CS 46A Lab 1

Lab Objectives:

- Practice basic programming problems and become more familiar with Java including:
  - How to write a basic Java program
  - How to use basic Java Graphics
- Become familiar with BlueJ including:
  - How classes interact with each other and how compilation works in BlueJ
  - Usage of autocomplete feature

Getting Started

1. You will work in pairs. Or a group of 3 if there are an odd number of students.
2. Create a google document or other shared document with the title "Lab01 - Name1, Name2. Both partners must submit the same document.
3. Follow instructions in this lab document to populate the shared document. You will submit this document for lab credit.
4. Stuck? Ask your partner, and if you are both stuck, ask for help!

Preparing Your Lab Writeup

For the report, you can either create a copy of this lab document with your answers filled in highlighted in a green color so we can easily see it, or in its own document with the questions. Make sure to include the answers to the offline questions as well.

Within the document, whenever you see a prompt, it should look bolded like this:
What happened at this step?

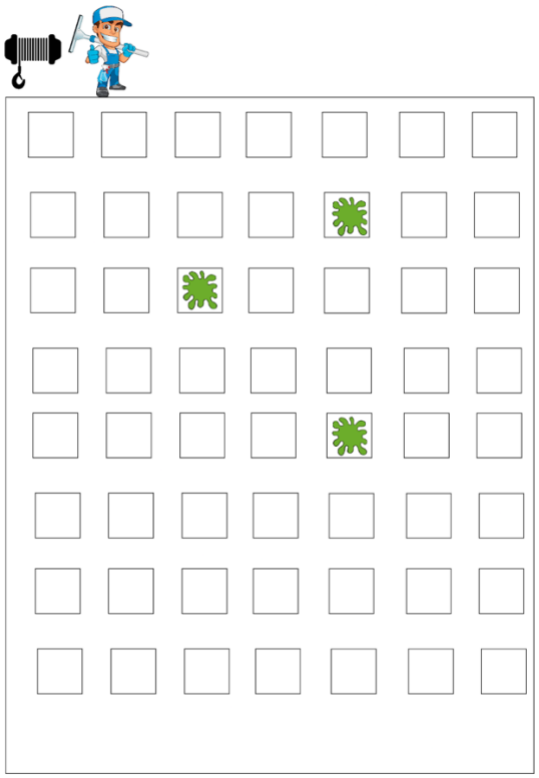
Fill in the section below the prompt as follows:
Example:
At this step, we tried to run the code and it didn't work.

If you are expected to submit screenshots, put those at the end of your report.

Offline questions:

With your partner, each of you choose one of the following questions to answer (two total for partners, three for groups of three). Reason through each one and discuss your answers. Then, create a document where you will put your answers with the label 'Problem x: *your answer here* where 'x' is the problem number.

1. Bob is a window cleaner. He has a device installed on the top left corner of the building that he works at that can winch him up and down a building. Using the image below, and assuming that one square is one meter, how many meters does he need to move the winch to the right, and how much cordage does he need to lower to lower his platform so he is about where the dirty windows are?



2. Overloading is a very useful tool that exists in Java and in many other programming languages. Essentially, you can give different methods the same name. You can do this with constructors, or mostly anything that can receive input!

   Take the following list of methods:

```
Student()
{
    // Creates a student object with all fields blank
}

Student(String name)
{
    // Creates a student object with a set name
}

Student(String name, Array[])
{

}

power(int x)
{
    // returns x^2
}

power(int x, int y)
{
    // returns x^y
}

greet(String name)
{
    // System.out.println("Hello, " + name + "!");
}
```

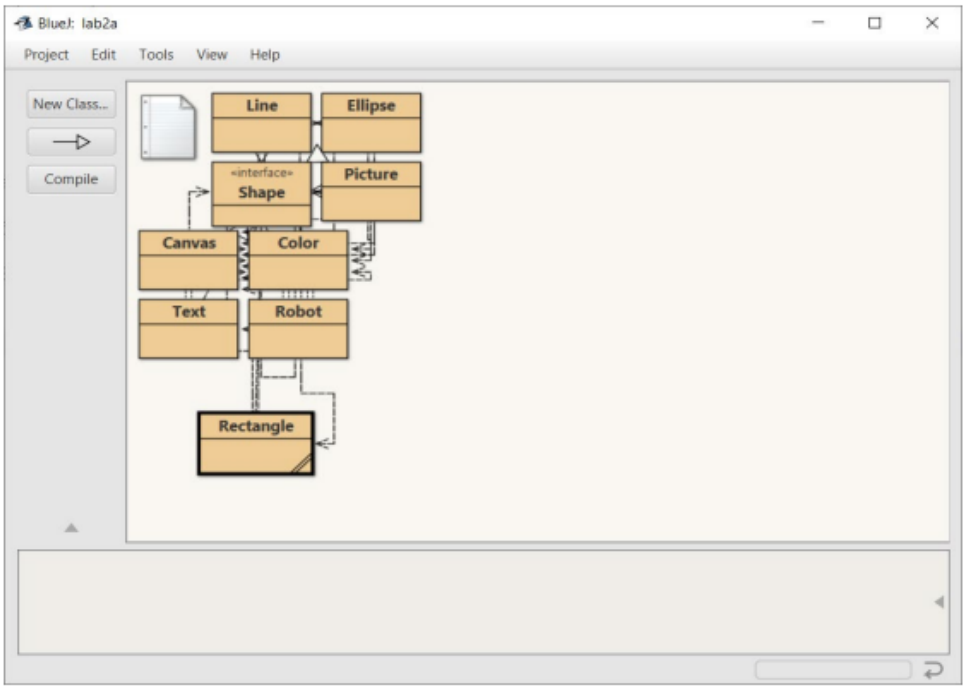
a. What are your options if you want to find the value of a number squared? Which would you choose if you wanted the method to say something to the user?
b. What 'Student' would you call if you want to create a student when you don't have any input?
c. What 'Student' would you call if you had a whole array of data to input into the student object?
d. Why do you think this is a capability that Java has? What uses can you imagine for it?

