Learning Outcomes Outline for Gunbarrel

Link to Code Showcase: https://youtu.be/nbEgITsEtGY
Link to Animation Demonstration: https://youtu.be/p5zotoWNo_E

ULO1: During the development of the Gunbarrel program, a lot of debugging was performed on various aspects of the sequencing and rendering system. Much of this debugging code still exists outside the code files in a text document called debugging_code.txt in the source code's main directory. What it did was it used the terminal to record and keep track of things such as the overall progress of the animation, how the bitmap cache (the bitmap array) was behaving as well as at one point, how the muzzle flash timer was behaving (this kept track of how long the muzzle flash image was on the screen for). It also kept track of all the properties of the moving circle from the first sequence. In terms of the actual code itself, all variables use snake case and are correctly indented in brackets.

ULO2: Evident in the code for the Gunbarrel program, the code features proper syntax and semantics so that it is easy to read by other programmers and easy to understand. The code itself has been written in a way that makes it as memory efficient as possible. This is why you will notice that procedures and functions will only make use of local variables when absolutely necessary. Pass-by-references are used heavily as well so as to avoid making copies of data in memory such as images and shapes every time procedures are called when a new frame is to be drawn onto the screen. Gunbarrel also makes use of an algorithm that can be found in the second hand execution task that inserts a new value into a vector at the position of the programmer's choosing. This was used in order to manipulate the order in which bitmaps were rendered on the screen.

ULO3: Gunbarrel makes use of custom data types such as structures and enumerations. It also makes use of nearly all the concepts the unit teaches such as while loops, if statements, switch statements and for loops. The program in its entirety is made up of modular functions and procedures that work together to create the animation it was intended to display. For instance, the rendering system runs inside the sequencing system, but the sequencing system also utilizes procedures that are specific to its role in the animation. As for the rendering system, it too is made up of much smaller procedures that tell it what things to render on the screen based on what the sequencing system decides.