Name: Yusuf J.

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

change "<" to ">" so it can loop Throw the array list and change things. As it is now the list doesn't loop.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

The name of the arrayList is newRabbits not rabbit So the rabbits ~~aren't~~ are not actually being removed. change rabbit to newRabbits to remove them.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

students = new ArrayList<Student>();
Pair match = new Pair(null, null);

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

~~Pair~~ Student

d. Would it be valid to say: `line.add("hi");` ? Explain.

~~Yes~~ because add has been used before to add a line and line can save Strings into it and "hi" is a string.

e. Name a method that you know the object in the **line** variable can run. br.readLine();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List(Wombat) wombatList = new ArrayList(Wombat)();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of a constructor is to call upon the class about the thing its making, and then make it. Make sure they data type is correct and that the class is correct.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

It means that the Fox class takes elements from the Animal class in order to complete it. the Fox is the child class while the Animal class is the parent where Fox stems from.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

They gave all the animals the same breeding age, max age etc

randomAge was never established.

foodLevel shouldn't have the rabbit food value set into it.

Name is never established.

Name: Maya J

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

The for-loop should loop through all of the locations instead of just continuing until i equals the size of the array. The i< part should change from newRabbits.size() to the total amount of locations

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

It should be newRabbits.remove(i). The rabbit has to be removed from the array

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

BufferedReader
Pair

b. On line 21, the variable is named br and the method is named close()

c. On line 17 we see *students*. What data type is it?

List<Student>

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

yes line is a string

e. Name a method that you know the object in the **line** variable can run.

trim

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The constructor sets up the code by defining the basic characteristics that something has.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

It means that the code for Fox includes all of the code in Animal, but with more added in the class. In other words, Fox is an addon with more specific details not put into Animal.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

age and alive already declared in Animal

no var in super, should have string

Name: HANNAH WONG 3/25/19

7°

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?) you need to add a rabbit. The code is not adding any, so newRabbits.add(Rabbit); is needed.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

It is not supposed to be rabbit.remove(i);. It is supposed to be newRabbits.remove(i) because the list is called newRabbits.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

line.trim();
br.readLine();

b. On line 21, the variable is named br and the method is named .close().

c. On line 17 we see *students*. What data type is it? ArrayList<Student>

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

d. Would it be valid to say: line.add("hi"); ? Explain.

No, because line is not an array.

e. Name a method that you know the object in the **line** variable can run. line.trim();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

st<Wombat>wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does? The purpose of a constructor is to make the object do something without needing an input. The key thing you need the code in the constructor to do is the method.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

It means that fox can get the codes from animal. The code in FOX is extra code that completes the code from animal.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

It should be protected int/boolean for age and alive. Alive should not equal false. It is not boulean randomage. There should be something with randomAge in the for loop. The else in the fox should not equal the rabbit food value.

Name: Meera Jagota

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

newRabbits.size() is the size of the arrayList, which starts out empty, because no rabbits have been added yet. It should be replaced with a constant variable like int MAX_POPULATION, which would be declared before.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

A rabbit needs to be removed from the arrayList It should look like newRabbits.remove(i);

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

Pair
BufferedReader

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it? array list

```
 9   public static void main(String[] args) {
10     students = new ArrayList<Student>();
11
12     BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13     String line = br.readLine();
14
15     while (line != null) {
16       line = line.trim();
17       students.add(new Student(line));
18       line = br.readLine();
19     }
20
21     br.close();
22
23     Pair match = new Pair(null, null);
24
25     // perform matching
26     while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

No, because line is not an array list, but a string

e. Name a method that you know the object in the **line** variable can run. trim

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

A constructor should have code which runs when a new object has been made.
The constructor should state what type of inputs it needs.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

the child Fox class can access code in the parent Animal class. There will be some code in the Fox class which has gaps that will be filled in by code from the Animal class.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

private int age doesn't have to be in the Fox class
private boolean alive doesn't have to be in the Fox class
alive should = true
the name variable isn't declared
super() should have an input for the name

Name: Zach Oren 7

QUon Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

The for-loop should cycle through each rabbit object in the rabbits list, but will end at whatever the last rabbit number is in the newRabbits list instead of the last rabbit in the rabbits list. To fix it replace newRabbits.size() with rabbits.size().

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

The problem is the rabbit itself is not an arraylist and cannot execute the remove() command by itself. It should be fixed by writing rabbits.remove(i) instead, because rabbits is an ArrayList and can run the remove() command.

3. Consider this scary-looking code that you've never seen before!

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

a. There are three constructors getting run in this code. Name two of them.

new BufferedReader(...) (line 12)

new Pair(null,null) (line 23)

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

Array list of Student objects

d. Would it be valid to say: `line.add("hi");` ? Explain.

no, because line is not an arraylist and cannot run the add() command.

e. Name a method that you know the object in the **line** variable can run. trim()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

