CSC 1351: Exam On Everything

Name and Last Digit of Student ID: ____________________________________________

1 Easy

1.1
What are the primitive types?

2 Syntax

2.1
Write a HelloWorld program that prints “Hello World” to the screen.

2.2
Write a minimal Java program.

3 Inheritance

3.1
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

class A {}
class B extends A {}
public class Main {
    public static void main(String[] args) {
        B b = new A();
    }
}

3.2

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
interface Planet { void orbit(); }
public class Mars implements Planet {
    public void orbit() { }
    public static void main(String[] args) {
        Planet a = new Planet();
        Mars b = new Mars();
    }
}
```

The code does not compile. Replace the code: `Planet a = new Planet();` with the code: `Planet a = new Mars();`

3.3

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
class A {}
public class B extends A {
    public static void main(String[] args) {
        A a = new B();
        if(a instanceof B) b = a;
    }
}
```

The code does not compile. Replace the code: `if(a instanceof B) b = a;` with the code: `B b = (B)a;`

3.4

Fill in the missing code.

```java
class A {}
public class B extends A {
    public static void main(String[] args) {
        A a = new B();
        B b = null;
        // ask if the value in a can be assigned to b
        if(----------------) {
            // set b using the value in a
            ----------------
        }
    }
}
```

3.5

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?
interface Moo {
    void moo(int a) {}
}

public class M implements Moo {
    public void moo(int a) {}
}

The code does not compile.

replace the code: void moo(int a) {}
with the code: void moo(int a);

3.6

Give then code below, write the code for the interface named Transformer.

public class Trans {
    public static void main(String[] args) {
        Transformer t = new Transformer() {
            public int transform(int n) { return n*2+1; }
        };
        System.out.println("t.transform(3)="+t.transform(3));
    }
}

3.7

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

class Rocket {
    int energy = 3;
    void launch() { energy -= 2; check(); }
    void check() {
        if(energy <= 0) System.out.println("out of power");
        else if(energy <= 1) System.out.println("low power");
    }
}

public class Starship extends Rocket {
    void launch() { energy -= 5; check(); }
    public static void main(String[] args) {
        Rocket r1 = new Rocket();
        Rocket r2 = new Starship();
        r1.launch(); r2.launch();
    }
}

3.8

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

4 Arrays

4.1

The code should print out 3, 9, 7, and 2 in sequence. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
import java.util.Iterator;
class IntIterator implements Iterator<Integer> {
    int[] array; int index = 0;
    IntIterator(int[] array) {
        this.array = array;
    }
    public boolean hasNext() {
        return index < array.length;
    }
    public Integer next() {
        return array[index++];
    }
}
class MyIter {
    public static void main(String[] args) {
        IntIterator ii = new IntIterator(new int[]{3,9,7,2});
        while(ii.hasNext()) {
            int v = ii.next();
            System.out.println(v);
        }
    }
}
```

4.2

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.Arrays;
class Foo {
    public static void main(String[] args) {
        String[] letters = {"O","C","F","N","A","L","S","H","V","Y"};
        String result = letters[5] + letters[1];
        result = letters[4] + result;
        result += letters[0];
        result = letters[2] + result + letters[3];
        StringBuilder sb = new StringBuilder();
        sb.append(result);
    }
}
```
for(int i=7; i<letters.length; i++)
    sb.append(letters[i]);
System.out.println(sb.toString());

4.3
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
public class ArraysInJava {
    public static void main(String[] args) {
        int[] i = new int[0];
        System.out.println(i[0]);
    }
}
```

The code compiles, but when it runs it throws an `ArrayIndexOutOfBoundsException`. Replace the code:

```java
int[] i = new int[0];
```

with the code:

```java
int[] i = new int[1];
```

4.4
Keep only array elements that are more than 3. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class MoreThan3 {
    public static int[] moreThan3(int[] a) {
        int n1 = 0, n2 = 0;
        for(int i=0; i<a.length; i++)
            if(a[i] > 3) n1++;
        int[] b = new int[n1];
        for(int i=0; i<a.length; i++)
            if(a[i] > 3) b[n2++] = a[i];
        return b;
    }
    public static void main(String[] args) {
        int[] r1 = new int[]{5, 2, 7, 9, 8, 3};
        int[] r2 = moreThan3(r1);
        System.out.println(Arrays.toString(r2));
    }
}
```

// output:
[5, 7, 9, 8]

4.5
Write another version of the `moreThan3` method that works with Lists.
4.6

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.Arrays;

public class Foo {
    public static void main(String[] args) {
        String result = letters[5] + letters[1];
        result = letters[4] + result + letters[2];
        result = result + result.substring(0, 3) + letters[0];
        result = result + letters[2] + letters[3] + result.charAt(2) + result.charAt(0);
        System.out.println(result);
    }
}
```

4.7

Write a method that, given an array of integers and an integer s, prints all pairs of array elements whose sum is equal to s. Don’t print other pairs, print pairs only once, and print the smaller number first. Assume the array does not contain duplicate entries.

4.7.1

What is the big-O speed of your solution, if N is the size of the array?

4.8

What is wrong with the `add()` method? The method is supposed to add an element to the end of a doubly linked list. Also, assume the constructor of a Node to be of the form
public Node(Integer d, Node p, Node n) {
    data = d;
    previous = p;
    next = n;
}

public class MyListImpl implements MyList {
    Node start;
    Node end;

    public void add(Integer i) {
        Node n = new Node(i, end, null);
        end = n;
        if (start == null)
            start = n;
        if (end != null)
            end.next = n;
    }
}

4.9
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

public class Alist {
    int[] data = new int[]{3,4,5,9,7}; int size=4;
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append('{');
        for(int i=0;i<size;i++) {
            if(i > 0) sb.append(',');
            System.out.println("i="+i);
            sb.append(i);
        }
        sb.append('}');
        return sb.toString();
    }
    public static void main(String[] args) {
        System.out.println(new Alist());
    }
}

4.10
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

import java.util.Arrays;
public class Foo {
    public static void main(String[] args) {
        String[] letters = {"G","A","I","N","Y","W","R"};
        String result = letters[4] + letters[0];
        result = letters[5] + result + letters[2];
    }
}
result = result + result.substring(0, 3) + letters[1];
result = result + result.charAt(3) + letters[6] + letters[3] + result.charAt(2) +
letters[5];
System.out.println(result);
}

4.11

Implement an ArrayList: Unlike an array, an ArrayList needs to be able to support appending elements. Usually, an ArrayList is implemented by maintaining an array that is larger than required, and another variable is used to keep track of how many elements have been stored in the array. Fill in the missing code for the add method, doing what is appropriate for a generic class.

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

public class Alist {
    Object[] data = new Object[10];
    int size = 0;
    public void add(Object item) {
        if (size >= data.length) {
            Object[] newData = new Object[data.length + 10];
            for (int i = 0; i < data.length; i++)
                newData[i] = data[i];
            data = newData;
        }
        data[size++] = item;
    }
}

4.12

What is wrong with the add() method? The method is supposed to add an element to the end of a doubly linked list. Also, assume the constructor of a Node to be of the form

public Node(Integer d, Node p, Node n) {
    data = d;
    previous = p;
    next = n;
}

public class MyListImpl implements MyList {
    Node start;
    Node end;

    public void add(Integer i) {
        Node n = new Node(i, end, null);
        if (end != null)
            end.next = n;
        end = n;
    }
}
4.13
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
public class MyList {
    Object[] o = new Object[10];
    int size = 0;
    void add(Object d) { o[size++] = d; }
    Object get(int n) { assert n < size; return o[n]; }
    public static void main(String[] args) {
        MyList ml = new MyList();
        ml.add(3); ml.add(2); ml.add(1);
        System.out.println(ml.get(3));
    }
}
```

The code compiles, but when it runs it throws an `AssertionError` replace the code: `System.out.println(ml.get(3));` with the code: `System.out.println(ml.get(2));`

4.14
Write a method that creates a reversed copy of an array. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
import java.util.Arrays;
public class Reverse {
    static int[] reverse(int[] input) {
        int[] output = new int[input.length];
        --------------------
        int n = input.length-1;
        --------------------
        for(int i=0;i<input.length;i++)
            --------------------
            output[n--] = input[i];
            --------------------
        return output;
        --------------------
    }
    public static void main(String[] args) {
        int[] in = new int[]{3,8,2,4};
        int[] out = reverse(in);
        System.out.println(Arrays.toString(in));
        System.out.println(Arrays.toString(out));
    }
}
```

// output:

```
[3, 8, 2, 4]
[4, 2, 8, 3]
```

4.15
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.Arrays;
public class ReverseShift {
    // Shift values to the right, thus [0,1,2] should become [2,1,0]
```
static void shift(int[] s) {
    int tmp = s[s.length-1];
    for(int i=0;i<s.length-1;i++)
        s[i+1] = s[i];
    s[0] = tmp;
}

public static void main(String[] args) {
    int[] n = new int[3];
    for(int i=0;i<n.length;i++) n[i] = i;
    shift(n);
    for(int i=0;i<n.length-1;i++)
        for(int j=i+1;j<n.length;j++)
            assert(n[i] != n[j]);
    System.out.println(Arrays.toString(n));
}
}

