**01 - Review**

**01) Direct addressing is to indirect addressing as (indicate the best match):**

*Primitive variable is to reference variable.*

**02) This structure is the best model for describing the chapters and sections and subsections of a textbook.**

*Tree*

**03) A protected variable:**

*Is visible inside the class within which it is defined, or within it's package, or within classes derived from it's classes.*

**04) In Java, method arguements are passed by:**

*Value*

**05) This Java statement is used to "anounce" that an exception has occured.**

*Throw*

**06) Subclasses are assignment compatible with the superclasses above them in the inheritance hierarchy.**

*True*

**07) A tree child element can only have a single parent.**

*True*

**08) A private variable cannot be accessed from anywhere.**

*False*

**09) A queue is a first in, first out structure.**

*True*

**10) A class definition can be used to create multiple objects.**

*True*

**11) Structure is best model for a group of customers waiting to check out in a grocery store.**

*Queue*

**12) Java programmers define their own exceptions by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ the library’s Exception class.**

*Extending*

**13) The following code results in garbage being created.**

Circle c1 = new Circle(8);

Circle c2;

c2 = c1;

*False*

**14) Java supports single inheritance only, i.e., a class can only directly inherit from a single superclass.**

*True*

**02 - Quiz**

**01) In the text we developed Java code for traversing a linked list. Here is a possibly flawed approach for**

**using a traversal to print the contents of the linked list of strings accessed through letters. Indicate the**

**description that best matches the code.**

LLNode>String< currNode = letters;

while (currNode != null)

{

currNode = currNode.getLink();

System.out.println(currNode.getInfo());

}

*Will bomb is list is not empty;*

**02) We deal with ADTs on three levels. On which level do we just deal with the “what” questions, as in**

**what does the ADT model do, what are its responsibilities, what is its interface?**

*Abstract / Logical*

**03) Suppose we have a linked list of Strings, as defined in the text, named presidents. Suppose it**

**contains three nodes, with the first node holding “Washington”, the second node “Adams”, and the third**

**node “Jefferson”. What would be the result of the following code?**

*'Washington' is printed*

**04) We deal with ADTs on three levels. On which level do we deal with the “how” questions, as in how do**

**we represent the attributes and fulfill the responsibilities of the ADT?**

*Implementation (or concrete / internal)*

**05) Stacks are useful in situations where:**

*?? ;*

**06) If N represents the number of elements on the stack, then the isFull method of the**

**ArrayBoundedStack class is O(N).**

*False*

**07) UML diagrams are a form of abstraction, since they hide details and allow us to concentrate just on**

**the major design components.**

*True*

**08) The StackOverflowException is an unchecked exception.**

*True*

**09) If N represents the maximum possible number of elements on the stack, then the LinkedStack**

**constructor is O(N).**

*False*

**10) The following expression is balanced: ( ( xx [ {{ \*\* }} ] XX ) )**

*True*

**11) Which of the following statements is a legal declaration/instantiation of a Stack object based on the text**

**code?**

*StackInterface stack = new ArrayBoundedStack (3);*

**12) Starting with an empty stack, the sequence of operations push(1), push(2), pop(), push(3), leaves the**

**stack, from top to bottom, as:**

*3, 1*

**13) Starting with an empty stack the sequence of operations push(1), push(2), push(3), pop(), leaves the**

**stack, from top to bottom, as:**

*2, 1*

**14) Generics have been part of the Java language since it was first created.**

*False*

**03 - Review**

**01) Assuming values is a full array of int, what does the following recursive method return (assume it is**

**passed a legal argument, a value between 0 and values.length)?**

int mystery(int n)

{

if (n == values.length)

return 0;

else

return (n + mystery(n + 1));

}

*none of these is correct*

**02) Given that values is of type LLNode and references a linked list (non-empty) of Integer**

**objects, what does the following code do if invoked as mystery(values)?**

void mystery(LLNode list)

{

if (list != null)

{

System.out.println(list.getInfo());

mystery(list.getLink());

