Name (print): ____________________________________________________________

• **INSTRUCTIONS:**
  
  – Keep your eyes on your own paper and do your best to prevent anyone else from seeing your work.
  
  – Do NOT communicate with anyone other than the professor/proctor for ANY reason in ANY language in ANY manner.
  
  – This exam is closed notes, closed books, and no calculator.
  
  – Turn all mobile devices off and put them away now. You cannot have them on your desk.
  
  – Write neatly and clearly indicate your answers. What I cannot read, I will assume to be incorrect.
  
  – Stop writing when told to do so at the end of the exam. I will take 5 points off your exam if I have to tell you multiple times.
  
  – Academic misconduct will not be tolerated. Suspected academic misconduct will be immediately referred to the Emory Honor Council. Penalties for misconduct will be a zero on this exam, an F grade in the course, and/or other disciplinary action that may be applied by the Emory Honor Council.

• **TIME:** This exam has 5 questions on 6 pages including the title page. Please check to make sure all pages are included. You will have 50 minutes to complete this exam.

> *I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Emory community. I have also read and understand the requirements and policies outlined above.*

Signature: ____________________________________________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>10</td>
<td>10</td>
<td>6</td>
<td>10</td>
<td>14</td>
<td>50</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Consider the function below which is part of a Java program. Assume that sNums is always a sorted array.

```java
public static int what(int[] sNums, int n){
    int begin = 0, end = sNums.length-1;
    while (begin <= end){
        int mid = begin + (end-begin) / 2;
        if(sNums[mid] == n){
            return mid;
        } else if(sNums[mid] < n){
            begin = mid + 1;
        } else {
            end = mid - 1;
        }
    }
    return -1;
}
```

Answers the following questions.

(a) (1 point) What type is the return type of the function `what`?

(b) (1 point) How many local variables does the `what` method have and what is/are their names?

(c) (1 point) How many parameter variables does the function `what` have and what is/are their names?

(d) (4 points) What will this method return for the following calls?

- `what({5,9,12,30,76,100,125}, 100)` -->
- `what({5,9,12,30,76,100,125}, 8)` -->
- `what({5,9,12,30,76,100,125}, 12)` -->

(e) (3 points) What is this method accomplishing?
2. (10 points) Write the output that the following code will print. You may assume the code compiles as written.

```java
public class ExamQuestion{
    public static int k = 1;
    public static int[] subtractK(int[] nums, int k){
        System.out.println(k);
        int[] result = nums;
        for(int i=0;i<nums.length;i++)
            result[i] = nums[i] - k;
        return result;
    }

    public static void addK(int[] nums, int k){
        System.out.println(k);
        for(int i=0;i<nums.length;i++)
            nums[i] = nums[i] + k;
    }

    public static void main(String[] args){
        System.out.println(k);
        int k = 2;
        int[] arr1 = {1,2,3,4,5};
        System.out.println(Arrays.toString(arr1));
        addK(arr1,k);
        System.out.println(Arrays.toString(arr1));
        System.out.println(k);
        int[] arr2 = subtractK(arr1 , ExamQuestion.k);
        System.out.println(Arrays.toString(arr1));
        System.out.println(Arrays.toString(arr2));
    }
}
```
3. (6 points) For each entry below, state whether the code is correct as written or has an error. Show where the error is (you can circle the incorrect code) and explain why it is an error.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error?</th>
<th>Why?</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static int foo(char c) {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>return c + 1;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String[] s = {&quot;a&quot;,&quot;b&quot;,&quot;c&quot;}; System.out.println(s[1].charAt(1));</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String s = &quot;book&quot;;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>char[] c = s.toCharArray();</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System.out.println(c[s.length()]);</td>
<td></td>
<td></td>
</tr>
<tr>
<td>int[] a;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a[0] = 10;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>public static boolean foo(int a) {</td>
<td></td>
<td></td>
</tr>
<tr>
<td>if( a &gt; 10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>return true;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Consider the following methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>What?</th>
</tr>
</thead>
</table>
| public static int method1(int[] nums, int n){
  int k = -1;
  for(int i=0;i<nums.length;i++){
    if(nums[i] == n){
      k = i;
      break;
    }
  }
  return k;
} |
| public static int method2(int[] nums, int n){
  for(int i=0;i<nums.length;i++){
    if(i==n)
      return nums[i];
  }
  return -1;
} |
| public static int method3(int[] nums, int n){
  if(n < nums.length && n>=0)
    return nums[n];
  else
    return -1;
} |
| public static int method4(int[] nums, int n){
  int i = 0;
  while(i<nums.length && nums[i]!=n){
    i++;
  }
  if(i < nums.length)
    return i;
  return -1;
} |

(a) (8 points) What is each method accomplishing? (Fill in the table)
(b) (2 points) Which methods are doing the same thing?
5. (14 points) Write a function $\text{mixArrays}$. The function takes two arrays of integer as parameters. It should return an array of integers made of the first element of the first array and the first element of the second array, followed by the second element of the first array and the second element of the second array, and so on. Input arrays can have different lengths. Any leftover elements go at the end of the result array.

$\text{mixArrays}({1,1,1},{2,2,2})$ returns $\{1,2,1,2,1,2\}$
$\text{mixArrays}({1,2,3},{4,4,4,5,6})$ returns $\{1,2,4,2,4,3,4,5,6\}$
$\text{mixArrays}({1,1,1,1},{2,3})$ returns $\{1,2,1,3,1,1,1\}$