4.16
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

public class Alist {
    Object[] data = new Integer[]{3,4,5,9,7}; int size=3;
    public void add(Object item) {
        if(size+1 >= data.length) {
            Object[] newData = new Object[data.length+10];
            for(int i=0;i<newData.length;i++)
                newData[i] = data[i];
            data = newData;
        }
        data[size++] = item; }
    public static void main(String[] args) {
        Alist a = new Alist();
        for(int i=0;i<20;i++) a.add(i); }
}

4.17
Find the best match between the column on the left and the column on the right.
4.18

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.*;
public class Shift {
    // shift values to the left
    static void shift(List<Integer> s) {
        int tmp = s.get(0);
        for(int i=1;i<s.size();i++)
            s.set(i-1,s.get(i));
        s.set(s.size()-1,tmp);
    }
    public static void main(String[] args) {
        List<Integer> n = new ArrayList<>();
        for(int i=0;i<3;i++) n.add(i);
        shift(n);
        for(int i=0;i<n.size()-1;i++)
            for(int j=i+1;j<n.size();j++)
                assert(n.get(i) != n.get(j));
        System.out.println(n);
    }
}
```

4.19

Return true if a list contains, somewhere, three adjacent numbers that double each time .... 2, 4, 8, ... or 3, 6, 12.

tripleTrouble([7, 1, 2, 4, 5, 3]) -> true
tripleTrouble([2, 4, 8]) -> true
tripleTrouble([0, -1, -2]) -> false
tripleTrouble([0, 0, 0]) -> true
tripleTrouble([]) -> false

The function prototype is this: boolean tripleTrouble(List<Integer> list) ... Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public static boolean tripleTrouble(List<Integer> array) {
    --------------------
    --------------------
    --------------------
    --------------------
    --------------------
}
```
4.20
What is the advantage of using an array class?

5 Anonymous Inner Class

5.1
Given the interface defined like this:

```java
public interface MouseListener {
    void mousePressed(int x, int y);
    void mouseReleased(int x, int y);
}
```

Write a complete program that uses an anonymous inner class that implements this interface, then calls mousePressed() to print the x and y values of the mouse event to the screen.

```java
public class Test {
    public static void main(String[] args) {
        MouseListener ml = new MouseListener() {
            public void mousePressed(int x, int y) {
                System.out.println("\(("+x+","+y+"")\)");
            }
            public void mouseReleased(int x, int y) {
            }
        };
        ml.mousePressed(230, 127);
    }
}
```

5.2
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
interface Operator { int eval(int a, int b); }
public class Anon {
    public static void main(String[] args) {
        Operator add =
            -------------------
            -------------------
            -------------------
            Operator mul =
            -------------------
            -------------------
            System.out.println("add="+add.eval(2,3)+"mul="+mul.eval(2,3));
    }
}
```

// output:
add=5 mul=6
5.3
Given the interface defined like this:

```java
public interface KeyListener {
    void keyPressed(char key);
    void keyReleased(char key);
}
```

Write a complete program that uses an anonymous inner class that implements this interface, then calls keyPressed() to print the key character 'A' to the screen.

```java
public class Test {
    public static void main(String[] args) {
        KeyListener kl = new KeyListener() {
            public void keyPressed(char c) {
                System.out.println(c); }
            public void keyReleased(char c) {
                System.out.println(c); }
        };
        kl.keyPressed('A'); } } 
```

5.4
The following program should print the string "Run!" to the screen.
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
interface RunMe { void run(); }
public class Runner {
    public static void main(String[] args) {
        RunMe rm =
            new RunMe() {
            --------------
            public void run() {
            --------------
                System.out.println("Run!");
            --------------
        }
        --------------
        rm.run(); }}

    // output:
    Run!
```

5.5
Figure out the anonymous inner class Fill in the missing code.

```java
import java.util.*;

public class Sorter {
    public static void main(String[] args) {
        Random rand = new Random();
        List<Integer> li = new ArrayList<>();
        for(int i = 0;i< 10;i++)
            li.add(rand.nextInt(20));
        // Comparator < Integer > has a single method
        // public int compare(Integer a, Integer b)
        // which must be supplied.
        Comparator<Integer> c = ---------------------------
        new Comparator<Integer>() {
            public int compare(Integer a,Integer b) {
            return b - a;
            }
        }
```
5.6

Given the interface defined like this:

```java
public interface Logger {
    void log(String msg);
}
```

Write a complete program that uses an anonymous inner class that implements this interface to print the message msg to the screen.

```java
public class Test {
    public static void main(String[] args) {
        Logger log = new Logger() {
            public void log(String msg) {
                System.out.println(msg);
            }
        };
        log.log("Hello World");
    }
}
```

5.7

Fill in the missing code.

```java
// interface Runnable {
//     public void run();
// }
public class Anon {
    public static void main(String[] args) {
        Runnable r = ----------------------
        ----------------------
        ----------------------
        ----------------------
        };  
        r.run();
    }
}
```

// output:

Hello, world
5.8

Fill in the missing code.

```java
interfaceGetInt {
    publicintvalue();
}

publicclassAnon2 {
    publicstaticvoidmain(String[]args) {
       GetIntgi = -------------
            ---------------
                ---------------
                    -------------
                        ---------------

        System.out.println("value="+gi.value());
    }
}

//output:
value=33
```

6 Recursion

6.1

Rewrite the following code to use recursion (no loops!).

```java
publicclassHello {
    publicstaticvoidhello(intn) {
        for(inti=0;i<n;i++)
            System.out.println("Hello:"+i);
    }
    publicstaticvoidmain(String[]args) {
        hello(3);
    }
}
```

6.2

From CodingBat.

Given a string, compute recursively a new string where all the adjacent chars are now separated by a "*".

### 6.3

What is wrong with the following program, what happens when you run it? If this method would have been implemented recursively, and would have a similar error, what would happen then if run?

```java
class JavaIsToJavascriptWhatCarIsToCarpet {
    public static int factorial(int n) {
        int result = 1;
        while (n > 1) {
            result *= n;
        }
        return result;
    }
    public static void main(String[] args) { 
        assert(factorial(4) == 24);
    }
}
```

There is an exit-condition missing in the while loop. When run, the program hangs. This is similar to infinite recursions, so the same in a recursive program would result in a stack overflow.

### 6.4

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
class Series {
    public static int func(int j) {
        System.out.println(j);
        if (j==1) return 1;
        return 2*func(j-1) + 5*func(j-2); }
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        if (N<1) {
            System.out.println("invalid argument");
            return;
        }
        System.out.println(func(N));
    }
}
```
6.5
Martian mathematicians have uncovered a mysterious recursive formula. Implement it below. No loops allowed.
MarsCode a:

1. if $a < 10 \rightarrow 10$
2. if $a$ is odd $\rightarrow$ three times the value MarsCode $((a+1)/2)$ plus the value of MarsCode $((a-1)/2)$
3. if $a$ is even $\rightarrow$ two times the value of MarsCode $(a/2)$

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class MarsCode {
    public static int eval(int a) {
        // ---------------
        // ---------------
        // ---------------
    }
    public static void main(String[] args) {
        for(int i=20;i<25;i++) {
            System.out.println("eval(\"+"+\")="+eval(i));
        }
    }
}
// output:
eval(20)=40
eval(21)=140
eval(22)=80
eval(23)=100
eval(24)=40
```

6.6
From CodingBat:
Given a string, compute recursively a new string where all the lowercase 'x' chars have been moved to the end of the string.

```java
public static String endX(String x) {
    return endX(0,x);
}
public static String endX(int n,String x) {
    if(n == x.length()) return "";
    if(x.charAt(n) == 'x')
        return ""+endX(n+1,x)+"x";
    return ""+x.charAt(n)+endX(n+1,x);
}
```

6.7
You want to implement flipping a coin on a computer. However, you want to be fancy and include the possibility of the coin standing on edge, with a probability of 2%. In that case you flip it again, until the coin lands on either side. “Heads” give 1 point, “Tails” give 0 points, and an edge gives an additional 2 points (potentially multiple times). Most of the time the return value will be 0 or 1. However, if the coin landed on edge once, the return value will be either 2 or 3, depending on the final toss. If it landed on edge twice, it could be either 4 or 5, in a similar way. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.
This task involves a game with plush owls. You start the game with a given number of owls. You can then give back some of the owls, according to the rules below. If multiple rules apply, it is your choice which to use. If none applies, you loose. The goal is to end up with exactly 42 owls. \( n \) is the number of owls you currently have.

- If \( n \) is even, you may give back exactly \( n/2 \) owls.
- If \( n \) is divisible by 3 or 4, you may multiply the last two digits of \( n \) and give back this many owls. (the last digit of \( n \) is \( n\%10 \), the next-to-last digit is \( (n\%100)/10 \).)
- If \( n \) is divisible by 5, then you may give back exactly 42 owls.

For example, suppose you start with 250 owls, you could do the following moves:

- Since 250 is divisible by 5, you may return 42 owls, leaving you with 208.
- Since 208 is even, you may return half of them, leaving you with 104.
- Since 104 is even again, you may do the same again, leaving you with 52.
- Since 52 is divisible by 4, you may multiply the last two digits (2*5=10), and return 10 owls, leaving you with 42 owls, and resulting in a win.