}

}

*Prints the list from start to end*

**03) Recursion is implemented by the system:**

*using run time storage allocation.*

**04) The number of recursive calls that a method goes through before returning is called:**

*the depth of recursion*

**05) Given that values is of type LLNode and references a linked list (possibly empty) of**

**Integer objects, what does the following code do if invoked as mystery(values)?**

int mystery(LLNode list)

{

if (list == null)

return 0;

else

return mystery(list.getLink());

}

*return 0*

**06) True or False? Recursive methods often have fewer local variables than the equivalent nonrecursive**

**methods.**

*True*

**07) The base case does not exist or is not reached when there is infinite recursion.**

*True*

**08) True or False? Recursive algorithms are usually implemented with while loops.**

*False*

**09) True or False? The binary search algorithm is O(N) where N is the size of the sorted array.**

*False*

**10) True or False? The static storage allocation approach creates space for a method when the method**

**is invoked.**

*False*

**11) The following code is supposed to return n!, for positive n. An analysis of the code using our "Three**

**Question" approach reveals that:**

int factorial(int n){

if (n == 0)

return 1;

else

return (n + factorial(n - 1));

}

**12) What does the following return, when passed the argument 1564?**

int recur(int n)

{

if (n < 0) return -1;

else

if (n < 10) return n;

else

return ((n % 10) + recur(n / 10));

}

*16*

**13) True or False? Recursive methods must always contain a path that does not contain a recursive call.**

*True*

**14) True or False? Recursive methods are always shorter and clearer than the equivalent nonrecursive**

**methods.**

*False*

**04 - Review**

**01) Suppose you start with an empty queue and perform the following operations: enqueue 1, dequeue,**

**enqueue 2, dequeue, enqueue 3, enqueue 4, dequeue, dequeue, enqueue 5. What are the resultant**

**contents of the queue, from front to back?**

*None of these is correct, it is 5*

**02) Recall that within the LinkedQueue the front and rear variables are of type LLNode holding**

**references to the front and rear nodes of the underlying linked list, and the numElements variable is an**

**int and holds the current size of the queue. Which of the following code sequences could be used to**

**correctly implement the isEmpty operation?**

*return (rear == null);*

*return (front == null);*

*return (numElements == 0);*

*All of these are correct.*

**03) Suppose you start with an empty queue and perform the following operations: enqueue 1, enqueue 2,**

**dequeue, enqueue 3, enqueue 4, dequeue, enqueue 5. What are the resultant contents of the queue,**

**from front to back?**

*3,4,5*

**04) The following sequence of operations essentially leaves a queue unchanged.**

*enqueue followed by dequeue*

**05) How did the text decide to handle the possibility of queue underflow?**

*Throw a QueueUnderflowException if it occurs, and provide an isEmpty operation so a client can prevent underflow.*

**06) True or False? We use a call to the join method to indicate that the main thread should wait until**

**another thread has completed executing before continuing.**

*True*

**07) True or False? The isEmpty operation as defined for the text's Queue ADT might throw the**

**QueueUnderflowException.**

*False*

**08) True or False? A java interface can inherit from at most one other interface.**

*False*

**09) True or False? A "dequeue" allows an application to peek at the front of rear values of a queue.**

*False*

**10) True or False? A queue is a first in, first out structure.**

*True*

**11) True or False? If N represents the number of elements in the queue, then the size method of the**

**ArrayBoundedQueue class is O(N).**

*False*

**13) True or False? If you enqueue 5 elements into an empty queue, and then perform the isEmpty operation 5**

**times, the queue will be empty again.**

*False*

**14) True or False? A standard linked list provides a good implementation of a "Deque".**

*False*

**15) True or False? When implementing a queue with a linked list, the front of the queue is also the front**

**of the linked list.**

*True*

**05 - Review**

**01) The equals method of the Object class returns true only if the two objects being compared:**

*have identical attributes*

*are the same object*

*are aliases of each other*

*are == to each other*

***\*\* All of these can be correct \*\****

**02) True or False? Even though our collections will be generic, our CollectionInterface is not generic.**

*False*

**03) True or False? The add method of our Collection ADT might throw the CollectionOverflowException.**

*False*

**04) True or False? If N represents the number of elements in the collection, then the contains method of the ArrayCollection class is O(1).**