```
wombatList = new ArrayList<Wombat>();
```

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of a constructor is to create a new object and initialize values for its variables. It is key for the code in a constructor to make new copies of all the variables used by the object.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

If Fox extends Animal, then Animal is a superclass and the Fox class is a subclass, meaning it uses the variables and methods in the Animal class. The code in Fox sets values for each variable, and adds methods and additional variables unique to the behavior of a Fox itself.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1) Not all animals will have the same Breeding age, max age, etc, so it should not be given values in Animal.
2) age and alive must be protected in animal, and not private
3) all animals start as dead when the constructor is run in Animal
4) The fox class initializes the age and alive variables even though they are part of the parent class
5) The super(); method must take the input (String n)

Name: Ray Dayan B#7

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

i = 2 → 1
i = 1

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

The loop creates new rabbits, yet it doesn't add them inside into the rabbit list. At the end of the loop, there should be a "rabbits.addEach(newRabbits);". and outside of it
In addition when a rabbit is dead, i is substracted from i, although it should run the loop for one object it already ran on, and the i++ in the for loop will cancel it out.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

The problem is that rabbit is a rabbit object and the method remove is associated with lists. Therefore, an "s" should be added to the object's name to make it "rabbits" which is a list.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

Pair(); Student();
ArrayList<>();

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
```

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

Array list

d. Would it be valid to say: `line.add("hi");` ? Explain.

No, because line is a String, therefore nothing can be added to it unless it is changed to be a list.

e. Name a method that you know the object in the **line** variable can run.

readLine();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of a constructor is to create a new instance of an object, the code in the constructor needs to set up the object so the methods inside it are able to function with basic values/variables.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

Fox is able to use the code inside Animal for its own purposes and Fox is an instance of an Animal object Fox can access variables and methods that are declared within Animal.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. super needs to have an input which defines the animal's name.
2. age and alive are declared in both Animal and Fox classes.
3. The values set within the Animal class would be the same for all animals.
4. foodLevel should be static because it is different for each instance of Fox.
5. If location is a variable that all animals share it should be declared inside the Animal class.

Name: Kayra N.

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

They're getting the size of newRabbits in the for loop, however, newRabbits doesn't have any rabbit objects because we haven't added any. you change newRabbits to a previous list of rabbits, that has rabbit objects in it.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

The remove method is the problem. You need to save rabbit.remove(i) into another variable.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

students, match

```
 9   public static void main(String[] args) {
10     students = new ArrayList<Student>();
11
12     BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13     String line = br.readLine();
14
15     while (line != null) {
16       line = line.trim();
17       students.add(new Student(line));
18       line = br.readLine();
19     }
20
21     br.close();
22
23     Pair match = new Pair(null, null);
24
25     // perform matching
```

b. On line 21, the variable is named br and the method is named close();

c. On line 17 we see *students*. What data type is it?

Arraylist

d. Would it be valid to say: `line.add("hi");` ? Explain.

no, line is a String, not a list, therefore you can't add values into line, you can only save values into line.

e. Name a method that you know the object in the **line** variable can run.

readLine();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does? A constructor creates/defines? a new object, arraylist You want to make sure that the code in the constructor creates a new item(list, object) and initializes variables that go along with those.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

Fox is a child class of the parent class Animal. It means that Animal is a template class for Fox, and Fox contains all of Animal's code plus it's own.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. The breeding age, max age, breeding probability and max litter size are all given values in Animal

2. age and alive are declared in Fox and Animal

3. The Animal constructor takes an argument n, but Fox doesn't give its super one.

4. age and alive are private instead of protected in Animal (this could be a personal preference)

5. age and animal are defined in Animal instead of Fox.

Name: Erik Johnson

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

We want to loop over ALL the rabbits to do the for loop. Instead of using "newRabbits", the list "rabbits" should be used.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

On Line 166, list "rabbit" is used. We want to use the list "rabbits".