Write a recursive function to meet this specification:

```java
// The return value should indicate whether it is possible to win this game if
// you start with \( n \) owls. For example:
// owls(250) is true
// owls(42) is true
// owls(84) is true
// owls(53) is false
// owls(41) is false
public class Owls {
    public static boolean owls(int n) {

}
6.9

The following program determines whether exactly 3 of the supplied values can be combined to produce the target value. Fill in the missing code. Use recursion. No loops. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class GroupSum3 {
    public static boolean sum3(int[] values, int target) {
        return sum3(0, values, target, 3);
    }

    public static boolean sum3(int start, int[] values, int target, int count) {
        if (start >= values.length) return target == 0 && count == 0;
        boolean ret = sum3(start + 1, values, target - values[start], count - 1);
        if (!ret) ret = sum3(start + 1, values, target, count);
        return ret;
    }
}
```

6.10

What is wrong with the following program, what happens when you run it? If this method would have been implemented using a loop, and would have a similar error, what would happen then if run?

```java
public class JavaIsToJavascriptWhatCarIsToCarpet {
    public static int factorial(int n) {
        return (n * factorial(n - 1));
    }

    public static void main(String args[]) {
        factorial(4);
    }
}
```

There is an exit-condition missing in the recursive function. When run, you get a stack overflow. This is similar to infinite loops, so the same in an iterative program would result in a hang during execution.

6.11

What is wrong with the following program, what happens when you run it? If this method would have been implemented recursively, and would have a similar error, what would happen then if run?

```java
public class JavaIsToJavascriptWhatCarIsToCarpet {
    public static int factorial(int n) {
        int result = 1;
        while (n > 1) {
            result *= n;
        }
        return result;
    }

    public static void main(String args[]) {
        assert (factorial(4) == 24);
    }
}
```

There is an exit-condition missing in the while loop. When run, the program hangs. This is similar to infinite recursions, so the same in a recursive program would result in a stack overflow.

6.12

Implement the factorial() method in the JavaIsToJavascriptWhatCarIsToCarpet class above recursively, correctly.
6.13
Write a recursive program to compute a double factorial, \( n!! = n*(n-2)!!, \) \( 0!! = 1, \) \( 1!! = 1. \)

6.14
What is recursion?
1. Recursion is a generic class.
2. Recursion is a process of setting a value based on it’s previous value.
3. Recursion is a process of defining a method that calls itself.
4. Recursion is a process of repeatedly calling other methods.

6.15
You want to implement combat within a role playing game on a computer. Specifically, the game rules for damage inflicted by a hit are:

- In order to figure out damage from one hit, you throw a N-sided die.
- The result of one throw will be between 1 and \( N \) (including both, e.g., a 6-sided die has six sides, labeled 1 to 6).
- If the result is 1 to \( N-1 \), that is the resulting damage from the hit.
- If the result in \( N \), however, you hit critically, and you throw again, adding the results.
- If you throw again, the same rules apply, potentially resulting in doubly or more critical hits.

For example, if you use a 4-sided die and throw a 3, the damage is 3. If you throw a 4 instead, you throw again. If that results in a 3, the total damage is 7. If you happen to throw two 4s after each other and then a 2, the total damage is 10. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
final static Random rand = new Random();
public static int damage(int n) {
    -----------------------
    if (result < n) return result;
    -----------------------
    return result + damage(n);
    -----------------------
}
```
6.16

What is the output of this program?

class recursion {
    int func (int n) {
        int result;
        result = func (n - 1);
        return result;
    }
}

class Output {
    public static void main(String args[]) {
        recursion obj = new recursion();
        System.out.print(obj.func(12));
    }
}

1. 0
2. 1
3. 12
4. Compilation Error
5. Runtime Error

6.17

Given a List of Integers, is it possible to choose a group of Integers, such that the sum of all Integers in that group matches a given target with this additional constraint: The number of integers chosen must be odd.

Do not use loops.

Example inputs and returns for groupOdd():

[[],0] false
[1],1 true
[1,1,3], 2 false
[1,1,3], 4 false
[1,1,3], 5 true
[1,2,3], 4 false

Fill in the missing code.

public static boolean groupOdd(List<Integer> list, int target) {
    ---------------------
}

public static boolean groupOdd(--------------------------) {
    ---------------------
    ---------------------
    ---------------------
    ---------------------
    ---------------------
    return works;
}
Given an List of Integers, is it possible to choose a group of Integers, such that the sum of all Integers in that group matches a given target with this additional constraint: If a value in the List is chosen to be in the group, the value immediately following it in the List must not be chosen. Do not use loops.

Example inputs and returns for `groups()`:

```
[], 0    false
[1], 0   false
[1], 1   true
[1,2], 1 true
[1,2], 3 false
[1,0], 1 true
[1,0], 0 true
[1,2,3], 4 true
```

You may (but don’t have to) use the `subList` method, defined in `java.util.List`, which is declared as follows, and returns a part (slice) of the input array (fromIndex inclusive, toIndex exclusive):

```
List<E> subList(int fromIndex, int toIndex)
```

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public static boolean groups(List<Integer> list, int target) {
    if (list.size() == 0) return false;
    if (list.get(0) == target) return true;
    if (list.size() >= 2 &&
        groups(list.subList(2, list.size()), target-list.get(0)) == true) return true;
    return groups(list.subList(1, list.size()), target);
}
```

This task involves a game with plush bunnies. You start the game with a given number of bunnies. You can then give back some of the bunnies, according to the rules below. If multiple rules apply, it is your choice which to use. If none applies, you lose. The goal is to end up with exactly 42 bunnies. `n` is the number of bunnies you currently have.

- If `n` is even, you may give back exactly `n/2` bunnies.
- If `n` is divisible by 3 or 4, you may multiply the last two digits of `n` and give back this many bunnies. (the last digit of `n` is `n%10`, the next-to-last digit is `((n%100)/10)`.
- If `n` is divisible by 5, then you may give back exactly 42 bunnies.

For example, suppose you start with 250 bunnies, you could do the following moves:

- Since 250 is divisible by 5, you may return 42 bunnies, leaving you with 208.
- Since 208 is even, you may return half of them, leaving you with 104.
- Since 104 is even again, you may do the same again, leaving you with 52.
- Since 52 is divisible by 4, you may multiply the last two digits (2*5=10), and return 10 bunnies, leaving you with 42 bunnies, and resulting in a win.

Write a recursive function to meet this specification:

```
// The return value should indicate whether it is possible to win this game if
// you start with n bunnies. For example:
// bunnies(250) is true
// bunnies(42) is true
```
// bunnies(84) is true
// bunnies(53) is false
// bunnies(41) is false
public class Bunnies {
    public static boolean bunnies(int n) {
        // smaller than 42
        if (n < 42) return false;
        // equal 42
        if (n == 42) return true;
        // divisible by 2
        if (n % 2 == 0 && bunnies(n / 2)) return true;
        // divisible by 3 or 4
        if ((n % 3 == 0 || n % 4 == 0) && bunnies(n - ((n % 10) * ((n % 100) / 10)))) return true;
        // divisible by 5
        if (n % 5 == 0 && bunnies(n - 42)) return true;
        return false;
    }
}

6.20
Which of these will happen if recursive method does not have a base case?

1. An infinite loop occurs, hanging forever.
2. The program stops after some time with an error.
3. After 1000000 calls it will be automatically stopped.
4. None of the mentioned

6.21
public class Exp {
    // Compute the exponential function
    public static double exp(double x) {
        if (x < .001) {
            return 1 + x + x * x / 2 + x * x * x / 6;
        } else {
            double ex = exp(0.5 * x);
            return ex * ex;
        }
    }
    public static void main(String[] args) {
        for (double x = 0; x < 10; x += 2.0) {
            System.out.printf("exp(%f)=%f (err=%f)\n", x, exp(x), Math.abs(Math.exp(x) - exp(x)));
        }
    }
}

// output:
exp(0.000000)=1.000000 (err=0.000000)
exp(2.000000)=7.389056 (err=0.000000)
exp(4.000000)=54.598150 (err=0.000000)
exp(6.000000)=403.428793 (err=0.000000)
exp(8.000000)=2980.957986 (err=0.000001)
7 Sort and Search

7.1
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class LunarLander implements Comparable {
    boolean manufacturedOnMars; // sort by this first, true is higher priority
    String name; // sort by this second
    int energyStorage; // sort by this third

    @Override
    public int compareTo(Object o) {
        LunarLander that = (LunarLander) o;
        int diff = 0;
        if(this.manufacturedOnMars) diff -= 1;
        if(that.manufacturedOnMars) diff += 1;
        if(diff == 0) diff = this.name.compareTo(that.name);
        if(diff == 0) diff = this.energyStorage - that.energyStorage;
        return diff;
    }
}
```

7.2
A special sort routine can be written if we are sorting integers in the range 0 to N-1. It looks like this:

```java
public static void sort(int N, int[] values) {
    int[] boxes = new int[N]; int n = 0;
    for(int i=0;i<values.length;i++)
        boxes[values[i]]++;
    for(int i=0;i<boxes.length;i++)
        while(boxes[i] > 0) {
            values[n++] = i;
            boxes[i]--;
        }
}
```

Does it work? What is its big O behavior in terms of the length of the array L? Is it O(L log L)? O(L^2)? Something else? Justify your answer.

7.3
What is the time complexity of each of these sort methods in terms of the array size, N?

1. Bubble Sort
2. Selection Sort
3. Merge Sort
4. Quick Sort
7.4

What kind of sort is this? Rewrite it to use ArrayList.