*False*

**05) True or False? If N represents the number of elements in the collection, then the contains method of the ArrayCollection class is O(N).**

*True*

**07 - Review**

**01) True or False? A depth-first traversal of a tree is also known as a level-order traversal.**

*False*

**02) True or False? When a binary search tree is balanced, it provides O(log2N) search, addition, and removal.**

*True*

**03) True or False? The number of nodes in a non-empty tree is equal to the number of nodes in its left subtree plus the number of nodes in its right subtree plus 1.**

*True*

**04) True or False? When a binary search tree is balanced, it provides O(N2) search, addition, and removal**

*False*

**05) Recall that within our BinarySearchTree class the root variable is of type BSTNode<T>. Which of**

**the following code sequences could be used as the body of the isEmpty method?**

*return root == null*

**06) For our binary search trees, the public method, size, calls the private recursive method, recSize,**

**and passes it a reference to the root of the tree, returning to the caller whatever is returned to it by**

**recSize. Critique the following implementation of recSize, assuming its parameter variable is node:**

if (node == null)

return 0;

else

return 1 + recSize(node.getLeft()) + recSize(node.getRight());

*the code works correctly*

**07) Show the post-order traversal of the tree that results from starting with an empty tree and adding 10,**

**20, 25, 5, 15, 17, 16, 14 and then removing 20.**

*5, 14, 16, 17, 15, 25, 10*

**08) In the above tree, the descendants of node A are:**

*M, D, B, H, G, T*

**09) Suppose xOrder is one of our binary search tree traversal methods that is passed a tree node node and a queue q. Which traversal order is provided by the following code?**

xOrder(node.getLeft(), q);

xOrder(node.getRight(), q);

q.enqueue(node.getInfo());

*post order*

**10) Show the post-order traversal of the tree that results from starting with an empty tree and adding 10, 15, 18, 8, 9, 5 and then removing 9**

*5, 8, 18, 15, 10*

**08 - Review**

**01) Our map iterators provide an iteration through**

*- keys*

*- values*

*- pairs*

*- All are correct*

*\*\* C -- key, value pairs*

**02) Suppose the map M holds pairs and is currently empty. What is output by**

**the following statement?**

System.out.println(M.put('C', "cow"));

*null*

**03) Our HMap class handles iteration by:**

*using an anonymous inner class (like Java's HashMap)*

*using a named inner class*

**04) The collision resolution approach called "linear probing" sometimes suffers from a problem called:**

*clustering (primary clustering)*

**05) Which of these statements describes a Map?**

*A structure which contains linked keys and values, in which the user can provide a key to get the stored value*

**06) True or False? In a map, a key can be null.**

*False*

**07) True or False? Using a balanced binary search tree to implement a map would allow an O(log2N) get operation (where N is the number of elements in the map)**

*True*

**08) True or False? All Java objects support the hashCode method**

*True*

**09) True or False? Our ArrayListMap is unbounded**

*True*

**10) True or False? Our HMap class supports the remove operation**

*False*

**11) A Hash Collision is when:**

*Two different objects produce the same hash value*

**12) True or False? When a hash function is used to determine the placement of elements in an array, the order in which the elements are added does not affect the resulting array**

*False*

**13) True or False? Linear probing is used as a collision resolution approach for hash based systems.**

*False*

**14) True or False? The values stored in a map must be unique.**

*False*

**15) A Hash function:**

*Turns a value into a deterministic set of numbers so that the same value would return the same hash result*

**09 - Review**

**01) True or False? A Priority Queue supports FIFO access**

*False*

**02) True or False? Suppose a (max) heap consists of the elements 1, 5, 7, 9, 10. Then the root of the heap "tree" is 10.**

