Page 2	Page 3	Page 4	Page 5	Page 7	Total
					33

Average: 70%

LAST NAME	
FIRST NAME	
STUDENT NUMBER	

A

COMP 249 Midterm Test

Winter 2016
Section QQ

Duration: 75 minutes
No calculators or additional resources

Write only on the provided sheets, however you may write on the back of each question sheet if you need extra space

[01] (1 mark) Consider the following code snippet. Specify the output or exception generated by the following code.

```
int [] a={7,7,7,7,7};
int [] b={3,1,3,3,7};
int [][] c={b,a};
int [] d={};
System.out.println(b[a[3]-b[2]]);
```

[02] (1 mark) What will the output/exception be if the last line of Question [01] is replaced with:

```
System.out.println(c[b[1]][c.length]);
```

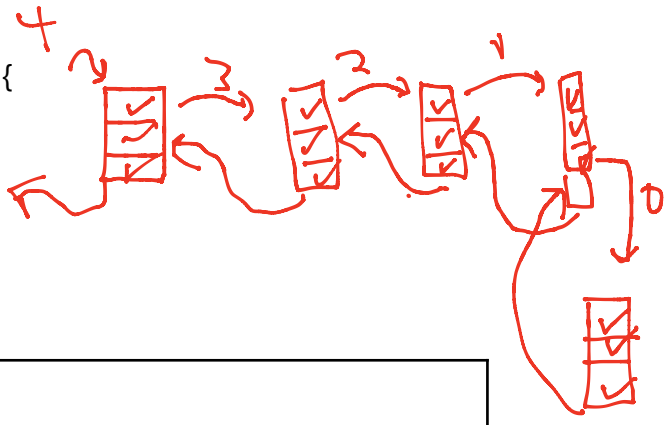
[03] (1 mark) What will the output/exception be if the last line of Question [01] is replaced with:

```
System.out.println(c[d.length].length);
```

[04] (3 marks) Consider the following method:

```
public static void Method(int a){
  if (a>1) Method (a-1);
  System.out.print(a);
  if (a==1) Method (a-1);
}
```

Write the output of a call to: `Method(4)`



[05] (3 marks) Write a method that returns the number of integers greater than or equal to 5 in a list of integers: *e.g.,* `Method(5,2,6)` returns 2 and `Method(3,1,3,3,7)` returns `1`. The list of parameters can be any size.

```
public int Method (int... a){
    int count = 0;
    for (int i : a){
        if (i >= 5) count++
    }
    return count;
}
```

[06] (8 Marks) An object `Child` extends the object `Parent` as follows.

```
public class Parent {
  public Parent () {
    this.hello2();
  }
  public static void hello(){System.out.println("Hi");}
  public void hello2(){System.out.println("Hey");}
}

public class Child extends Parent {
 public Child () {
   this.hello();
 }
 public static void hello() {System.out.println("Salute");}
 public void hello2() {System.out.println("Bonjour");}
 public void hello3() {System.out.println("Hello");}
}
```

For each line below, write the corresponding output if it is valid, or the note why the line is invalid.

Code Snippet	*Output*
`Child c = new Child();`	Bonjour Salute
`Parent c2 = new Child();`	Bonjour Salute.
`c.hello();` `c2.hello();`	Salute Hi
`c.hello2();` `c2.hello2();`	Bonjour Bonjour
`c2.hello3();`	Error: cannot find symbol
`System.out.println(` `  c.getClass()==c2.getClass());`	True
`Child c3 = new Parent();`	Error: incompatible types.
`Child c4 = (Child)c2;` `c4.hello3();`	Hello.

[07a] (4 Marks) Consider the following code snippet. There are different two problems with this code that will prevent it from compiling. Describe both issues.

```
try{
   PrintWriter f =
      new PrintWriter(FileOutputStream("src.txt",true));
   f.println("Writing line to file");
}

catch (Exception e){
   File ff = new File("src.txt");
   if (!ff.canWrite())
     System.out.println("File is read only.");
}

/* Continued on next page . . . */
```

```
catch (FileNotFoundException e){
   System.out.println("Unknown problem writing to file.");
}
finally{f.close();}
```

* f is local to try block but called from finally.
* Exceptions are out of order.

[07b] (2 Marks) Assuming the errors are corrected in the previous question, describe what happens if src.txt does not exist in the directory. What happens if a file named src.txt does exist in the same directory (and you have permission to write to it)?

* If it doesn't exist, creates it
* Append to the file.

[08] (10 marks)
Montreal has a bylaw that buildings cannot exceed 200m. The following code should read a file called `buildings.txt` which contains a list of integers. The integers represent the height of various proposed buildings. For each integer read, the code should print if the height is legal (e.g., 200 or below) or restricted (more than 200).

The code should use exception handling to make it robust against any problems being able to read from `buildings.txt`, and with any invalid integers that might be included in the file—specifically, negative numbers are disallowed and zero is disallowed. To assist with this, the following two new exceptions are created:

```
class InvalidIntException extends RuntimeException{};
class ZeroIntException extends InvalidIntException{};
```

The code below is missing certain parts, as identified by black boxes.

```
public static void main(String... args) {

  final int HEIGHT = 200;
  Scanner s = null;
  File f = new File("buildings.txt");

  try {
    s =  /* 01 */ ;
  }
  catch (FileNotFoundException e) {
    if (/* 02 */) {
      System.out.println("File does not exist");
    }
    else if (/* 03 */) {
      System.out.println("Cannot read file");
    }
    else {
      System.out.println("File error");
    }
  }

  int temp = 0;
  try {
    while(/* 04 */){
      temp = s.nextInt();
      if (temp==0) throw new ZeroIntException();
      else if (temp<0) throw new InvalidIntException();
      else if (temp<=HEIGHT) System.out.println(temp+" is
           legal");
      else System.out.println(temp+" is restricted");
    }
  }
  catch (/* 05 */) {
    System.out.println(/* 06 */);
  }
  catch (/* 07 */) {
    System.out.println(/* 08 */);
  }
  finally {System.out.println("Finished.");}
}
}
```

The following is the output of the program based on a different `buildings.txt` files.

File Contents	*Program Output*

`45 900 422 193`	`45 is legal`
	`900 is restricted`
	`422 is restricted`
	`193 is legal`
	`Finished.`

`45 900 -343 0 422 193`	`45 is legal`
	`900 is restricted`
	`-343 is invalid.`
	`Finished.`

`45 900 0 -343 422 193`	`45 is legal`
	`900 is restricted`
	`Zero is invalid.`
	`Finished.`

Fill in the missing portions of code. You can assume relevant packages are imported.

1) new Scanner (f) new Scanner (new FIS(b.txt));	5) zeroIntExp. e
2) ! f .exists()	6) "Zero is invalid"
3) ! f.canRead()	7) invalidIntExp e
4) s.hasNextInt()	8) temp+" is invalid"

Are ZeroIntException and InvalidIntException checked or unchecked assumptions? If you want to change them, how would you do that?

* unchecked

* make them extend Exception instead of RunTimeException.