```java
static void sort(int end, int[] values) {
    if (end < 2)
        return;
    for (int i = 1; i < end; i++) {
        if (values[i - 1] > values[i]) {
            int tmp = values[i];
            values[i] = values[i - 1];
            values[i - 1] = tmp;
        }
    }
    sort(end - 1, values);
}
```

This is a bubble sort because it's constantly comparing adjacent elements.

```java
static void sort(int end, ArrayList<Integer> values) {
    if (end < 2)
        return;
    for (int i = 1; i < end; i++) {
        if (values.get(i - 1) > values.get(i)) {
            int tmp = values.get(i);
            values.set(i, values.get(i - 1));
            values.set(i - 1, tmp);
        }
    }
    sort(end - 1, values);
}
```

7.5

What kind of sort is this? Rewrite it to use ArrayList.

```java
static void sort(final int start, final int end, int[] values) {
    if (end - start < 1)
        return;
    int[] scratch = new int[end - start + 1];
    int r = rand.nextInt(scratch.length) + start;
    int p = values[r];
    int lo = 0, hi = scratch.length - start;
    int numps = 0;
    for (int i = start; i <= end; i++) {
        if (values[i] < p)
            scratch[lo++] = values[i];
        else if (values[i] == p)
            numps++;
        else
            scratch[--hi] = values[i];
    }
    int n = start;
    for (int i = 0; i < lo; i++)
        values[n++] = scratch[i];
    for (int i = 0; i < numps; i++)
        ...
values[n++] = p;
for(int i=hi;i<scratch.length;i++)
    values[n++] = scratch[i];
sort(start,start+lo-1,values);
sort(start+hi,end,values);
}

The random number generation gives it away. This is a quicksort.

static void sort(final int start,final int end,ArrayList<Integer> values) {
    if(end-start < 1)
        return;
    int[] scratch = new int[end-start+1];
    int r = rand.nextInt(scratch.length)+start;
    int p = values.get(r);
    int lo = 0, hi = scratch.length;
    int numps = 0;
    for(int i=start;i<=end;i++) {
        if(values.get(i) < p)
            scratch[lo++] = values.get(i);
        else if(values.get(i) == p)
            numps++;
        else
            scratch[--hi] = values.get(i);
    }
    int n = start;
    for(int i=0;i<lo;i++)
        values.set(n++,scratch[i]);
    for(int i=0;i<numps;i++)
        values.set(n++,p);
    for(int i=hi;i<scratch.length;i++)
        values.set(n++,scratch[i]);
    sort(start,start+lo-1,values);
    sort(start+hi,end,values);
}

7.6

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

public class Bug implements Comparable {
    String type;  // sort by this if num_legs and num_eyes are equal
    int num_legs; // sort by this first
    int num_eyes; // sort by this if num_legs is equal
    @Override
    public int compareTo(Object o) {
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
                        ---------------------
    }
}
7.7

Make the above code generic.

7.8

What kind of sort is this? Something was done incorrectly, however. What is wrong? Fix it.

```java
static void sort(int[] values) {
    sort(values, 0, values.length);
}
static void sort(int[] values, int start, int end) {
    if (end - start < 2) return;
    int mval = values[(start + end) / 2];
    int i1 = start, i2 = end;
    for (int i = start; i < end; i++) {
        if (values[i] < mval) {
            int tmp = values[i];
            values[i] = values[i1];
            values[i1] = tmp;
            i1++;
        }
    }
    for (int i = end - 1; i >= start; i--) {
        if (values[i] > mval) {
            i2--;
            int tmp = values[i];
            values[i] = values[i2];
            values[i2] = tmp;
        }
    }
    assert i1 < i2 : "i1=" + i1 + "i2=" + i2 + "start=" + start + "end=" + end ;
    sort(values, start, i1); sort(values, i2, end);
}
```

7.9

What kind of sort is this? Rewrite it to use ArrayList.

```java
static void sort(int start, int[] values) {
    if (values.length - start < 2)
        return;
    int m = start;
    for (int i = start + 1; i < values.length; i++)
        if (values[i] < values[m])
            m = i;
    int tmp = values[m];
    values[m] = values[start];
    values[start] = tmp;
```
7.10
Find the best match between the column on the left and the column on the right.

(1) bubble sort   (a) compares adjacent elements
(2) quick sort   (b) can always be rewritten as recursive
(3) iterative   (c) uses a random number
(4) binary search (d) keeps finding the next smallest element
(5) merge sort   (e) is $O(N \log N)$
(6) recursion    (f) is $O(\log N)$
(7) selection sort (g) can cause StackOverflowError

7.11
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class Bug implements Comparable {
    int num_legs; // sort by this first
    int num_eyes; // sort by this if num_legs is equal
    // sort by this if the names are equal
    @Override
    public int compareTo(Object o) {
        Bug that = (Bug) o;

        -------------------
        int diff = this.num_legs - that.num_legs;
        -------------------
        if (diff != 0) return diff;
        -------------------
        return this.num_eyes - that.num_eyes;
        -------------------
    }
}
```

7.12
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class Cat implements Comparable {
    int numKittens; // sort by this if the names are equal
    String name; // sort by this first
    @Override
    public int compareTo(Object o) {
    }
}
```
7.13

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class MarsLander implements Comparable {
    boolean canTakeOff; // sort by this first, true is higher priority
    String homeCountry; // sort by this second
    int fuelCap;     // sort by this third

    @Override
    public int compareTo(Object o) {
        MarsLander that = (MarsLander) o;
        int diff = 0;
        if (this.canTakeOff) diff -= 1;
        if (that.canTakeOff) diff += 1;
        if (diff == 0) diff = this.homeCountry.compareTo(that.homeCountry);
        if (diff == 0) diff = this.fuelCap - that.fuelCap;
        return diff;
    }
}
```

7.14

What kind of sort is this? Something was done incorrectly, however. What is wrong? Fix it.

```java
public static Random RAND = new Random();
static void sort(int[] values) {
    sort(values, 0, values.length);
}
static void sort(int[] values, int start, int end) {
    if (end - start < 2) return;
    int mid = RAND.nextInt(end - start) + start;
    sort(values, start, mid);
    sort(values, mid, end);
    int[] newvalues = new int[end - start];
    int i1 = start, i2 = mid, i3 = 0;
    while (i1 < mid && i2 < end) {
        if (values[i1] <= values[i2]) {
            newvalues[i3++] = values[i1++];
        } else {
            newvalues[i3++] = values[i2++];
        }
    }
    while (i1 < mid) newvalues[i3++] = values[i1++];
    while (i2 < end) newvalues[i3++] = values[i2++];
```
for(int i=0;i<newvalues.length;i++)
    values[i+start] = newvalues[i];
}

8 Collections

8.1
What is the difference between a Java Set and a List?

8.2
Name four interfaces within the Java collections framework.

8.3
The program below is intended to prove that radar is a palindrome. If the assertion fails, it fails. Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.*;
public class Palindrome {
    public static void main(String[] args) {
        String pal = "radar";
        Queue<Character> queue = new LinkedList<>();
        Stack<Character> stack = new Stack<>();
        for (int i=0; i<pal.length(); i++) {
            char c = pal.charAt(i);
            queue.add(c);
            stack.push(c);
        }
        while (queue.size()>0) {
            Character st = stack.remove();
            Character qu = queue.remove();
            assert st.equals(qu);
        }
    }
}
```
8.4

Does the following code compile? If it does not, how can it be fixed? If it does compile, does it still contain an error? If so: how can it be fixed?

class Node {
    Integer data;
    Node next;

    public Node() {}  
    public Node(Integer d, Node n) {
        data = d;
        next = n;
    }

    // Returns a String representing this, and following elements.
    public String nextString() {
        String ret = Integer.toString(data);
        return ret + "", next.nextString();
    }
}

class Test {
    public static void main(String[] args) {}
}

8.5

Which interface directly extends the Collection interface and is implemented in the ArrayList class?

8.6

Suppose the list letters contains elements “S”, “P”, “R”, “I”, “N”, and “G”. Draw the contents of the list and the iterator position for the following operations:

ListIterator<String> iter = letters.iterator();  
iter.next();
iter.remove();
iter.next();
iter.remove();
iter.next();
iter.add("A");
iter.next();
iter.next();
iter.next();
iter.remove();
8.7
Name two interfaces that Stack is implementing.

8.8
Name two classes within the Java collections framework that implement the List interface, but no other interface besides the ones that List is extending.

