed.

```
s.push(1); s.pop(); s.push(2); s.pop();
s.push(3); s.push(4); s.push(s.pop());
s.push(5);
```

- (a) *top* | 4, 3, 5
  - (b) *top* | 5, 3, 4
  - (c) *top* | 3, 4, 5
  - (d) *top* | 5, 4, 3
14. Suppose we were to implement a queue using an array such that both *enqueue* and *dequeue* have  $O(1)$  worst-case time complexity. (Recall that to ensure constant time operations, we had to treat the array as *circular*. Assume that the *front* and *rear* markers begin at index 0.) Suppose also that the queue has a fixed capacity of 5. That is, there is no array resizing done. Which choice below depicts the contents of the array after the following sequence of operations? (The array is shown from left to right beginning at index 0. The symbol  $\bullet$  is used to denote an empty cell.)

```
enqueue(1); enqueue(2); enqueue(3);
dequeue();
enqueue(4); enqueue(5);
dequeue();
dequeue();
enqueue(6); enqueue(7);
```

- (a) [4, 5, 6, 7,  $\bullet$ ]
- (b) [ $\bullet$ , 4, 5, 6, 7]
- (c) [5, 6, 7,  $\bullet$ , 4]
- (d) [6, 7,  $\bullet$ , 4, 5]

15. What does the queue `q` contain after the following sequence of operations? Note that the front of queue `q` is the left-most element listed and the rear of queue `q` is the right-most element listed.

```
q.enqueue(1); q.dequeue(); q.enqueue(2); q.dequeue();
q.enqueue(3); q.enqueue(4); q.enqueue(q.dequeue());
q.enqueue(5);
```

- (a) `front | 4, 3, 5 | rear`
  - (b) `front | 5, 3, 4 | rear`
  - (c) `front | 3, 4, 5 | rear`
  - (d) `front | 5, 4, 3 | rear`
16. Recall the stack-based postfix evaluation algorithm from class. If it were applied to the following postfix expression, what would the stack contain after the 9 has been read but *before* the multiplication operator has been read? As before, the top of the stack is the left-most element listed.

Postfix: 7 8 2 5 4 - + + 9 \* 3 + +

- (a) `top | 9, 11, 7`
  - (b) `top | 9, 9, 7`
  - (c) `top | 7, 8, 2, 9`
  - (d) `top | 7, 8, 9`
17. Suppose we have a queue and we add four Strings as follows.

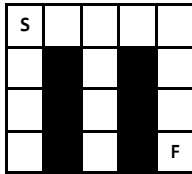
```
q.enqueue("Abe"); q.enqueue("David"); q.enqueue("Sam"); q.enqueue("Eli");
What is left on the queue after the following call to the method below?
josephus(q, 4);
```

```
public void josephus(Queue q, int N) {
    while (q.size() > 1) {
        for (int i = 0; i < N; i++) {
            q.enqueue(q.dequeue());
        }
        q.dequeue();
    }
}
```

- (a) Abe
  - (b) David
  - (c) Sam
  - (d) Eli
18. Suppose we were to implement a stack using an array named `elements` as the storage data structure. Assume that we have ensured that `push`, `pop`, `peek`, and `size` all have  $O(1)$  worst-case time complexity. Which of the following statements would appear in the `peek` method?

- (a) `return elements[size() - 1];`
- (b) `return elements[elements.length - 1];`
- (c) `return elements[0];`
- (d) `return top;`

19. Recall the stack-based depth-first search algorithm that we developed to solve a maze. Suppose we applied that algorithm to the maze below where 'S' marks the starting position and 'F' marks the ending (goal) position. Also assume that the order in which adjacent positions are considered is down, left, up, right. When the algorithm finds the goal ('F'), how many positions will be on stack?



- (a) 0
- (b) 8
- (c) 14
- (d) 16

#### Section 4. Recursion

20. Which statement should replace line 5 to correctly complete the tail-recursive factorial method below?

```

1 public int fact_tr(int n, int f) {
2     if (n == 1)
3         return f;
4     else
5         _____;
6 }
```

- (a) `return fact_tr(n - 1) * n;`
- (b) `return fact_tr(n * f, n - 1);`
- (c) `return fact_tr(n - 1, n + f);`
- (d) `return fact_tr(n - 1, n * f);`

21. What is the return value of the following call to the method below?

```

foo(9, "");

public String foo(int n, String s) {
    if (n < 2) {
        return n + s;
    }
    return foo(n / 2, (n % 2) + s);
}
```

- (a) 1001
- (b) 0110
- (c) 1111
- (d) 1010

22. Which statement should replace line 5 to correctly complete the recursive length method below?

```
1 public int length(Node n) {  
2     if (n == null)  
3         return 0;  
4     else  
5         _____;  
6 }
```

- (a) `return length(n.next);`
- (b) `return 1 + length(n - 1);`
- (c) `return length(n.next) + 1;`
- (d) `return 1 + length(n);`

23. What are the contents of the array `a = [1, 2, 3, 4, 5, 6]` after the following call to the method below?

`baz(a, 0, 5);`

```
public void baz(int[] a, int left, int right) {  
    if (left < right) {  
        int temp = a[left];  
        a[left] = a[right];  
        a[right] = temp;  
        baz(a, ++left, --right);  
    }  
}
```

- (a) `[2, 1, 4, 3, 6, 5]`
- (b) `[6, 1, 4, 3, 5, 2]`
- (c) `[1, 2, 3, 4, 5, 6]`
- (d) `[6, 5, 4, 3, 2, 1]`

24. What is the return value of the following call to the method below?

`bar(4, 5);`

```
public int bar(int a, int b) {  
    if (b == 0)  
        return 0;  
    else if (b % 2 == 0)  
        return bar(a + a, b / 2);  
    else  
        return bar(a + a, b / 2) + a;  
}
```

- (a) `0`
- (b) `4`
- (c) `16`
- (d) `20`

# Answer Key for Exam | | |---| | A | |---|

## Section 1. Collections

- |        |        |        |
|--------|--------|--------|
| 1. (c) | 4. (a) | 7. (b) |
| 2. (b) | 5. (c) |        |
| 3. (b) | 6. (c) |        |

## Section 2. Linked Structures

- |        |         |         |
|--------|---------|---------|
| 8. (b) | 10. (c) | 12. (a) |
| 9. (d) | 11. (b) |         |

## Section 3. Stacks and Queues

- |         |         |         |
|---------|---------|---------|
| 13. (d) | 16. (a) | 19. (b) |
| 14. (d) | 17. (b) |         |
| 15. (a) | 18. (a) |         |

## Section 4. Recursion

- |         |         |         |
|---------|---------|---------|
| 20. (d) | 22. (c) | 24. (d) |
| 21. (a) | 23. (d) |         |