Miami University EXAM 01 2006-02-23

Miami University - CSA274 - Spring 2006 - Exam 1
There are 31 questions for a total of 160 points.

Name: ______________________________

Instructions

Please read through this entire exam very carefully before starting.

This exam is closed notes and closed books.

All work must be written on the exam pages in order to be graded. Any scrap paper used, must be the scrap paper provided during the exam period.

For programming questions: Please be as accurate as possible with your Java syntax: this includes appropriate use of braces, semicolons, and the proper use of upper/lowercase letters.

No electronic devices may be used during the exam: this includes calculators, iPods, PDAs, and cellular phones.

You have 75 minutes (1 hour and 15 minutes) to complete the exam.

There are 160 possible points, but the exam will be graded out of 150 points.

Good Luck!

Definitions / Fill in the blank

1. (2 points) What does ADT stand for? ____________________

2. (2 points) ____________________ abstraction is the separation of what is to be achieved, from how it is achieved

3. (2 points) In Java: ____________________ define only methods signatures and constants.

4. (2 points) In Java: ____________________ define method signatures, methods (optionally) and data members, but cannot be instantiated.

5. (2 points) Exceptions in Java come in two flavors: ____________________ and ____________________ (in regards to differentiating factor of which must be declared that they are thrown.)

6. (2 points) When writing a class, ____________________ is the keyword for explicitly calling a constructor on the class that your class extends.

7. (2 points) When writing a class, ____________________ is the keyword for explicitly calling another constructor on your class.

8. (2 points) Defining two methods with the same name, but different parameters is called method ____________________.

9. (2 points) When you define a method with the same name and same parameters as a method defined in your parent class, this is called method ____________________.

10. (2 points) Finding the correct method to execute at runtime is called: ____________________.

11. (2 points) If a type parameter is not specified when instantiating a generic collection, what type is that collection then given? ____________________

12. (2 points) When accessing every item in a linked list, if it better to use a for loop and the get method, or the collection's iterator? ____________________

13. (2 points) When class A extends class B, we would say that class A ____________________ class B.

14. (2 points) When class C has a private data member of type class D, we would say that class C ____________________ class D.

15. (2 points) The process of converting a primitive `int` to the class type `Integer` is called: ____________________

16. Provide the name of each of these running time growth rate classifications:
    (a) (2 points) $O(n^3)$ ______________________
    (b) (2 points) $O(1)$ ______________________
    (c) (2 points) $O(n!)$ ______________________
    (d) (2 points) $O(\log n)$ ______________________
    (e) (2 points) $O(n^2)$ ______________________
    (f) (2 points) $O(n)$ ______________________
    (g) (2 points) $O(2^n)$ ______________________
    (h) (2 points) $O(n * \log n)$ ______________________

17. Put the above growth rate classifications in order, starting with the slowest growth rate:
    (a) (2 points) ______________________
    (b) (2 points) ______________________
    (c) (2 points) ______________________
    (d) (2 points) ______________________
    (e) (2 points) ______________________
    (f) (2 points) ______________________
    (g) (2 points) ______________________
    (h) (2 points) ______________________

Multiple Choice

Please circle the best answer(s), you may select more than one answer for any given question. The code listing below applies for the multiple choice questions. In this situation each player on the team is mentored by one of the captains, and the team has multiple captains.

```
public class Player extends java.lang.Object {
   private String name;
   private int number;
   private Captain mentor;

   // assume get and set methods for each of the properties defined
}

private class Captain extends Player {
   private List<Player> players = new ArrayList<Player>();

   public List<Player> getMentoredPlayers() { return players; }
   public void addMentoringResponsibility( Player e ) { players.add( e ); }
   public boolean removeMentoringRespondibility( Player e ) {
      return players.remove( e );
   }
}
```

18. (2 points) A variable of type `Object` can be assigned instances of type(s):
    (a) `Player`
    (b) `Captain`
    (c) neither

19. (2 points) A variable of type `Player` can be assigned instances of type(s):
    (a) `Player`
    (b) `Captain`
    (c) neither

20. (2 points) A variable of type `Captain` can be assigned instances of type(s):
    (a) `Player`
    (b) `Captain`
    (c) neither

Short Answer

Provide a short answer to the following questions. Please use complete sentences when answering.