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

"Students" and "match"

b. On line 21, the variable is named line and the method is named close.

c. On line 17 we see *students*. What data type is it? ArrayList

```
 9   public static void main(String[] args) {
10     students = new ArrayList<Student>();
11
12     BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13     String line = br.readLine();
14
15     while (line != null) {
16       line = line.trim();
17       students.add(new Student(line));
18       line = br.readLine();
19     }
20
21     br.close();
22
23     Pair match = new Pair(null, null);
24
25     // perform matching
26     while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

No. line is a String, but it is not a list of Strings. So, it is not valid.

e. Name a method that you know the object in the **line** variable can run. br.readLine()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

ArrayList<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

A constructor makes a new object, ie a list. Some key things in the constructor is the type of object, and the name of the object.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

Fox extends Animal means that Fox inherits all the variables and methods in Animal that Fox can access. This means that Fox's code is shorter and more concise, as well as new animals can be added with ease when using "extends"

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
1   alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
2 private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
    3 foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
    4 foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. if we set alive to false right away, all animals will die instantly !!.
2. a variable in caps can only be created in the parent class.
3. that variable in caps isn't created in Animal.
4. that variable in caps isn't created in Animal (just like 2).

Name: Akash Jain

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

b/c its initialized right above the loop. newRabbits has no length. You can change it to rabbits.size which is the actual arraylist you want the length of.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

Line 166 has an error because rabbit is not an arraylist. You can change "rabbit" to "rabbits" which is an arraylist.

3. Consider this scary-looking code that you've never seen before!

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
```

a. There are three constructors getting run in this code. Name two of them.

Students = new ArrayList<Student>();
Pair match = new Pair(null,null);

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

ArrayList<Student>()

d. Would it be valid to say: `line.add("hi");` ? Explain.

No because line is a string, not an Arraylist, so you can't add String values to something; only replace it.

e. Name a method that you know the object in the **line** variable can run.

trim()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombats = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

the constructor is where a variable or a set of variables are defined. You want to make sure the code in a constructor defines the value or ArrayList to that variable accordingly

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

It means Fox is a sub class of Animal, the super class. It inherits any characteristics of the Animal class which are common to all animals.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

[illegible] need to be final as they aren't changeable.

- The animal name should not be set in the super class.
- Age & alive are repeated in Fox & Animal
- [illegible] should be final and set in Animal class & Animal to all subclasses that eat rabbits.
- random Age in Fox class should not be boolean. It should be int.
- Alive in Animal should not be false, which means it is dead

Name: Chris Oh B7

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

newRabbits is the list that makes new rabbits to add to the list rabbits, while this for loop is meant to get the rabbits in the list rabbits to do stuff. It should loop i < rabbits.size() instead of i < newRabbits.size()

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

It's calling remove method from the rabbit object instead of the arraylist. It should be rabbits.remove(i);

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

10: students = new ArrayList<Student>(),
12: BufferedReader br = new Buffered...

b. On line 21, the variable is named br and the method is named close().

c. On line 17 we see *students*. What data type is it?