8.9
Which interface directly extends the Collection interface and is implemented in the LinkedList class?

8.10
The program below is intended to prove that radar is a palindrome. If the assertion fails, it fails. Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.*;
public class Palindrome {
    public static void main(String[] args) {
        String pal = "radar";
        Queue<Character> queue = new LinkedList<>();
        Stack<Character> stack = new Stack<>();
        for (int i=0; i<pal.length(); i++) {
            char c = pal.charAt(i);
            queue.add(c);
            stack.push(c);
        }
        while (queue.size()>0) {
            Character st = stack.pop();
            Character qu = queue.remove();
            assert !st.equals(qu);
        }
    }
}
```
8.11
Assume elements 2, 4, 6, 8 being put element-wise first into a queue, taken out again, then put onto a stack, taken out again, put onto another stack, and taken out again. In which order do you now have these elements, and which order were they after each step?

8.12
Is Stack an interface or a class? What other interface or class does it directly extend or implement, and is that an interface or a class?

8.13
Assume elements 2, 4, 6, 8 being put element-wise first onto a stack, taken out again, then put into a queue, taken out again, put onto a stack, and taken out again. In which order do you now have these elements, and which order were they after each step?

8.14
Which class is Stack inheriting from, and which interface is that class implementing?

8.15
Assume elements 2, 4, 6, 8 being put element-wise first into a queue, taken out again, then put onto a stack, taken out again, put into a queue, and taken out again. In which order do you now have these elements, and which order were they after each step?
8.16
Complete the code within the `containsNot()` method, returning `false` if the doubly linked list contains an element with the value of `i`, and `true` otherwise.

```java
class Node {
    Integer data;
    Node previous;
    Node next;
}

public class MyList {
    Node start;
    Node end;

    public boolean containsNot(Integer i) {
    
    }
}
```

8.17
What is wrong with the following code to evaluate the top of the operators and numbers stack of an expression evaluator? How could you fix it?

```java
static Stack<Double> numbers = new LinkedList<>();
static Stack<String> operators = new LinkedList<>();
static void evaluateTop() {
    Double n2 = numbers.pop();
    Double n1 = numbers.pop();
    String op = operators.pop();
    if (op.equals("-"))
        numbers.push(n1-n2);
    else if (op.equals("+"))
        numbers.push(n1+n2);
    else if (op.equals("*"))
        numbers.push(n1*n2);
    else if (op.equals("/"))
        numbers.push(n1/n2);
}
```

8.18
Name three classes within the Java collections framework that implement the `List` interface.

```java
ArrayList, LinkedList, Vector, Stack
```
8.19
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.Stack;
public class Calc {
    Stack<Character> ops = new Stack<>();
    Stack<Integer> vals = new Stack<>();
    public Calc() {
        ops.push('+'); ops.push('-'); ops.push('+');
        vals.push(1); vals.push(2); vals.push(3); vals.push(4); }
    Integer eval() {
        while(!ops.isEmpty()) {
            Character c = ops.pop();
            Integer v1 = vals.pop();
            Integer v2 = vals.pop();
            if(c == '+') vals.push(v1+v2);
            else if(c == '-') vals.push(v1-v2);
        }
        return vals.pop(); }
    public static void main(String[] args) {
        Calc c = new Calc();
        System.out.println("c="+c.eval());
    }
}
```

c=6

8.20
Suppose the list `letters` contains elements “S”, “P”, “R”, “I”, “N”, and “G”. Draw the contents of the list and the iterator position for the following operations:

```java
ListIterator<String> iter = letters.iterator();
iter.next();
iter.next();
iter.remove();
iter.next();
iter.remove();
iter.next();
iter.remove();
iter.add("U");
iter.next();
iter.next();
iter.remove();
```
8.21
Name three interfaces within the Java collections framework that directly extend the Collection interface.

8.22
What is wrong with the following code to evaluate the top of the operators and numbers stack of an expression evaluator? How could you fix it?

```java
static Stack<Double> numbers = new Stack<>();
static Stack<String> operators = new Stack<>();
static void evaluateTop() {
    Double n2 = numbers.peek();
    Double n1 = numbers.peek();
    String op = operators.peek();
    if (op.equals("-"))
        numbers.push(n1-n2);
    else if (op.equals("+"))
        numbers.push(n1+n2);
    else if (op.equals("*"))
        numbers.push(n1*n2);
    else if (op.equals="/")
        numbers.push(n1/n2);
}
```

The peek() doesn't remove elements off the stack, use pop() instead.

8.23
Complete the code within the contains() method, returning true if the doubly linked list contains an element with the value of i, and false otherwise.

```java
class Node {
    Integer data;
    Node previous;
    Node next;
}

public class MyList {
    Node start;
    Node end;

    public boolean contains(Integer i) {
        Node current = start;
        while (current != null) {
            if (i.equals(current.data)) return true;
            current = current.next;
        }
        return false;
    }
}
```

8.24
A code that checks to see if W is a palindrome puts every character in W into both a stack and a queue (using push() and add()). It then repeatedly takes characters out of the stack and queue (using pop() and remove()) and compares them until
the stack and queue are empty. If all the pairs of characters are equal, then \( W \) is a palindrome. Why does this work?

9 Time Complexity (Big O)

9.1

The following table gives approximate running times for a program with \( N \) inputs for various values of \( N \).

<table>
<thead>
<tr>
<th>( N )</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>8 seconds</td>
</tr>
<tr>
<td>2000</td>
<td>1:04 minutes ( \approx ) 64 seconds</td>
</tr>
<tr>
<td>5000</td>
<td>15 minutes ( \approx ) 1000 seconds</td>
</tr>
<tr>
<td>10000</td>
<td>2.2 hours ( \approx ) 8000 seconds</td>
</tr>
</tbody>
</table>

Which of the following best describes the likely running time of this program for \( N = 20,000 \)?

1. A few hours (6,400 seconds)
2. About a day (64,000 seconds)
3. A few days (256,000 seconds)
4. About a week (640,000 seconds)

9.2

What is the Big-O notation for the following function?

\[
T(N) = 1000N + 0.0003N^2
\]  

(1)

9.3

What is the Big O speed for a linear search? For a merge sort? For an insertion sort?

For linear search it’s \( O(N) \).

For merge sort it’s \( O(N \log N) \).

For insertion sort it’s \( O(N^2) \).

9.4

What is the Big-O notation for the following function?

\[
T(N) = N(N^2 - N \log N)
\]  

(2)

9.5

Find the best match between the column on the left and the column on the right.

| (1) \( O(n^2) \) | (a) linear search |
| (2) Requires a sorted list | (b) insertion sort |
| (3) \( O(n \log n) \) | (c) bubble sort |
| (4) Repeatedly find the smallest value | (d) quick sort |
| (5) Uses a random number | (e) merge sort |
| (6) \( O(n) \) | (f) binary search |
9.6
What is the Big-O notation for the following function?
\( T(N) = 100000 + 10N + N^{-2} \)

9.7
What is the Big-O notation for the following function?
\( T(N) = 100000 + 10N - 3 + N^2 \)

9.8
What is the Big-O notation for the following function?
\[
T(N) = 3N + \frac{37N + 93\sqrt{N^7}}{N^2}
\]

9.9
What is the Big O speed for a binary search? For a bubble sort? For a quick sort?

9.10
What is the Big-O notation for the following function?
\( T(N) = 100000N^3 + 10N + N^{-2} \)

9.11
The following table gives approximate running times for a program with \( N \) inputs for various values of \( N \).

<table>
<thead>
<tr>
<th>( N )</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>5     seconds</td>
</tr>
<tr>
<td>2000</td>
<td>20    seconds</td>
</tr>
<tr>
<td>5000</td>
<td>2     minutes</td>
</tr>
<tr>
<td>10000</td>
<td>8     minutes</td>
</tr>
</tbody>
</table>

Which of the following best describes the likely running time of this program for \( N = 100,000 \)?

1. A few minutes
2. A few hours
3. Half a day
4. A few days
10 Exceptions

10.1
What is the Mars location Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
public class Mars {
    String location = "Valles_Marineris"; double fuel = 10;
    void travel(double fuelUsed,String location) throws Exception {
        if(fuelUsed > fuel) {
            fuel = 0; throw new Exception();
        } else {
            fuel -= fuelUsed; location = "Gale_Crater"; }
    }
    public String toString() { return "{{"+location+"":"+fuel+"}}"; }
    public static void main(String[] args) throws Exception {
        Mars m1 = new Mars(); Mars m2 = new Mars();
        try {
            m1.travel(9.5,"Gale_Crater");
            m2.travel(10.5,"Meducae_Fossae");
        } catch(Exception e) {
            System.out.println(m1);
            System.out.println(m2);
        } finally {
            System.out.println("travel_complete");
        }
        System.out.println("Success");
    }
}
```

```
{{Valles Marineris:0.5}}
{{Valles Marineris:0.0}}
travel complete
Success
```

I’ll also take
```
{{ Gale Crater :0.5}}
{{ Valles Marineris :0.0}}
travel complete
Success
```

10.2
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
public class Oops {
    static void foo(int[] a,int n) {
        int i = a[n];
        System.out.println("i="+i);
        foo(a,i);
    }
    public static void main(String[] args) {
        try {
            try {
                foo(new int[]{1,2,3,4,-1},0);
                System.out.println("zero");
            } catch(Exception e) {
                foo(null,0);
                System.out.println("one");
            } finally {
                System.out.println("two");
            }
        } catch(Exception e2) {
```
10.3
What is the bank balance at the end of main? Why?

```java
public class Bank {
    double balance;
    Bank(double b) { balance = b; }
    void debit(double amount) throws Exception {
        if(amount > balance) throw new Exception();
        balance -= amount;
    }
    public static void main(String[] args) throws Exception {
        Bank b = new Bank(5.2);
        try {
            b.debit(2.0);
        } catch(Exception e) {
            b.debit(1.0);
        }
    }
}
```

