CST8130: Data Structures
Midterm Test #2 : Version B

Name: _____

Instructions

Answer all questions on the test paper. There are 19 questions worth a total of 44 marks.
If you have any questions, raise your hand and I will come and answer.

Part 1 – Multiple Choice (12 marks) Circle the best answer on the paper.

The following class – which is the same one from Midterm #1 - will be used for a number of questions:

```
public class Student {

    private  int studentNumber;
    private String studentName;
    private float studentGPA;

    public Student (int studentNumber, String studentName, float studentGPA) {
            this.studentNumber = studentNumber;
            this.studentName = studentName;
            this.studentGPA = studentGPA;
    }

….other methods……..

    public String toString( )  {
          return studentName + " " + studentNumber + " has average " + studentGPA;
    }

    public boolean  isEligible() {
           if (studentGPA >= 2.7f)
                  return true;
           else return false;

   }
}




public class Course {
   private ArrayList<Student> students;
   private String courseName;

  …. methods……..

}
```

1. The code for method **toString** in the **Course** class to display all the data in the **Course** data members would be:

a) **return courseName +" " + students.toString();**

b) **return courseName + students;**

c) **String out = courseName + " ";**
for (int i=0; i<students.size(); i++)
out += students.get(i).toString() + "\n";
return out;

d) **a and c**

e) None of the above

2. What would be the most *correct* code for the default constructor of **Course** class?

a)
```
public Course () {
        super();
        courseName = new Student();
        students = new ArrayList();
}
```

b)
```
public Course () {
        courseName = new String();
        students = new ArrayList<Student>();
}
```

c)
```
public Course () {
        super();
        courseName = new Student();
        students = new ArrayList<Student>();
}
```

d)
```
public Course () {
        courseName = new String();
        students = new ArrayList();
}
```

e)
```
public Course () {
}
```

3. If an **Integer** object contains 1 Byte of memory (to hold the primitive int inside it), and a reference uses 1 Byte of memory, how many Bytes of memory does a singly linked list with 4 elements of type **Integer** have in memory do not include any memory you need for head/tail?

a) 8 **b) 12** c) 13 d) 18 e) 21 f) 36

4. Which of the following is the worst algorithm measurement?

a) $O(n \log_2 n)$ b) $O(2)$ c) $O(n^2 \log_2 n)$ d) $O(\log_2 n^2)$ **e) $O(n^3)$**

5. Given the following numbers added in this order to a binary tree 6, 10, 8, 2, 3, 1
What is the pre-order traversal of this tree?

a) 6 2 1 3 10 8
b) 2 1 3 10 8 6
c) 6 2 1 3 10 8
d) 1 2 3 6 8 10
e) 1 3 2 8 10 6

6. Consider **LList** and **LLNode** classes (single-direction with only head). The code needed to delete the object at the head of the list is:

```
a)if (head != null) {
        LLNode toDelete = head;
        head = head.getNext();
        }
```

```
b)  if (head.getNext() != null) {
        LLNode toDelete = head.getNext();
        head = toDelete.getNext();
        }
```

```
c)if (head != null ) {
        LLNode toDelete = head;
        head.updateNext(toDelete);
        }
```

```
d)  if (head != null) {
        LLNode toDelete = head.getNext();
        head = head.getNext().getNext();
        }
```

e) none of the above.

7. What is the Big-O for the following algorithm? Assume that `DoIt(…)` has an efficiency of O(*n*).

```
 j = n;
 while (j > 0) {
    DoIt(…);
    j = j / 2;
    }
 }
i = 1;
while (i < n){
    i = i * 2;
    DoIt(…);
 }
```

What is the Big-O for the above algorithm? Assume that `DoIt(…)` has an efficiency of O(*n*).

a) **O($n \log_2 n$)** b) O(n^3) c) O($n^2 \log_2 n$) d) O($\log_2 n^2$) e) O(n^2)

8. An array contains these elements (in ascending order):

12	14	17	24	29	33	41	47	58	62	68	79	83	92

In what order would a binary search look to find the number 43 in the array (and 43 is not there) where midpoint is computed as (lowerIndex + upperIndex)/2 in each iteration?

