This exam is closed book, closed notes. No electronic devices (including calculators) are allowed. Be concise, but show your work. Write legibly. When writing code, please indent appropriately and give your variables meaningful names.

Time: 120 minutes.
Question 1 (16 points):
Answer the following questions regarding a stack class with the following structure:

```java
public class Stack {
    private int[] stackData;
    private int stackTop;

    public Stack(int size) {
        stackData = new int[size];
        stackTop = -1;
    }

    public void push(int item) {
        ...}

    public int pop() {
        ...}

    public boolean isEmpty() {
        return (stackTop == -1);
    }
}
```

a) Assume the push() and pop() methods are implemented correctly. What is the output of the following program? (4 points)

```java
public class StackQuestion {
    public static void main(String[] args) {
        Stack s = new Stack(10);
        s.push(3);
        s.push(7);
        s.push(2);
        s.push(3);
        s.push(5);

        while (!s.isEmpty()) {
            int item = s.pop();
            System.out.print(item + " ");
        }
    }
}
```

Output:

```
5 3 2 7 3
```
b) Provide an implementation of the push() method. (6 points)

```java
public void push(int item) {
    stackData[++stackTop] = item;
}
```

c) Provide an implementation of the pop() method. (6 points)

```java
public int pop() {
    return stackData[stackTop--];
}
```
Question 2 (24 points):
Please answer the following questions regarding sorting of arrays with integer or String values. It is assumed that a sorted array has the lowest integer or String value at the lowest index (i.e., ascending order). Assume a suitable swap() method is provided.

a) Describe how the following code differs from bubblesort. (4 points)

```java
static public void sort(int[] values) {
    if (values == null || values.length == 0) {
        return;
    }
    for (int i = 1; i <= values.length - 1; i++) {
        for (int j = i; j > 0; j--) {
            if (values[j - 1] > values[j]) {
                swap(values, j - 1, j);
            }
        }
    }
}
```

The loop boundaries are different. In bubblesort, the outer loop (i) goes from 0 to length-1 and the inner loop (j) goes from 0 to length-i-1. In the given code, the range of the inner loop increases, while the inner loop of bubblesort decreases in range. Finally, the inner loop goes in the opposite direction of the outer loop, while in bubblesort both loops go in the same direction.

b) Does the code in a) sort the array correctly? Clearly state “yes” or “no” and provide an explanation. (4 points)

Yes. The given code maintains a sorted region in the left (lower index) part of the array. In each iteration, the next (unsorted) element of the array is “bubbled” leftward. Thus, the sorted region grows with i. This sorting approach is similar to insertion sort, but comparisons are performed all the way to the left-most element in the sorted region.

c) What is the asymptotic computational complexity of the code shown in a) as a function of the array size, n? Please explain your answer. (2 points)

O(n^2), because the comparison operation is performed 1+2+3+...+n-2 = (n-2)(n-1)/2 times.
d) Complete the following method to implement sorting using the insertion sort algorithm. Note that the array contains String variables. (12 points)

```java
static public void insertionSort(String[] values) {
    // Check for empty or null array
    if (values == null || values.length == 0) {
        return;
    }

    // do actual sorting
    for (int i = 1; i < values.length; i++) {
        String temp = values[i];
        int j = i;
        while (j > 0 && values[j - 1].compareTo(temp) > 0) {
            values[j] = values[j - 1];
            j--;
        }
        values[j] = temp;
    }
}
```

e) What is the asymptotic computational complexity of your code in d) as a function of the array size, n? Please explain your answer. (2 points)

$O(n^2)$, because the outer loop (i) runs n-1 times and the inner loop runs up to i-1 times. Thus, the sum is the same order as in c).
Question 3 (18 points):
Given an array of integers sorted in **descending** order, answer the following questions regarding binary search.

