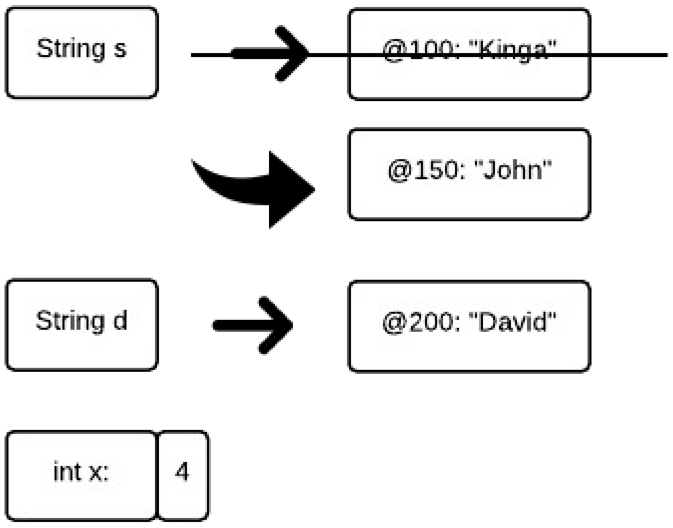
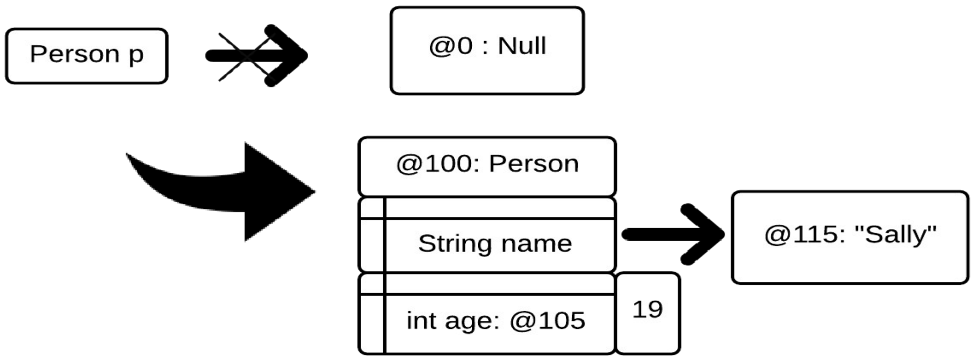
1.

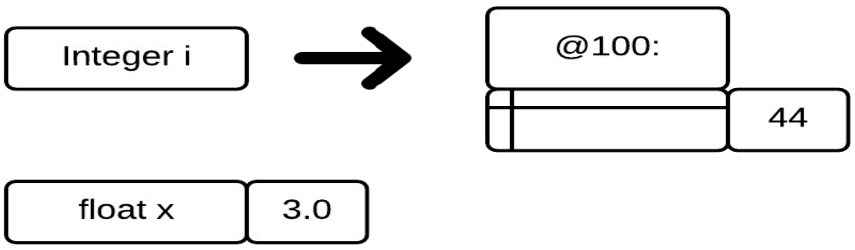
a)


String s
~~@100: "Kinga"~~
@150: "John"
String d
@200: "David"
int x:
4

b)


Person p
@0 : Null
@100: Person
String name
@115: "Sally"
int age: @105
19

c)


Integer i
@100:
44
float x
3.0

2. Results of expressions:

Part	Expression	Type	Value
a	`1 == 2;`	boolean	false
b	`3 / ((float 2);`	float	1.5

c	`x = false;`
	`“3” + x;`
		String	“3false”

d	`x = 4;`
	`x++;`
	`x / 4 + 3;`
		int	4

e	`true && p1==p1`	boolean	true

f	`String s1 = “ki”;`
	`s1.charAt[1]`
		char	i

g	`“1” + 3;`	String	13

3. I managed to get this code to work on Java Visualizer. Use it to guide your thinking.

   Click here

   Output:

```
2
John
John 11
John 11
John 11
n 0
false
true
2
John
Bob 3
Bob 3
Liz 10
John 0
```

4. Sample Code:

```java
// Part g.
public class Animal{
        // Part a fulfilled below
        private String species; // Private is optional, but good practice.
        private int weight;
        public Animal(){ // Part b
                this.species = "Arbitrary Animal";
                this.weight = 0; // Arbitrary weight;
        }

        public Animal(String species, int weight){ // Part c
                int len = species.length();
                if(len >= && len < 20){ // Part f
                        this.species = species; // Whatever you like.
                        this.weight = weight;
                } else { // Of course, what if they fail Part f?
                        /*
                         * Specs don't say what to do.
                         * In this case, you choose.
                         * Since I won't probably won't see this class again
                         * outside of this class... I'll just throw an
                          * exception.
                         */

                        throw new Exception("Species length failed :" + len);
                }
        }

        public void setSpecies(String species){ // Part e
                int len = species.length(); // Strings have a... size method?
                if(len >= 1 && len < 20){ // Part f
                        this.species = species;
                } else { // In case the species name wasn't good...
                        System.out.println("Species length failed :" + len);
                }
                /*
                 * You know when you've typed a word too many times...
                 * It doesn't look like the word anymore?
                 * "species" looks like such a weird word.
                 */
        }

        public String toString(){ // Part d. Anyway you like it.
                return "Species: "+this.species+"\nWeight: "+this.weight;
        }
}
```

5.

Question	Answer and Justification
A public static method of Person can be called on p1	True. Though discouraged, you can do it.

A public static method of Person can be called on Person (the class) in main	True. This is allowed.
A static method can update both static and non-static attributes of the same class	False. A Static method can only update static attributes of the same class.
A public static method can call other public static methods within its body, but not private static ones	Ambiguous. A public static method can call other public static methods and private static methods within the same class.
A private method can only access private attributes within the same class	False. Private methods have access to all attributes in the same class.
The main method can call a private method of Person on p2	False. Private methods can only called within the classes they were written in.
The equals method can be called on p1 with p2 as an argument	True. This is appropriate.
I must import java.util.Scanner before using a Scanner object	True. To use objects not included in java.lang, you need to import it.