a) 47 24 33 41
b) 47 24 14 17
c) 47 41
d) 41 47
e) 41 68 58 47

9. Given the following numbers added in this order to a binary tree 6, 10, 8, 2, 3, 1
   What is the post-order traversal of this tree?
   a) 6 2 1 3 10 8
   b) 2 1 3 10 8 6
   c) 6 2 1 3 10 8
   d) 1 2 3 6 8 10
   e) 1 3 2 8 10 6

10. Given the following code, what is output?.

```
public static int recurse(int n) {
    if (n <= 1)
      return 1;
    else
      return  recurse(n-3) + n;
}
public static void main(String [] args)  {
        System.out.println ( recurse(10) );
}
```

   a) **22** b) 21 c) 13 d) 7 e) none of the above

11. Given the following code, what is output?

```
public static int dolt (int num) {
         int total = 0;
         for (int i=0; i<num; i++)
                  for (int j=num-i; j>0; j--)
                           total+=j;
        return total;
}
public static void main(String [] args)  {
        System.out.println (dolt(2));
}
```

   a)7 b) 4 **c) 3** d) 5 e) 6 f) none of the other answers

12. Given the following numbers added in this order to a binary tree 6, 10, 8, 2, 3, 1
   What is the in-order traversal of this tree?
   a) 6 2 1 3 10 8
   b) 2 1 3 10 8 6
   c) 6 2 1 3 10 8
   d) 1 2 3 6 8 10
   e) 1 3 2 8 10 6

Part 2 – Short Answer (21 marks)

13. Given the following code assuming Stack is a class which implements the operations of a stack data structure, what is in the stack (and in what order) when it completes: **(2 marks)**

```
Stack   stack1 = new Stack();
stack1.push(5);
stack1.push (7);
stack1.push(2);
stack1.push (stack1.pop( ) - stack1.pop( ));
stack1.push(10 );
stack1.push(stack1.pop( ));
```

10
9
5
---
bottom of stack

14. Given the following code assuming Queue is a class which implements the operations of a queue data structure, what is in the queue (and in what order) when it completes: **(2 marks)**

```
Queue   queue1 = new Queue();
queue1.push(5);
queue1.push (7);
queue1.push(2);
queue1.push (queue1.pop( ) - queue1.pop ( ));
queue1.push (10) ;
queue1.push(queue1.pop( ));
```

2
10
12
---
start of queue

15. Fill in the following table with Big-O efficiencies and a description of how that efficiency is achieved and memory allocations for implementing a stack and queue data structure: **(10 marks)**

Criteria	Stack using dynamically allocated array	Queue using doubly linked list
Memory allocation for n items on the structure	***reference for start of array *block of n references (for data) *n blocks of size of object on stack**	***two references for head/tail *n blocks of 3 references (1 for data, l for next, 1 for previous) *n blocks of size of object on queue**
push operation	**O(1) - add at end of array**	**O(1) - add at head or tail**
pop operation	**O(1) – remove at end of array**	**O(1) – remove at tail or head (opposite to add)**
isFull operation	**O(1) – check numEntries == maxSize**	**O(1) – always false**
isEmpty operation	**O(1) – check numEntries == 0**	**O(1) – check head == null**