Subtracting 1.0 does not cause an exception to be thrown, so the exception handling code does not execute. 3.2

10.4
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
public class Stacker {
    static void stack(int n) {
        if(n == 0) throw new RuntimeException();
        System.out.println("before="+n);
        stack(n-1);
        System.out.println("after="+n);
    }
    public static void main(String[] args) {
        Integer a = null;
        try {
            stack(2);
        } catch(RuntimeException re) {
        }
    }
}
```
10.5
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class Stacker {
    static void stack(int n) {
        if(n == 0) throw new RuntimeException();
        stack(n-1); }
    public static void main(String[] args) {
        Integer a = null;
        try {
            stack(4);
        } catch(RuntimeException re) {
            re.printStackTrace();
            --------------------
        }
    }
}
```

// output:
```
java.lang.RuntimeException
    at Stacker.stack(Stacker.java:3)
    at Stacker.stack(Stacker.java:4)
    at Stacker.stack(Stacker.java:4)
    at Stacker.stack(Stacker.java:4)
    at Stacker.stack(Stacker.java:4)
    at Stacker.main(Stacker.java:8)
```

10.6
What is the bank balance at the end of main? Why?

```java
public class Bank {
    double balance;
    Bank(double b) { balance = b; }
    void debit(double amount) throws Exception {
        if(amount > balance) throw new Exception();
        balance -= amount;
    }
    public static void main(String[] args) throws Exception {
        Bank b = new Bank(5.2);
        try {
            b.debit(10.0);
        } catch(Exception e) {
            b.debit(1.0);
        }
    }
}
```

10.7
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class Show {
    public static void main(String[] args) {
        --------------------
        System.err.println("Exception printed!");
    }
}
```

// output:
When is an infinite loop not infinite? Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
public class Code {
    public static void main(String[] args) {
        int sum = 0, count = 0;
        int[] arr = new int[]{1, 2, 3, 2, 1};
        try {
            int n = 0;
            while(true) {
                sum += arr[n++];
                count++;
            }
        } catch(ArrayIndexOutOfBoundsException ex) {} 
        System.out.println("sum = " + sum);
    }
}
```

10.9

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
class Exa extends RuntimeException {}
class Exb extends RuntimeException {}
public class Flow {
    public static void main(String[] args) {
        try {
            try {
                System.out.println("step 1");
                if(true) throw new Exa();
            } catch(Exa a) {
                System.out.println("step 2");
                if(true) throw new Exb();
            } finally {
                System.out.println("step 3");
                System.out.println("step 4");
            } catch(Exception ex) {} 
        }
    }
}
```

10.10

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?
public class Oops {
    static void foo(int[] a, int n) {
        int i = a[n];
        System.out.println("i=\n");
        foo(a, i);
    }
    public static void main(String[] args) {
        try {
            try {
                foo(new int[]{2, 3, 6, 4, -1}, 0);
                System.out.println("zero");
            } catch(Exception e) {
                foo(null, 0);
                System.out.println("one");
            } finally {
                System.out.println("two");
            }
            catch(Exception e2) {
                System.out.println("three");
            } finally {
                System.out.println("four");
            }
        } catch(Exception e) {
            System.out.println("three");
        } finally {
            System.out.println("four");
        }
    }
}

public class Npe {
    public static void main(String[] args) {
        Integer a = null;
        try {
            int b = a + 3;
        } catch(NullPointerException npe) {
            npe.printStackTrace();
        }
    }
}

// output:
java.lang.NullPointerException
    at Npe.main(Npe.java:5)

public class Code {
    public static void main(String[] args) {
        try {
            try {
                System.out.println("HelloAlice");
                if(true) throw new NullPointerException("TheEnd");
                System.out.println("GoodbyeAlice");
            } finally {
                System.out.println("HelloBob");
            }
        }
    }
}
11 File IO

11.1
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.io.*;
import java.util.Scanner;
public class FilesRead {
    public static void main(String[] args) {
        File f = new File("/etc/group");
        if(f.exists()) {
            Scanner s = new Scanner(f);
            while(s.hasNextLine())
            System.out.println(s.nextLine());
        }
    }
}
```

The code does not compile.

Replace the code: `public static void main(String[] args) {` with: `public static void main(String[] args) throws IOException {`

11.2
What is java.io.File, and what is it used for?

11.3
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.io.*;
import java.util.Scanner;
public class FilesRead {
    public static void main(String[] args) {
        File f = new File("birds.txt");
        FileWriter fw = new FileWriter(f);
        fw.write("Holy Mackerel
        fw.write("Holy Mackerel\n")
        fw.close();
    }
}
```
11.4
Fill in the missing code.

import java.io.IOException;
import java.util.Scanner;
public class FilesWrite {
    public static void main(String[] args) throws IOException {
        String fileName = "/tmp/out.txt";
        FileWriter fw = -------------------------
        fw.write("Hello");
        fw.close();
    }
}

11.5
Write a code that prints “Hello, World.” to a file named hello.txt.

import java.io.*
import java.util.*
public class HelloFile {
    public static void main(String[] args) throws IOException {
        File f = new File("hello.txt");
        FileWriter fw = new FileWriter(f);
        fw.write("Hello, World\n");
        fw.close();
        Scanner s = new Scanner(f);
        assert s.nextLine().trim().equals("Hello, World");
        f.delete();
    }
}

11.6
Formatting using String.format() or System.out.printf().
Find the best match between the column on the left and the column on the right.

| (1) %d | (a) floating point with 2 decimal places |
| (2) %3s | (b) integer, width 4 chars, zeros in front |
| (3) %4d | (c) string |
| (4) %f | (d) string, width 3 chars |
| (5) %04d | (e) floating point |
| (6) %s | (f) new line, os independent |
| (7) %.2f | (g) integer, width 4 chars |
| (8) %n | (h) integer |

11.7
Reading and Writing from files.
Find the best match between the column on the left and the column on the right.
(1) FileWriter  (a) write to memory
(2) BufferedWriter (b) read from a file
(3) Scanner   (c) write to a file
(4) Writer    (d) parse input
(5) StringWriter (e) has printf
(6) PrintWriter (f) just an optimization
(7) FileReader (g) superclass of FileWriter and StringWriter

11.8
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class FilesRead {
    public static void main(String[] args) throws IOException {
        String fileName = "/etc/group";
        File file = new File(fileName);

        if(file.exists()) {
            Scanner s = new Scanner(file);
            while(s.hasNextLine()) {
                System.out.println(s.nextLine());
            }
        }
    }
}
```

12 Regular Expressions

12.1
Fill in the missing code.

```java
String in = "225-123-4567
982-333-4444
115-765-4321"
Scanner s = new Scanner(in);
while(s.hasNextLine()) {
    String pattern = "(\d+)-(\d+)-(\d+)";
    if(s.findInLine(pattern) != null) {
        MatchResult mr = s.match();
        int areaCode = Integer.parseInt(mr.group(1));
        int exchange = Integer.parseInt(mr.group(2));
        int number = Integer.parseInt(mr.group(3));
        System.out.printf("(%03d)%03d-%04d%n",areaCode,exchange,number);
    } else {
        System.out.println("Incorrect");
    }
    s.nextLine();
}
```

// output:
(225)123-4567
(982)333-4444
(115)765-4321

12.2
Fill in the missing code.

```java
String in = "Deadpool␣891-22-3954
Wade␣113-24-5566"
```
Scanner s = new Scanner(in);
while(s.hasNextLine()) {

    String pattern = "(\w+)\s+(\d+)-(\d+)-(\d+)";
    if(s.findInLine(pattern) != null) {
        MatchResult mr = s.match();

        int a = Integer.parseInt(mr.group(2));
        int b = Integer.parseInt(mr.group(3));
        int c = Integer.parseInt(mr.group(4));
        String d = mr.group(1);
        System.out.printf("Name:␣%s,␣ssn:␣%d-%d-%d%n",d,a,b,c);
    }
    s.nextLine();
}

// output:
Name: Deadpool, ssn: 891-22-3954
Name: Wade, ssn: 113-24-5566

12.3
Fill in the missing code.

String in = "Methuselah,␣age:938
Wilma,age:␣45
";
Scanner s = new Scanner(in);
while(s.hasNextLine()) {

    String pattern = "(\w+)\s*,\s*age:\s*(\d+)";
    if(s.findInLine(pattern) != null) {
        MatchResult mr = s.match();

        int a = Integer.parseInt(mr.group(2));
        String b = mr.group(1);
        System.out.printf("Name:␣%s,␣age:␣%d%n",b,a);
    }
    s.nextLine();
}

// output:
Name: Methuselah, age: 938
Name: Wilma, age: 45

12.4
Fill in the missing code.

String in = "9+␣104␣=113
8␣+4=␣5
";
Scanner s = new Scanner(in);
while(s.hasNextLine()) {