Lab01 - Basics of Graphic Design

Graphic Design is a field generally perceived as 'related but different' from computer science. While advanced topics like 3d design and CAD modeling are a different field, there exist other subfields in CS, like frontend design, where you will find it important to know a few ins and outs of how images get lined up in a window.

Complete the following steps:

1. Download the graphics package from Canvas (see link in Module 1) and unzip it to your cs46a/labs folder. The folder should have 9 Java files and three png files.
2. Create a new folder lab01 in your cs46a/labs folder if you have not done it yet.
3. Launch BlueJ and make a new project lab1a in your lab01 folder. Import the graphics package to your BlueJ lab1a project.
    1. Select "Project", then select "Import"
    2. Navigate to the graphics directory in your cs46a/labs folder
    3. Select the folder and confirm
4. Hit Compile to compile all graphics classes.
5. You should have a window that looks similar to this:



6. Create a new class named BasicGraphicViewer
    1. Modify the description to say 'calls various methods from the graphics package to create a collage'
    2. Select an image from those provided with the graphics package or download one from the internet.
    3. Delete all of the code in the class and add the main method as below:

```
public class BasicGraphicViewer
{
     public static void main (String[] args)
     {
          Rectangle r1 = new Rectangle(100,10,20,30);
          r1.draw();
          Picture p1 = new Picture();
          p1.load("image.jpeg"); // If downloading a new image,
                                 // save the image
                                 // in the cs46a/labs/lab01/lab1a
                                 // (lab1a folder)
          p1.draw();
     }
}
```

    Replace the "image.jpeg" text with your image.
    What image did you use?

7. Before you run the code, what do you think will happen?
    Write down your expectations before you run the code.

    Compile and run the code:
    Note any differences between expectations and reality.

8. From here, you will add code of your own.
    Write code so that you make another image <u>below</u> the image currently in the code so that they are next to each other but not touching each other.
    Put a copy of the code you added into your report.

9. Try out some of the methods that are available to the Rectangle and Picture class. You are not required to do this for completion of the lab itself, but you may be asked questions on how the classes function.
    a. Translate - A function that moves the graphic according to the given input
    b. Grow - a function that increases the size of the graphic according to the given input

10. If there is more time in the lab, come back and look at the methods also available to the Picture class.

Controlling a Robot and BlueJ tricks

1. Create a new folder called lab1b within cs46a/labs/lab1 folder and create a new BlueJ project called lab1b. Follow the instructions for the previous section to import the graphics.

2. Create a new class called RobotDemo

3. Modify all descriptive values to your preference (@author, @version (date) and description).

4. Delete all of the code inside, create a main method (look at your previous code for an example) and create a new 'robot' using the method below:

```
Robot robbie = new Robot();
```

This will construct a Robot object with the image robot.png. Look in the constructor.
Do you think the robot will appear without additional method calls?
Write the answer in your report.

5. Execute the code, see the robot! He's alive! He can't do anything yet though. So let's truly make our robot move!

6. But let's do it with a twist. On the line after you create the robot, type 'robbie' and a period. Then hit Ctrl+Space. On the Mac, really use the Control key NOT Command.
   What happens? Describe what appears.

7. This feature is called autocompletion. It is extremely useful because it means you don't have to remember the exact names of all methods. Find a method that makes the Robot move and select it. You still need to add the semi-colon.
   What is the method called?

8. Now add another line:

```
String name = "Robbie Robot";
```

   Just like the previous code, type name then a period, and hit Ctrl+Space. After you see the dropdown menu, type the letter 't'. What happened to the menu? Do you think this technique could save you time?

9. Now, let's make the robot turn and move around as we wish. Pick a number between 2-9, try to make the robot trace the steps to draw said number! (The robot won't leave a permanent trail so watch it take its path).

10. Say you don't like the look of robbie (how could you?). You can create a robot with a different image and even location. Try the following constructor instead.

```
Robot roomba = new Robot(0, 0, "roomba.png");
```

11. If you have more time at the end of the lab, look at the other methods that exist within Robot.

12. Of the following, what methods DON'T exist within the Robot class? Record the letters in your lab document.
    a. `move()`
    b. `moveForward()`
    c. `moveLeft()`
    d. `turnAround()`
    e. `turnRight()`
    f. `turnLeft()`
    g. `rightHasWall()`
    h. `leftHasWall()`

Before you Leave:

At the end of the report, put the following:

1. A screenshot of the output from the first part of the lab (the two pictures)
2. A screenshot of the robot after it traced the number.
3. Check in with you Lab Instructor.

Submit your lab document and make sure that you recorded all lab prompts. Additionally, make sure you have all screenshots required for your submission.