21. (4 points) What is the difference between `size` and `capacity` for a list structure that is backed by an array? What happens when an `add( E elem )` operation is invoked and size is equal to capacity?

22. For `ArrayList`:
    (a) (3 points) Why do we by default insert at the end? What is the running time?

    (b) (3 points) When we insert in the middle of the list, how is room made? What is the running time?

23. (6 points) For linked lists, why is it more efficient to use an iterator instead of `get(int)` to traverse a list? What is the running time of each of these methods to traverse all of the elements in the list?

24. (4 points) What is the acronym to describe the order in which items are added to and removed from a stack? What does it mean?

25. (4 points) What is the acronym to describe the order in which items are added to and removed from a queue? What does it mean?

26. (6 points) Use a stack to show how the following postfix expression would be evaluated. Please show the stack at each stage of the evaluation.
`1 2 + 3 4 - 5 6 + * +`

Short Programming & Evaluation

27. (10 points) Write a new queue method called `moveToRear` that moves the element currently at the front of the queue to the rear of the queue. Do this using the methods `Queue.offer` and `Queue.remove`. You may assume that the Queue is defined in a generic fashion and that you have a type defined of name `E`. This method should to nothing if the queue is empty, and should not throw an exception in the event that it is.

```
public class Queue<E> {

  /**
   * Insert a new element in the queue
   */
  public void offer( E element ) { /* ... */ }

  /**
   * remove the first element from the queue
   * @throws NoSuchElementException if the queue is empty
   */
  public E remove() throws NoSuchElementException { /* ... * / }

  public void moveToRear() {
      // BEGIN ANSWER




      // END ANSWER
  }
}
```

28. Answer the following questions using the code listing provided below.

```
private static long calculate( long i, long j ) {
    if ( j == 1 ) {
        return i;
    } else {
        return i * calculate( i, j - 1 );
    }
}

public static long calculate( long i ) {
    return calculate( i, i );
}
```

(a) (2 points) What is the result when `calculate( 3 )` is called?

(b) (2 points) What is the result when `calculate( 5 )` is called?

(c) (6 points) Textually, what does this code calculate?

29. (10 points) Complete the following section of code. You are to write the appropriate assignment statements to assure that the new node is added to the list **before** the node that `curNode` points to, and that no part of the list is lost in the operation. The start of the function takes care of advancing the `curNode` local variable to the correct position in the code.

```
public class LinkedList<E> implements java.util.List<E> {

    private Node<E> head;
    private Node<E> tail;

    private static class Node<E> {
        private E data;
        private Node<E> prev;
        private Node<E> head;
    }

    private void add( int i, E element ) {
        checkIndex( i );

        Node<E> curNode = head;
        for( int idx = 0; idx < i; idx++ ) {
            curNode = curNode.next;
        }

        Node<E> newNode = new Node<E>( element );

        // write the code to insert in the middle of the linked list here
        // use only curNode and newNode to complete this task

        // BEGIN ANSWER










        // END ANSWER
   }
}
```

30. (16 points) Write a recursive method to determine which coins are given as change. For example, if 65 is the input, the string `"quarter quarter dime nickel "` should be returned. The function should work for *any* amount passed in (i.e. not just amounts less than 100), but should only return the change in the coins: quarter (25), dime (10), nickel (5), and penny (1).

```
public static String calculateChange( int amount ) {
    // BEGIN ANSWER









    // END ANSWER
}
```

31. (16 points) Write a recursive method to perform binary search on an array. The incoming array will already be sorted and the method should return -1 in the event that the target is not found in the array.

```
public int binarySearch( int[] array, int target ) {
  return binarySearch( array, target, 0, array.length - 1 );
}

private int binarySearch( int[] array, int target, int first, int last ) {









}
```