

1. (15 %) Identify 5 differences in syntax between python and *Java*. Describe each with at least one sentence and provide an example which shows both the *Java* and python versions.

2. (5 %) Why is it not necessary to indent your code in *Java*?

3. (10 %) Consider the following program and its output:

```
public class SizeProgram
{
    public static void main()
    {
        System.out.println("char:\t" + Character.SIZE / 8);
        System.out.println("int:\t" + Integer.SIZE / 8);
        System.out.println("double:\t" + Double.SIZE / 8);
        System.out.println("short:\t" + Short.SIZE / 8);
        System.out.println("Object:\t" + 4);
    }
}
```

Output:

```
char:    1
int:     4
double:  8
short:   2
Object:  4
```

Calculate how much memory (in bytes) is created on the stack for each of the following declarations:

Statement	Number of Bytes
a. <code>int i = 5;</code>	
b. <code>char[] str = new char[90];</code>	
c. <code>Double dPtr = null;</code>	
d. <code>double[] numbers = { 3.1, 4.15, 9.2 };</code>	
e. <code>short s = 65000;</code>	
f. <code>Short sPtr = new Short();</code>	

4. (15 %)

a. Draw a table of all the variables and how they change throughout the following program:

```

public class QuestionFour
{
    public static void main()
    {
        int i;
        int j;
        int result = 0;
        j = 5;

        for (i = 0; i < 10; i++)
        {
            result += i++ * ++j;
        }

        System.out.println("i is " + i);
        System.out.println("j is " + j);
        System.out.println("result is " + result);
    }
}

```

i	j	result
?	?	0
	5	

b. What is the output?

5. (15 %) Complete the following method.

```
public class QuestionFive
{
    /*
    * Method: insert
    * Purpose: This function inserts an element into an
    *           array of integers.
    *
    * Parameters:
    *   array - the array of integers
    *   size - the number of elements currently in the array
    *   elem - the element to insert into the array
    *   i - the index of the element to insert the new element
    *        before in the array.
    *
    * Precondition:
    *   'size' is less than the maximum size of the array.
    *
    * Postcondition:
    *   The 'array' should be unchanged from 0 to i - 1 and all
    *   of the elements from i upward should be shifted to the
    *   right. Element i should now be 'elem'.
    */
}
```

6. (10 %) In object-oriented programming, we introduced the concept of *mutability*. Is the following structure in *Java* mutable or immutable? Why?

```
public class Location
{
    public int longitude;
    public int latitude;
};
```

Bonus Question (2 %): How could you change its mutability (e.g., from immutable to mutable or vice versa)?

7. (10 %) What are the two main components that make up an abstract data type (ADT)?

8. (20 %) Consider the following program. Draw a diagram of memory for when the program reaches the marked point in the code (including the stack, heap, and all global variables). Make sure each variable on the stack and in the global variable space is labeled and that the value of each variable is specified (when known). You may use curly braces ({}) to name a group of variables, arrows (→) to represent references, and question marks (?) to represent uninitialized data. Also label the part of memory associated with each function.

Point.java:

```
public class Point
{
    public int x;
    public int y;

    public Point() {}

    public Point(int x, int y)
    {
        this.x = x;
        this.y = y;
    }
}
```

QuestionEight.java:

```
public class QuestionEight
{
    public static final Point g = new Point( 5, 3 );

    public static Point addPoints(Point p1, Point p2)
    {
        Point result = new Point();
        result.x = p1.x + p2.x;
        result.y = p1.y + p2.y;

        return result;
    }

    public static void main(String[] args)
    {
        Point a;
        Point b;

        a = new Point();
        a.x = 4;
        a.y = 5;

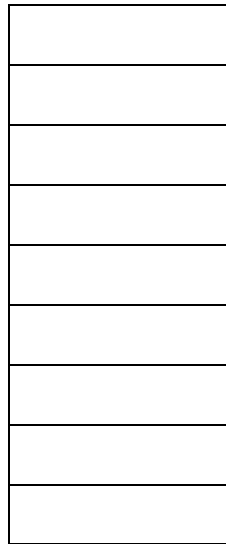
        b = add_points( g, a );
    }
}
```



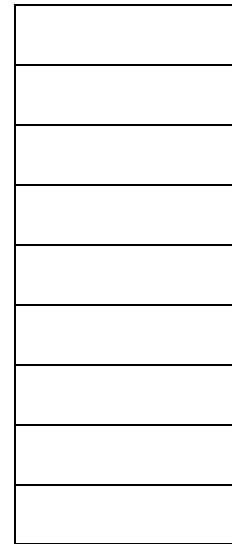
Here



Stack



Heap



Global Variables