    String pattern = "(\d+)\s*\+\s*(\d+)\s*=\s*(\d+)";
    if(s.findInLine(pattern) != null) {
        MatchResult mr = s.match();

        int a = Integer.parseInt(mr.group(1));
        int b = Integer.parseInt(mr.group(2));
        int c = Integer.parseInt(mr.group(3));
        if(a + b == c) System.out.println("Correct");
    }
}
else System.out.println("Incorrect");
} s.nextLine();

// output:
Correct
Incorrect

12.5
Regular expressions. Find the best match between the column on the left and the column on the right.

<table>
<thead>
<tr>
<th>(1) *</th>
<th>(a) means a digit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2) \w</td>
<td>(b) means a group</td>
</tr>
<tr>
<td>(3) \t</td>
<td>(c) means a whitespace character (space, tab, newline)</td>
</tr>
<tr>
<td>(4) \b</td>
<td>(d) means word boundary</td>
</tr>
<tr>
<td>(5) \n</td>
<td>(e) means tab</td>
</tr>
<tr>
<td>(6) \s</td>
<td>(f) means a letter, digit, or underscore</td>
</tr>
<tr>
<td>(7) ()</td>
<td>(g) means anything</td>
</tr>
<tr>
<td>(8) \d</td>
<td>(h) means line feed</td>
</tr>
<tr>
<td>(9) .</td>
<td>(i) means 1 or more</td>
</tr>
<tr>
<td>(10) +</td>
<td>(j) means 0 or more</td>
</tr>
</tbody>
</table>

12.6
Fill in the missing code.

```java
String in = "Mon␣Jan␣2,␣1982
Wed␣Aug␣14,␣1979\n";
Scanner s = new Scanner(in);
while(s.hasNextLine()) {
    String pattern = ----------------------------------------
    if(s.findInLine(pattern) != null) {
        MatchResult mr = s.match();
        String weekday = -----------;
        String month = -----------;
        int day = -----------------------------;
        int year = -----------------------------;
        System.out.printf("Date:␣%s␣%d/%s/%d%n",weekday,day,month,year);
    } else { System.out.println("Incorrect");
} s.nextLine();

// output:
Date:␣Mon 2/Jan/1982
Date:␣Wed 14/Aug/1979
```

13  Linked Lists

13.1
Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class Linked {
    static class Node { Object data; Node next, prev; }
    Node head, tail;
    public void add(Object d) {
        Node n = new Node();
```
13.2
What disadvantages do linked lists have over arrays (2)?

• They require more overhead, specifically O(N) overhead.
• It is slow to access specific element by position: O(N).

13.3
What advantages do linked lists have over arrays (2)?

• They don’t require a continuous memory block.
• It is cheap to insert anywhere in the list: O(N).
• It is cheap to remove anywhere in the list: O(N).

13.4
Fill in the missing code.

```java
public class Link {
    String value;
    final Link next;
    public Link(String val, Link next) {
        value = val; this.next = next;
    }
    public static void main(String[] args) {
```
Link l1 = new Link("a", new Link("b", new Link("c", null)));

for (Link x = --------------------------) {
    System.out.println("item = " + x.value);
}

// output:
item = a
item = b
item = c

13.5
Create a circular singly linked list. This means that the “next” field on Node should never be null. It also means if you keep following the “next” pointer, you will come back to the node you started with.
Note that instead of keeping two pointers, one for head and one for tail, we have a single tail pointer.
Fill in the missing code.

class Node { Node next; Object data; }
public class CircularLinkedList {
    Node tail;
    public Object getHead() { ------------------------ }
    public void add(Object d) {
        Node n = new Node();
        --------------------
        n.data = d;
        --------------------
        if (tail == null) {
            --------------------
            tail = n.next = n;
            --------------------
        } else {
            --------------------
            n.next = tail.next;
            --------------------
            tail.next = n;
            --------------------
            tail = n;
            --------------------
        }
    }
}

13.6
Recursion is fun. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

class Link {
    String value;
    final Link next;
    public Link(String val, Link next) {
        value = val; this.next = next;
    }
    public String toString() {
        ------------------------
        ------------------------
    }
}
```java
public static void main(String[] args) {
    Link l1 = new Link("a", new Link("b", new Link("c", null)));
    System.out.println(l1);
}
}
```

// output:
```
a,b,c
```

14 Binary Trees

14.1

The following values are added to a binary tree. Please draw the resulting tree structure. d b e f a c g

14.2

Consider the Btree below: Fix the add method. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```
import java.util.*;
class Node<T> {
    T data;
    Node<T> L, R;
    Node(T v) { data = v; } }
public class BTree<T extends Comparable<T>> {  
    Node<T> head;
    public void add(T key) {
        if(head == null) head = new Node<>(key);
        else add(head, key);
    }
    private void add(Node<T> n, T key) {
        if (n.data.compareTo(key) < 0) {
            if (n.L == null) n.L = new Node<>(key);
            else add(n.L, key);
        } else {
            if (n.R == null) n.R = new Node<>(key);
            else add(n.R, key);
        }
    }
}
```
14.3
Find the best match between the column on the left and the column on the right.

| (1) HashMap          | (a) requires a Comparator or Comparable |
| (2) Set              | (b) requires a hashCode               |
| (3) TreeMap          | (c) an Interface                      |
| (4) TreeSet          | (d) key value pairs traversed in order |
| (5) ArrayList        | (e) unique items                      |
| (6) HashSet           | (f) efficient item lookup             |
| (7) Queue            | (g) a subclass of Vector              |
| (8) Stack            | (h) efficient insert and delete       |
| (9) LinkedList       | (i) key values pairs maybe in order   |

14.4
Which of the following values is the best hash code for String s?
1. s.charAt(0)
2. 1
3. s.length()

   ← This one

Answer 1 won’t work if the string has zero length.
Answer 2 is usable, but highly inefficient.
Answer 3 will always work and provide some discrimination between items.

14.5
What is wrong with this code?

```java
import java.util.*;
public class Monster implements Comparable<Monster> {
    int numTeeth, numClaws;
    String name;
    Monster(String n,int t,int c) { name = n; numTeeth = t; numClaws = 2; }
    public int compareTo(Monster that) {
        int diff = this.numTeeth - that.numTeeth;
        if(diff == 0) diff = this.numClaws - that.numClaws;
        return diff;
    }
}
```

The number of teeth and claws are not final, therefore sorts will be undone and binary tree lookups will fail.

14.6
What is wrong with this code?

```java
import java.util.*;
public class MathWizard {
    final int level;
    int spells, id;
    MathWizard(int id,int lvl,int spells) {
        this.id = id; this.level = lvl; this.spells = spells;
    }
    public int hashCode() { return id; }
    public static void main(String[] args) {
        Set<MathWizard> s = new TreeSet<>();
        s.add(new MathWizard(1,2,3));
    }
}
```
14.7

The following values are added to a binary tree. Please draw the resulting tree structure. e d b c h f n m

14.8

The following binary tree implementation needs to be fleshed out.

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

class Node {
    Comparable item;
    Node L, R;
    Node(Comparable c) { item = c; }
}

class BTree {
    Node head;
    public void add(Comparable item) {
        if(head == null) head = new Node(item);
        else add(head,item);
    }
    private static void add(Node n,Comparable item) {
        int d=item.compareTo(n.item);
        if(d < 0) {
            // add left
            add(n.L,item);
        } else if(d > 0) {
            // add right
            add(n.R,item);
        }
    }
}

14.9

The following binary tree implementation needs to be fleshed out and made generic. Do both things.

Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

class Node {
    Comparable item;
    Node L, R;
    ...
Consider the Btree below: Fix the add method. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
import java.util.*;

class Node {
    int data;
    Node L, R;
    Node(int v) { data = v; }
}

class BTree {
    Node head;
    public void add(Comparable item) {
        if (head == null) head = new Node(item);
        else add(head, item);
    }
    private static void add(Node n, Comparable item) {
        int d = item.compareTo(n.item);
        if (d < 0) {
            if (n.L == null) n.L = new Node(item);
            else add(n.L, item);
        } else if (d > 0) {
            if (n.R == null) n.R = new Node(item);
            else add(n.R, item);
        }
    }
}
```

14.10
public void add(int key) {
    if(head == null) head = new Node(key);
    else add(head,key); }
private void add(Node n,int key) {
    if(key > n.data) {
        ---------------------
        else add(n.L,key);
    }
    if(key < n.data) {
        ---------------------
        else add(n.R,key);
    }
}
void fill(List<Integer> list) { fill(list,head); }
void fill(List<Integer> list,Node n) {
    fill(list,n.L);
    list.add(n.data);
    fill(list,n.R); }
public static void main(String[] args) {
    BTree b = new BTree();
    for(int i : new int[]{1,3,8,2,9,4,7,6,16}) b.add(i);
    List<Integer> list = new ArrayList<>();
    b.fill(list);
    System.out.println(list); }

14.11
What does the fill method do in the code above? What is wrong with it? How can you fix it?

14.12
What is the output of this code?

14.13
Consider the Btree below: Fix the add method. Fill in the missing code. The number of lines corresponds to the answer key.
Your code may vary.
import java.util.*;
class Node {
    int data;
    Node L, R;
    Node(int v) { data = v; }
}
public class BTree {
    Node head;
    public void add(int key) {
        if(head == null) head = new Node(key);
        else add(head,new Node(key)); }
    private void add(Node d,Node k) {
        if(k.data > d.data) {
            ---------------------
            else add(d.R,k);
        }
        if(k.data < d.data) {
            ---------------------
        }
}
void print() { print(head); }
void print(Node n) {
    if (n == null) return;
    System.out.print("("+n.data+")");
    print(n.R);
}

public static void main(String[] args) {
    BTree b = new BTree();
    for (int i : new int[]{1,3,8,2,9,4,7,6,16}) b.add(i);
    b.print();
}

14.14
What does the fill method do in the code above? What is wrong with it? How can you fix it?

14.15
What is the output of this code?

14.16
Consider the following binary tree.