a) Complete the following method to implement binary search. The data array contains integers in ascending order and is filled completely (i.e., from index 0 to index data.length-1). If the value that is searched for (item) is in the array, then your method should return the index of that occurrence in data. If the search item is not in the array, your method should return -1. Use variable names “lo”, “hi”, and “mid” for the appropriate operations in your binary search method. (12 points)

```java
static int binarySearch(int[] data, int item) {
    int lo = 0;
    int hi = data.length-1;
    while (lo<=hi) {
        int mid = (lo+hi)/2;
        if (item > data[mid]) {
            hi = mid-1;
        } else if (item < data[mid]) {
            lo = mid+1;
        } else {
            return mid;
        }
    }
    return -1;
}
```

b) Assume data contains the following 8 items:

```java
data = [18,17,13,12,11,8,5,3]
```

Consider the values for lo, hi, and mid for your code in a) when searching for integer 5. Show the values of the three variables at the beginning of each iteration in the following table. (6 points)

<table>
<thead>
<tr>
<th>iteration</th>
<th>lo</th>
<th>hi</th>
<th>mid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>
Question 4 (18 points):
Answer the following questions regarding links and objects. Consider the following code:

```java
public class LinksAndObjectsQuestion {

    static private class SomeObject {
        public int x;
        public SomeObject link;

        public SomeObject(int value) {
            x = value;
            link = null;
        }

        public String toString() {
            return "(\"x\")";
        }
    }

    public static void main(String[] args) {
        SomeObject a, b, c;

        a = new SomeObject(3);
        b = new SomeObject(4);
        c = new SomeObject(5);
        a.link = b;
        b.link = c;
        c.link = a;

        System.out.println(a.link,"+b.link","+c.link);
        // answer question a)

        a.link = b.link;
        b.link = c.link;
        c.link = a.link;

        System.out.println(a.link,"+b.link","+c.link);
        // answer question b)

        a.link.x = 6;
        b.link.link.x = 7;
        c.link.link.link.x = 8;

        System.out.println(a.link,"+b.link","+c.link);
        // answer question c)
    }
}
```
a) What is the output of the first `System.out.println` instruction? Provide an illustration of the objects, their values, and their link relationships. (Only showing the output will not give you full points.) (6 points)

(4), (5), (3)

b) What is the output of the second `System.out.println` instruction? Provide an illustration of the objects, their values, and their link relationships. (Only showing the output will not give you full points.) (6 points)

(5), (3), (5)

c) What is the output of the first `System.out.println` instruction? Provide an illustration of the objects, their values, and their link relationships. (Only showing the output will not give you full points.) (6 points)

(8), (3), (8)
Question 5 (24 points):
Answer the following questions regarding linked lists. For this question, you can assume that the linked list is singly linked and only has one pointer to the beginning of the list. Each list element should store a string that is provided by the user.

a) Write the code that is necessary to define the class(es) necessary to create the required linked list. Your classes only need to specify variables used within that class as well as one suitable constructor method. You do not need to provide any other methods. (6 points)

```java
public class ListObject {
    public String text;
    public ListObject next;

    public ListObject(String t) {
        text = t;
        next = null;
    }
}

public class LinkedList {
    ListObject first;

    public LinkedList() {
        first = null;
    }
}
```

b) Assume that a list with multiple items based on your answer in a) is provided. Write a method concatenate() that returns a string that is a concatenation of all elements in the linked list. The first element in your linked list should be at the beginning of the string. (i.e., if your list contains elements “abc” (first element) and “def” (second element), you should return “abcdef”). (8 points)

```java
public String concatenate() {
    String s = "";
    ListObject current = first;

    while (current!=null) {
        s = s+current.text;
        current = current.next;
    }
    return s;
}
```
c) Write a method deleteEveryOther() that removes every other element from the list starting with the second element. (I.e., the method deletes elements 2, 4, 6, etc.) (10 points)

```java
public void deleteEveryOther() {
    ListObject current = first;

    while (current != null) {
        if (current.next != null) {
            current.next = current.next.next;
            // delete element after current
        }
        current = current.next;
    }
}
```