Report

-----assignment 1-----

- Yixi rao (kyrie)
- Fri15 L124_left 15:00-17:00

  Alex Smith
- U6826541

Report

Introduction:

1. My program is based on the codeworld api, it has three parts (helper function, render picture, event) to help me deal with the problems through clicking some points at the coordinates and create a colourful shape on the screen and some key presses that can be used for special intention such as change colour.

Content:

For the part 1 functions (helper function), I designed to solve some problems that will happen in the part 3(handle event), which include functions changing colours and tools, which I wrote it in a case statement and used the prepared order.

For the part 2 functions (render colour), the first function is colourNameToColour, which I finished it by using the case statement to change colour into codeworld colour according to a prepared order. Secondly, shapeToPicture, I used the case statement to separate different patterns, besides polygon have different situation such as we just have two points or one point or 0, learning form math, I perceived at least three points to form a polygon, so I use three patterns to solve it, than I typed some math formula to draw

the rectangle,circle,ellipse, and I use where statement to write some expressions ,because it will make the code more distinct and understandable, which includes the radius expression and scale expression on x and y by mathematic method. And the function colourShapeToPicture colour the picture by using the precious function shapeToPicture to turn it into a picture that will make it easier to colour it and used the "coloured" function in the codeworld function to render picture. colourShapesToPictur, which takes a list of shape and renders all, combines it as well. I used recursion on a list (use colourShapeToPicture to the first elements on the list and the second one…), the base case is a blank, and I use "&" to combine each of it.

For the part 3, at the **pointpress and pointrelease,** first, as the pointpress, I use the helper function to add the point p to the tool, for polygon, it just added in to list. The function will help me solve the different cases. I use tool as input of the helper function, which helps me to avoid tremendous amount of trouble such as not enough variables to use. Secondly, I use case statement under the case statement of the handlevent, double case statement help me to use more variables as I typed it under the second case, so it can add the first point to the ultimate list. Finally, changing the colours and the tools were done by using the previous functions that we have already defined in part 1, than, as the "**backspace**" and "**space key**" I use a

helper function and case statement to finish it, backspace just use "_: xs " than take xs omitting the first, for the space key, I wrote a helper function that has tool as input and transform only polygon to model.

Assumptions

1.I think the major problem that the user will come across when try to use the program is the point that they click and release. So, I make the assumptions that the points they click are "Coordinates of opposite corners of the rectangle" and "Centre point, then point on the circumference of the circle" and "Coordinates of the bounding box for the ellipse". So, when I designed the code, I must satisfy the conditions above.

Testing

1. On the part 1 functions I just used the cabal test to test it and I also loaded it to try some input to see whether it satisfy the output that I expected.
2. On the part 2 especially on the **shapeToPicture** I wrote some points on a paper and drew the picture in advance to collate it after I received the output by using the function of drawingOf. Besides, I checked every picture by generating it on different coordinates included negative and positive and both coordinates, I also checked the conditions of the points which I assumed in Assumptions above such as bounding box for the ellipse to see whether they satisfy what I expected or not.
3. On the part 3, it is convenient to use cabal run to run the program, I

used click-drag-release to test of pointpress and point release at the same time, I pressed 'D' to track it every steps, and used my picture that I had already draw on the paper to collate it, I checked the key presses by pressing some key to see what happen especially in the mystery picture, I used backspace to see the cat picture what happened . And I also tried to draw some pictures in a different direction, right to left or left to right especially the polygon.

4. Other tests that I done were the edge cases, on the part 2 I tried to render a picture but it is an empty list(**colourShapesToPictur)** as input. and I tried to just give one or two or 0 points to the polygon shape to see what happen

Reflection

1.when I commenced to do the part 3, at the **pointpress** and **point release**, at the beginning, I used guard under the "pointrelease p", but I noticed that there was no ways to have another variables, since we just have one variable p, the final point, besides, this situation also happened when I wrote the pointpress and other places we must use two variables. I overcame this problems by using another case statement to have another variable that under the "case t of" such as "LineTool (Just q)-> Model [(c,Line p q)] (LineTool Nothing) c", which perfectly tackles this problem, but another problem exposed, which is Pattern match(es) are non-exhaustive, I perceived that t represents tool but I do not want polygontool appeared in my pointrelease because I just want to create a polygon through using pointpress to click some points. The problem was solved when I browsed all the codes

in the “Controller”, I found “_ -> m”, which simply returns the model without changing anything, hence, I added it as the last pattern of the pointrelease.