Student

```
 9   public static void main(String[] args) {
10     students = new ArrayList<Student>();
11
12     BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13     String line = br.readLine();
14
15     while (line != null) {
16       line = line.trim();
17       students.add(new Student(line));
18       line = br.readLine();
19     }
20
21     br.close();
22
23     Pair match = new Pair(null, null);
24
25     // perform matching
26     while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

no, b/c line is a String, not an arraylist

e. Name a method that you know the object in the **line** variable can run.

trim()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

It creates the object so that the code can interact with it and assigns variables values.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

Fox inherits characteristics of Animal, such as having the int variable age, and also methods. In code, Fox says extends Animal, has a super(x); in the constructor, and doesn't have to declare the variables or methods it inherits.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. Animal's constructor receives String n but Fox's constructor receives boolean RandomAge.
2. nothing went into super();
3. declares age variable in both classes
4. assigns value for age in both constructors and also in different ways (inconsistent)

Name: Nikhil Jagodar

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

the list that is getting the size should be the rabbits list. If it lists as many times as the NewRabbits size, then it wont last at all. What it is supposed to be doing is go through every existing rabbit, not the new rabbits.

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

What it should say is rabbits.remove(i). The problem is that you are asking the rabbit object to remove but you can only remove it from the list.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

BufferedReader

Pair

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

ArrayList<Student>

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

It would not be valid because the variable "line" does not have a method "." add"

e. Name a method that you know the object in the **line** variable can run.

trim()

I the the the th the the the the the the the the

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

ArrayList<Wombat> wombatList = new ArrayList<Wombat>;

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of a constructor is to make a new object and assign values to its variables. The key thing you want to make sure the code in a constructor does is assign the variables values.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

What this means is that the Fox class and all fox objects will have a copy of the Animal class, all the variables and methods, in the code.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. "Name" is never defined in the animal class.
2. Fox doesn't take an input "n" and doesn't give it to the "super"/Animal class.
3. "Age" and "alive" are being defined again in Fox even though they are already defined in Animal.
4. In the animal constructor, alive is being set to false when it should be set to true.

Name: Julie Li

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

The i < newRabbits.size(); is the mistake because the ArrayList is for newRabbits. I would fix it with Rabbit.size();

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

The (i) is the problem, you want to remove the rabbit's location than i instead.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

line
br

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

arrayList

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

No, you need to add an object to the ArrayList command.

e. Name a method that you know the object in the **line** variable can run.

br.readLine();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of a constructor is to assign values to the variables. One key thing to make sure the code does is assign values that the program will use.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

That means fox is an extension to the animal class. In terms of code, the fox inherits every code in the animal class.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

super(); needs to take in a variable, randomAge.

The if(randomAge) doesn't make sense for the if statement.

The private boolean alive doesn't make sense since the animal class has it already.

The name=n; because name hasn't been assigned a variable type like String in the code.

Name: George Yordanov

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?) The sign which states "i < newRabbits.size()" should be flipped to say "i > newRabbits.size()".

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).
Line166 stated to remove "i", but that is incorrect because we want to remove the rabbit so we need to add "rabbit" into the parenthesis.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.
1. Student
2 br

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it? Arraylist

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
      while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.
Yes, because "hi" is a string.

e. Name a method that you know the object in the **line** variable can run. line = line.trim();

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does? The purpose of a constructor is to make a storage for your objects. You want to make sure that the code in the constructor includes "new".

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)
When extends is implemented into Fox, this means that the Fox gets all the code from Animal added to itself

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. There is an "age" integer in both Animal and Fox when you can just have one "public int age"
2. There is "alive" boolean in both when there can just be a public one in Animal.
3. No need for "super();" in the fox code.
4. No need for _____, because we already created an age integer called age.

Name: Sreeganesh Sw. 725 B.7

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

Since you want it to loop through the Arraylist and Arraylists start with a size of 0, it will not do anything until you put objects (Rabbits) in the arraylist. To fix it you need to create and put rabbits into the list

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

Since rabbit is an object and not a list type you cannot use the remove method. you can change rabbit to newRabbits which is a list.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

BufferedReader and ( ) FileReader ( ) and Student()

b. On line 21, the variable is named br and the method is named close.

c. On line 17 we see *students*. What data type is it?

an arraylist of Student objects

```
 9   public static void main(String[] args) {
10     students = new ArrayList<Student>();
11
12     BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13     String line = br.readLine();
14
15     while (line != null) {
16       line = line.trim();
17       students.add(new Student(line));
18       line = br.readLine();
19     }
20
21     br.close();
22
23     Pair match = new Pair(null, null);
24
25     // perform matching
26     while (students.size() > 0) {
```

d. Would it be valid to say: line.add("hi"); ? Explain.

No because line is a String type and not a list type so you cannot use the add method.

e. Name a method that you know the object in the **line** variable can run. readLine()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

List<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The constructor is to define the variables needed for a particular object. You want to make sure that the ~~values~~ variables for the object are given values in the code for the constructor. and recieve inputs

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

It means that the Fox is a 'child' of the animal class and it inherits all the methods and variables from the Animal class. So the fox object has all the methods in Animal even though the code is not in the Fox class

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

1. the variables in animals have values so all classes that extend Animals have those values for those variables (they should be given values in Fox constructor
2. Alive = false so all animals are automatically dead
3. The variables are redundant with those in the animal class so they are not needed since Fox extends Animal already
4. When super is called it goes to the animal constructor which takes String n as an input which is not given when super is called

Name: Katya Teodorovich

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

It should be i < rabbits.size() - you're looping through all rabbits here you just make new list & loop through it, but it's empty

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

rabbit**s** is the name of the list

you want to remove this rabbit from the list, so you use the plural

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

line 10 → new ArrayList<Student>()

17 → new Student(line)

b. On line 21, the variable is named br and the method is named close().

c. On line 17 we see *students*. What data type is it?

ArrayList (of Students)

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
26    while (students.size() > 0) {
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

no. line is a String & Strings don't have a .add() method

e. Name a method that you know the object in the **line** variable can run. .trim()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

ArrayList<Wombat> wombatList = new ArrayList<Wombat>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

Constructor makes a new object of some class.
It sets all the values (all fields)

use it & save into variable

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

fox has everything animal has & you can add more

inherits everything from Animal class

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

- don't need age & alive in fox - already inherited from animal
- can move location to animal as well, all animals have one
- super needs a string (name) as param/ input
- animal's age and alive fields are private, should be protected because you want children to access them

Name: Aayush. M.

QU on Foxes & Rabbits.

CODE INSIDE THE SimulateOneStep method in SIMULATOR.JAVA

