# Hubert Czajczyk 20510716

# Initial Planning for Developing Software for a Drawing Robot

# Outline of the Problem to be Solved

(Maximum 1 page)  
The goal is to develop software, programmed in C language, to control a writing robot that would write out text given by a user from a file at a font specified by a user by console input.

The program is required to:

* Read and store font data from a text file “SingleStrokeFont.txt”
* Get text height (font size) and text file name from the user keyboard input
* Read and process text from the input file making sure the text fits within 100 mm width
* Process and send each word separately to the robot
* Convert font data to G-Code readable by the robot for each character before sending it to the robot

# Key Data Items

|  |  |  |
| --- | --- | --- |
| Name | Data type | Rationale |
| FontData Character[] | Array of Structs | * Structs allow to group different data types together – text data contains three values (X, Y, P) for each character * they are more readable than arrays in the code, for example, x, y, and penState are easier to read than array indices, * less error-prone when updated |
| FontSize | int | Character height/font size is an integer |
| Scale | int | Scale to multiply the robot movements by |
| GCode[] | string | A string containing GCode to be read by the robot including \n , string have dynamically allocated memory |
| Buffer[] | string/char array | Stores the character being processed |
| OffsetX, OffsetY | int | Stores the offset coordinated needed for the next character to be printed |

# Function Declarations

int ReadAndStoreFontData( const char \*FileName, struct FontData \*Character[])  
 Parameters:  
 FileName – name of the file with font data: “SingleStrokeFont.txt”  
 FontData Character[] – pointer to array of structures to store data for each letter from the font data file, includes coordinates for each movement and ascii code for the character and number of operations/lines  
 Return value – returns 1 if successful, 0 if failed

int GetFontSizeAndScale( int \*FontSize, double \*Scale )  
 Parameters:  
 FontSize – pointer to return font height input by the user  
 Scale – pointer to return for scaled robot movements by user input to create desired font size  
 Return value – returns 1 if successful, 0 if failed

int GenerateGCode( int Scale, struct FontData \*Character[], char \*GCode)  
 Parameters:  
 Scale - scaled robot movements based on input font height

FontData Character[] – pointer to array of structures storing required movement coordinates data for each letter  
 GCode[]– pointer to char array to store GCode for each character  
 Return value - returns 1 if successful, 0 if failed

# Testing Information

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Test Case | Test Data | Expected Output |
| Main() | Subfunctions executing correctly | Subfunctions return values | Return 1; |
| Main() | Opening File | (inputFile != NULL) | 0 -> return 1; |
| int ReadAndStoreFontData() | Opening File | (inputFile != NULL) | 0 -> return 1; |
| int GetFontSizeAndScale() | (4 < FontSize < 10) | FontSize | 1 -> Return 1; |
| int GenerateGCode() | GCode[ ] == “\0” | GCOde[ ] | 0 -> Return 1; |

# Flowchart(s)

Attached PDF in git repository