Technical Report
COMP1100 Assignment 1

Jacob Bos
ANU u7469354

April 4, 2022

Lab: Tuesday 11am

Tutor: Abhaas Goyal

Contents

1 Introduction

This Technical report documents the structure of the assignment solution and offers a reflective analysis of design choices made and the results and structure of the testing regime. The program is designed to take user inputs and produce a picture onscreen using the Haskell CodeWorld package.

2 Design Documentation

Part 1 consists of three functions. The first, `toolToLabel` in `model View` case matches an input of a tool, ignoring all other properties to return a string output of instructions to the user on tool use. The second function `nextColor` in `module Controller` uses case matching to cycle through colours according to the specified order. Finally, the function `nextTool` in `module Controller` cycles a particular input of an empty tool to the next tool in the sequence also with empty parameters. It also uses case matching.

Part 2 Contains four functions the first of which, `colourNameToColour` in `module View` case matches elements of type `ColourName` and returns the same information in the typef `Colour` which codeworld uses. The second function `shapeToPicture` takes the information kept within type `Shape` and converts it to a codeworld `Picture` that can be printed to the display. Where most inputs were case matched to equivalent the specifications of `Rectangle` were through some linear algebra converted to a `solidPolygon` of four points. The `Cap` is a combination of the codeworld `clipped` and `circle` functions transposed as the user specifies. The third function `colourShapeToPicture` takes input of the type `colourShape` and returns the coloured shape in type `Picture`. The helper functions `distance` and `otherTriPoint` assisted implementation, the former calculating circle radii and the latter the third point of the isosceles triangle. Finally, the function `colourShapesToPicture` recursively runs through an input of type `[ColourShape]` and returns each member as a composite `Picture`.

Part 3 consists of one key function and six helper functions. The main function `handleEvent` cases on either keystroke inputs or mouse key inputs to produce the intended picture output. Presses of backspace and delete calls the function `deletePress` which removes the head of the list of shapes to remove the most recently added shape from the image. The spacebar input calls the function `endPoly` that takes any list stored in type `Tool` and then adds a codeworld polygon to the list of colourshapes. Key inputs of + or - call the functions `scaleRect` and `negScaleRect` respectively that add or decrease the scaling factor stored in a rectangle tool. Mouse presses call the function `pointPress` that cases on the type of shape tool being used to store the pressed point in the desired tool. Further the helper `pointRel` is called upon

mouse releases generally to complete a shape adding it to the list of shapes and returning an empty tool. There are two cases on `CapTool` to determine if it is storing the second point or the y coordinate of the cutoff point.

3 Reflection

Part 1 used a case statement for all three functions as they were all nessisarially injective with a finite domain. Consequent to this there was no need to test the truth of any statements. `nextTool` used a wildcard on the final cast to catch any half completed shapes and avoid the program crashing and instead just returning the held tool.

Part 2 Whilst `colourNameToColour` was a very simple case matching function the function `shapeToPicture` is more complicated. It case matches on the tool used. For both the triangle and rectangle inputs the `solidPolygon` function was used to create the associated picture due to the specifications of the input not aligning well to a unique CodeWorld function. For the triangle the points used were the two given points and a third given by the function `otherTriPoint` that calculated the other isosceles point. For rectangles some vector maths is used in teh definition to define the two other points as a translation of the first two points of a degree dictated by the scaling factor. The specifications for producing a cap required another nested case to determine if the cutoff was below the circle or not. if it was it would just return a circle, otherwise the desired cap would be produced. This was necessitated by the particular clip window and translations used. `colourShapeToPicture` used a simple casing on the possible pairs of colourshapes to return the appropiiate coloured picture using the appropriate CodeWorld function. Finally it was necessary to recurse through the list of colourshapes in the `colourShapesToPicture` function as the list could be of any length.

Part 3 was a simple implementation. The main function `handleEvent` cased on different inputs and would, instead of nesting cases, call appropriate helper function(s) which could case on the required part of the input to produce the desired output . All six helper functions were guarded by a wildcard at the end to reduce the risk of errors.

3.1 Assumptions

Due to a gap in the specifications for `handleEvent` it was assumed that the function `pointRel` should reinitialise the scale factor of the rectangle tool at `1.0` upon the completion of a rectangle on release of cursor.

This is hoped to enhance functionality when a user has used an extreme value of the scale factor and wants to quickly return to a reasonable scale factor for the next rectangle input. For `colourShapesToPicture` it was assumed that in case of an empty shapes list it should return a blank canvas, thus the prespecified CodeWorld function `blank` was used.

4 Testing

Part 1 Part 1 composed of the functions `toolToLabel` `nextTool` and `nextColour` was tested using the provided cabal test run under the command `cabal v2-test`. It passed `1 of 1 test suites`. Further simple tests were condcuted within development based on calling function inputs in the terminal to ensure the case matching was working in correspondence to the intended inputs.

Part 2

Part 3