```
158
159     List<Rabbit> newRabbits = new ArrayList<Rabbit>();
160
161
162     for (int i = 0; i < newRabbits.size(); i++) {
163       Rabbit rabbit = rabbits.get(i);
164       rabbit.act(field, updatedField, newRabbits);
165       if (!rabbit.isAlive()) {
166         rabbit.remove(i);
167         i--;
168       }
169     }
```

1. The for-loop on line 162 has a mistake in it. What's the problem with it? How do you fix it? (2 sentences). (think of the purpose for the loop; is it doing that purpose?)

newRabbits.size() is not been declared anywhere. I would first declare the rabbits.size

2. Line 166 has a problem on it, what is it? How do you fix it? (2 sentences).

~~It should be outside the for loop~~

we don't have the method for remove. First make the method in animal class that describes what .remove will do.

3. Consider this scary-looking code that you've never seen before!

a. There are three constructors getting run in this code. Name two of them.

BufferedReader
Student line.

b. On line 21, the variable is named br and the method is named close

c. On line 17 we see *students*. What data type is it? It's an arraylist

```
9   public static void main(String[] args) {
10    students = new ArrayList<Student>();
11
12    BufferedReader br = new BufferedReader(new FileReader("16-17APCS.txt"));
13    String line = br.readLine();
14
15    while (line != null) {
16      line = line.trim();
17      students.add(new Student(line));
18      line = br.readLine();
19    }
20
21    br.close();
22
23    Pair match = new Pair(null, null);
24
25    // perform matching
```

d. Would it be valid to say: `line.add("hi");` ? Explain.

~~not a valid list~~ Yes, it would be valid because line is a ~~data~~ string and adding "hi" would mean adding a string.

e. Name a method that you know the object in the **line** variable can run.

readline()

4. Show how to declare a variable called wombatList which holds and ArrayList of Wombat objects. Be sure your code actually creates a new list and saves it into the variable.

~~ArrayList<Wombat> new ArrayList Wombat list~~

ArrayList<Wombat> wombatList = new ArrayList<>();

5. In 2 sentences. What's the purpose of a constructor? What key thing do you want to make sure the code in a constructor does?

The purpose of constructor is to ~~set initial values~~ initialize objects. meanny, to set initial values to the attributes ~~states~~ of the objects. Code in a constructor should not have a return type and ~~it should~~ the name of correct constructor must match the name of class.

6. What does it mean to say that Fox *extends* Animal? What does it mean in terms of the code in Fox? (2 sentences)

Fox extends Animal means that the fox is a type of animal meaning, fox shares or used the code in animals to run itself.

7. Consider this code from an Animal class and a Fox class that extends Animal. There are at least 4 things that don't make sense in this code. Tell me in 1 short sentence each what they are. Please don't elaborate a lot.

```
public abstract class Animal {
  protected int BREEDING_AGE = 3;
  protected int MAX_AGE = 50;
  protected double BREEDING_PROBABILITY = 0.15;
  protected int MAX_LITTER_SIZE = 6;
  private int age;
  private boolean alive;

  public Animal(String n) {
    name = n;
    age = 0;
    alive = false;
  }
```

```
public class Fox extends Animal implements Serializable {
  private int age;
  private boolean alive;
  private Location location;
  private int foodLevel;
  private int RABBIT_FOOD_VALUE = 6;

  public Fox(boolean randomAge) {
    super();

    if (randomAge) {
      age = (int)(Math.random()*MAX_AGE);
      foodLevel = (int)(Math.random()*RABBIT_FOOD_VALUE);
    } else {
      foodLevel = RABBIT_FOOD_VALUE;
    }
```

- private int, private boolean should be protected int; protected boolean. (in Animal class).
- (Fox) ~~this~~ this. should be used.
- (Animal) public Animal (String n) should have a return type.
- protected int food level is missing in animal class.