*True*

**03) True or False? Suppose a heap consists of the elements 1, 5, 7, 9, 10. Then the underlying tree is full.**

*False*

**04) True or False? All heaps with 10 elements have the same "shape."**

*True*

**05) True or False? You can implement a priority queue using a linked list.**

*True*

**06) True or False? A precondition of the reheapUp operation is that the last index position of the tree is empty.**

*True*

**07) True or False? Using a sorted linked list (sorted in reverse order) to implement a priority queue would require an O(N) dequeue operation (where N is the number of elements in the priority queue).**

*False*

**08) True or False? A precondition of the reheapUp operation is that the root of the tree is empty.**

*False*

**09) True or False? Using a sorted linked list to implement a priority queue would require an O(N) enqueue operation (where N is the number of elements in the priority queue).**

*True*

**10) True or False? Suppose a heap consists of the elements 1, 5, 7, 9, 10, 15, 19. Then the underlying tree is full.**

*True*

**11) True or False? The underlying tree of a heap is a binary search tree**

*False*

**12) True or False? A precondition of the reheapDown operation is that the last index position of the tree is empty.**

*False*

**13) True or False? Using a sorted linked list to implement a priority queue would allow an O(1) enqueue operation.**

*False*

**10 - Review**

**01) The getToVertices(T vertex) method from the Graph Interface in the book:**

*Returns an Iterator*

**02) True or False? A connected graph is complete**

*False*

**03) True or False? The adjacency matrix approach to graph implementation is very efficient in terms of time**

*True*

**04) True or False? A graph can have more edges than vertices**

*True*

**05) True or False? A graph with just one vertex is connected.**

*True*

**06) True or False? A connected graph with N vertices must have at least N-1 edges.**

*True*

**07) True or False? All trees are graphs**

*True*

**08) True or False? A graph with five vertices and four edges is connected**

*False*

**09) True or False? A complete graph is connected**

*True*

**10) True or False? A directed graph must be weighted**

*False*

**11) Undirected graph V consists of vertices A, B, C, D, E, F and edges (A,B), (A,C), (A,D), (B,C), (B,D) and (C,D). Graph V is:**

*Disconnected, Complete*

**True or False? A connected graph is complete**

*False*

**True or False? A graph can have more edges than vertices**

*True*

**11 - Review**

**01) True or False? The efficiency of the Insertion Sort is O(N2) where N is the size of the list being sorted.**

*True*

**02) True or False? The efficiency of the Merge Sort is O(N2), where N is the size of the list being sorted.**

*False*

**03) True or False? After one pass of the Bubble Sort, the smallest element is in its correct place in the array.**

*True*

**04) True or False? For a "randomly arranged" array, the efficiency of the Quick Sort is O(Nlog2N) where N is the size of the list being sorted.**

*True*

**05) True or False? The efficiency of the Heap Sort is O(N2), where N is the size of the list being sorted.**

*False*

**06) True or False? The efficiency of the Quick Sort is always O(Nlog2N), where N is the size of the list being sorted.**

*False*

**07) True or False? The efficiency of the Heap Sort is O(Nlog2N), where N is the size of the list being sorted.**

*True*

**08) True or False? The efficiency of the Selection Sort is O(N2), where N is the size of the list being sorted.**

*True*

**09) True or False? In the worst case, Quick Sort uses O(log2N) extra space, where N is the size of the list being sorted.**

*False*

**10) True or False? After one pass of the Selection Sort the largest element is in its correct place in the array.**

*False*

**11) True or False? The efficiency of the Quick Sort is O(N2) in the worst case, where N is the size of the list being sorted.**

*False*

**12) rue or False? After one pass of the Selection Sort, the smallest element is in its correct place in the array.**

*True*

**13) True or False? In the best case, Quick Sort uses O(log2N) extra space, where N is the size of the list being sorted.**

*True*

**14) True or False? In the worst case, Quick Sort uses O(N) extra space, where N is the size of the list being sorted.**

*True*

**15) True or False? You can perform a binary search on an unsorted array.**

*False*