```
f
 / \
 c  h
 /  \
 b  e  k
 /  /
 a  d  j  m
```
Which of the sequences of node values below could *NOT* have produced the tree?

1. f c h b e k a d j m
2. f h c k e b m j d a
3. f c b a e d h k j m
4. f c b a d e h k j m
5. f h k m j c e d b a

15 Hash Tables

15.1
Which of the following values is the best hash code for an ArrayList<Integer> alist

1. alist.get(0)
2. 666
3. alist.size()
Which of the following values is the best hash code for the Bug class:

```java
public class Bug {
    int legs, arms;
    String species;
}
```

1. arms + legs + species.hashCode()
2. System.identityHashCode(this)
3. super.hashCode()

The answer is "arms + legs + species.hashCode()". The other two answers are the same. Both will return the location of the object in memory. That means that even if arms, legs, and species are all the same, the hash codes might be different.

16 Generics

16.1

The following class holds the minimum value passed to it via the `setIfLower()` method. Make it contain a generic type instead of an Integer.

```java
public class MinHolder {
    Integer data;
    public void setIfLower(Integer d) {
        if(data == null || d < data) data = d;
    }
    public Integer get() { return data; }
}
```

```java
public class MinHolder<T extends Comparable<T>> {
    T data;
    public void setIfLower(T d) {
        if(data == null || d.compareTo(data) < 0) data = d;
    }
    public Integer get() { return data; }
}
```

16.2 Make a BTree of objects using Comparable.

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
class Node<T> {
    T data;
    Node<T> L, R;
    Node(T v) { data = v; }
}
```

```java
public class BTree<T> {
    Node<T> head;
    public void add(T key) {
        if(head == null) head = new Node<>(key);
        else add(head,key);
    }
    private void add(Node<T> n,T key) {
        if(key.compareTo(n.data) > 0) {
            if(n.L == null) n.L = new Node<>(key);
            else add(n.L,key);
        }
        if(key.compareTo(n.data) < 0) {
            if(n.R == null) n.R = new Node<>(key);
            else add(n.R,key);
        }
    }
}
```
16.3
Write a generic method to count the number of elements in an array that are equal to a given argument. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
import java.util.List;
public class Count {
    public static<T> int count(List<T> list, T arg) {
        int n = 0;
        for(T t : list)
            if(t.equals(arg)) n++;
        return n;
    }
}
```

16.4
Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
public class Min<T> {
    public static T getMin(T a, T b) {
        if(a.compareTo(b) < 0)
            return a;
        else
            return b;
    }
}
```

16.5
What are some advantages of using generic types? 

---

1from https://docs.oracle.com/javase/tutorial/java/generics/QandE/generics-questions.html
2from http://www.baeldung.com/java-generics-interview-questions
import java.util.*;
class genericstack <E> {
    Stack <E> stk = new Stack <E>();
    public void push(E obj)
    {
        stk.push(obj);
    }
    public E pop()
    {
        E obj = stk.pop();
        return obj;
    }
}

public class Output
{
    public static void main(String args[])
    {
        genericstack <String> gs = new genericstack<String>();
        gs.push("Hello");
        System.out.println(gs.pop());
    }
}

16.7 Make a BTree of objects using a Comparator.

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

class Node<T> {
    T data; Node<T> L, R;
    Node(T v) { data = v; }
}

class BTREE<T> {
    Node<T> head; Comparator<T> c;
    public BTREE(Comparator<T> c) { this.c = c; }
    public void add(T key) {
        if(head == null) head = new Node<>(key);
        else add(head, key);
    }
    private void add(Node<T> n, T key) {
        int diff = key.compareTo(n.data);
        if(diff > 0) {
            if(n.L == null) n.L = new Node<>(key);
            else add(n.L,key);
        } else if(diff < 0) {
            if(n.R == null) n.R = new Node<>(key);
            else add(n.R,key);
        }
    }
}
16.8 Sort using a Comparator.

Does the following code compile? If it does not, how can it be fixed? If it does, what is it’s output? Does it throw an exception? If so, how can it be fixed?

```java
public class Sorter<T> {
    Comparator<T> c;
    public Sorter(Comparator<T> c) { this.c = c; }
    void sort(List<T> list) {
        for(int i=0;i<list.size();i++) {
            for(int j=i+1;j<list.size();j++) {
                int diff = c.compare(list.get(i), list.get(j));
                if(diff > 0) {
                    T tmp = list.get(i);
                    list.set(i, list.get(j) );
                    list.set(j, tmp );
                }
            }
        }
    }
}
```

The code does not compile. Replace the code: list.get(i).compareTo(list.get(j)); with the code: int diff = c.compare(list.get(i), list.get(j));

16.9

The following code dies with a ClassCastException on the line indicated whenever it is used to permute a list with more than one element. What’s wrong? How would generics have solved the problem?

```java
import java.util.ArrayList;

public class Permute {
    public static ArrayList permutations(ArrayList list) {
        ArrayList permuted_list = new ArrayList();
        if(list.size() < 2)
            return list;
        for(int i=0;i<list.size();i++) {
            ArrayList shorter_list = new ArrayList();
            for(int j=0;j<list.size();j++) {
                if(i != j)
                    shorter_list.add(list.get(j));
            }
            for(Object o : permutations(shorter_list)) {
                // ClassCastException occurs on this line
                ArrayList ilist = (ArrayList)o;
                ilist.add(list.get(i));
                permuted_list.add(ilist);
            }
        }
        return permuted_list;
    }
}
```
16.10
Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.*;
public class Coord implements Comparable<Coord> {
    final int x,y;
    public Coord(int x,int y) { this.x = x; this.y = y; }
    @Override
    public int compareTo(Object o) { Coord that = (Coord)o;
        return 10000*(this.x - that.x) + (this.y - that.y);
    }
    public static void main(String[] args) {
        List Coord = new ArrayList();
        Coord.add(new Coord(3,4));
        Coord.add(new Coord(9,2));
        Collections.sort(Coord);
    }
}
```

The code does not compile. Replace the code: `public int compareTo(Object o) { Coord that = (Coord)o;` with the code: `public int compareTo(Coord that) {`.

16.11
Implement the following comparable interface and use generics. Fill in the missing code. The number of lines corresponds to the answer key. Your code may vary.

```java
public class RaceCar implements Comparable {
    public int compareTo(Object o) {
        RaceCar that = (RaceCar)o;
        ---------------------
        int diff = 0;
        ---------------------
        if(this.topSpeed < that.topSpeed) diff = 1;
        ---------------------
        else if(this.topSpeed > that.topSpeed) diff = -1;
        ---------------------
        if(diff == 0) diff = this.number - that.number;
        ---------------------
        return diff;
    }
    double topSpeed; // sort by this first
    int number; // sort by this if num_legs is equal
}
```
16.12   Sort Comparable objects.

Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
public class Sorter<T> {
    void sort(List<T> list) {
        for(int i=0; i<list.size(); i++) {
            for(int j=i+1; j<list.size(); j++) {
                int diff = list.get(i).compareTo(list.get(j));
                if(diff > 0) {
                    T tmp = list.get(i);
                    list.set(i, list.get(j));
                    list.set(j, tmp);
                }
            }
        }
    }
}
```

The code does not compile. Replace the code:

```java
public class Sorter<T> {
    void sort(List<T> list) {
        for(int i=0; i<list.size(); i++) {
            for(int j=i+1; j<list.size(); j++) {
                int diff = list.get(i).compareTo(list.get(j));
                if(diff > 0) {
                    T tmp = list.get(i);
                    list.set(i, list.get(j));
                    list.set(j, tmp);
                }
            }
        }
    }
}
```

16.13

What is type erasure? 4

16.14

This question will not be on the exam. 5 Does the following code compile? If it does not, how can it be fixed? If it does, what is its output? Does it throw an exception? If so, how can it be fixed?

```java
import java.util.*;

class Animal {}
class Cat extends Animal {}

public class Farm {
    List<Animal> animals = new ArrayList<>();
    public void addAnimals(List<Animal> moreAnimals) {
        animals.addAll(moreAnimals);
    }

    public static void main(String[] args) {
        List<Cat> cats = new ArrayList<>();
        Farm f = new Farm();
        f.addAnimals(cats);
    }
}
```

4 from http://www.baeldung.com/java-generics-interview-questions
5 from http://www.baeldung.com/java-generics-interview-questions
What if a generic type is omitted when instantiating an object? Will the code compile?  

---

6from http://www.baeldung.com/java-generics-interview-questions