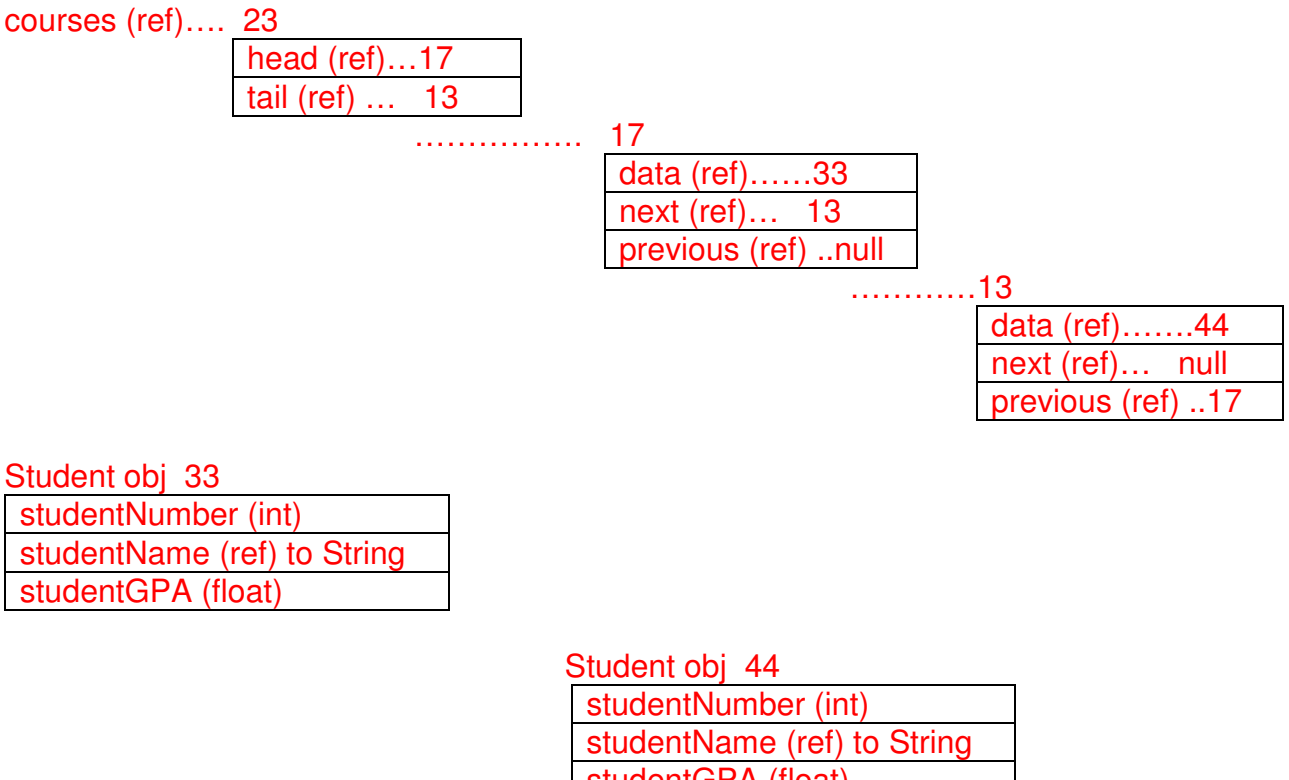
16. Describe two advantages to implementing a stack data structure with a singly linked list compared to using a dynamically allocated array. ***(2 marks)*** - **don't need block of n references which may be hard to allocate; is never full**

17. Given the following declaration and the **Student** and **Course** classes from above**:**

LinkedList <Course> courses = new LinkedList<Course>();

Remember that the implementation of **LinkedList** is a doubly linked list with head and tail references.

Draw a memory map after adding 2 elements to **courses**. Include the memory for **Course** (in general). **(5 marks)**



Part 3 – Programming Questions (11 marks)

18. Using the class **Student** and **Course** from above, write an initial constructor for the class **Course** that takes two parameters – one for the initial size of the object **students** and the other for the name of the course. **(4 marks)**

```
public Course (int size, String name) {
       courseName = new String (name);
       if (size > 0)
           students = new ArrayList<Student> (size);
      else
           students = new ArrayList<Student> ();
}
```

19. Using the class **Student** and **Course** from above, assume in main, the structure has been filled with data properly. Write the code necessary for a method called **count** in **Course** class which takes a **(float) gpaValue** and returns an **int** count of the number of objects in the structure whose **studentGPA** value is greater than or equal to the parameter **gpaValue**. If you need to add any methods to Student class, write it/them as well. **(7 marks)**

Write the method in **Course** class here:

```
public int  count (float gpaValue) {

     int num = 0;
     for (int  = 0; i<students.size(); i++)
          if (students.get(i).isEligible(gpaValue)
               num++;

     return num;
}
```

Write any methods needed in **Student** class here:

```
public boolean isEligible (float gpaValue) {
     if (studentFPA >= gpaValue)
          return true;
     else return false;
}
```