# Lily-May Silk 20458465

# Software Planning Project 2024/25

# Outline of the Problem to be Solved

(Maximum 1 page)

The program needs to read a type of file called a txt file, which only contains 3 columns of numbers.

First in the first column it will have a 999. This represents the start of a character. In the following column will be the ASCII value, and the last column will show how many strokes the robot will take to make the character, both off and on the paper.

Following on from the row with the 999 there will be a series of rows that map out the x (first column) and y (second column) coordinates to program where the robot moves with each stroke. The last column of these rows will contain either 0 or 1, where 1 represents the pen on the paper, and 0 represents the pen off the paper.

Additionally, you cannot split words, so you have to ensure that when you get to the end of the paper, you start a new row BEFORE you begin to write the next word. This means that you have to read each character 1 by 1 until you get to a character that is a space (ASCII value = 32) and then go back and write out this word in a row with sufficient space. This also means you need to keep a track of your x coordinate, to know how much space is left until the end of the row (where the x-coordinate spans from 0 to 100). Additionally, you will need to input the ascii value for a new line before you write out the word (ASCII value = 11). Hence you read the txt file one character at a time, and you process the txt file one word at a time.

Size of characters: maximum height (for letters like H) will be 10mm and minimum height (e.g. for the small part of the letter d) will be 4mm. All characters will be within an 18x18 units grid where minimum x and y axis will be 4\*(18/10)=7.2 units. Only one character will occupy the 18 units in the x direction at one time.

# Key Data Items

What data are you stroing – e.g. stored data in a structure

|  |  |  |
| --- | --- | --- |
| Name | Data type | Rationale |
| DataSet | Structure | This will be used to read in and store the values in the x, y and z columns. |
| grid | Array | This will store the values from the text files all in one place as one variable. |
| i  j  d  h | Integer | These integers will be used to keep track of which lines have been read, and will allow the program to progress to the next line in the txt file, and will record the height of the font. Chosen int as I wont need to take up too much storage (so doesn’t need to be unsigned) |
| y0  x0  X | Floating point data | These have been chosen as float, as due to the division that takes place, more bytes will be used up hence need to use floating point data rather than integer data |
| SingleStrokeFont.txt | FILE | This will contain all the coordinates |
|  |  |  |

Extend table as required

# Function Declarations

*Only include functions that you will develop.*

*Int main()*

*Float ProcessingWord (float x0, float y0, float X, float I, \*float Word)*

*Parameters:*

*X0 –start value of the x-coordinates of each word in mm - input*

*Yo - start value of the y-coordinates of each word in mm - input*

*X – length of word in mm*

*I – number of characters within the word*

*Word – pointer to return the array of the word - output*

*Return value – returns 1 if successful, 0 if failed*

*Float StoreData(FILE fOutput, float \*grid)*

*Parameters*

*fOutput – contains the SingleStrokeData.txt data - input*

*grid – pointer to return the completed array containing the txt file data - output*

*Return value – returns 1 if successful, 0 if failed*

# Testing Information

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Test Case | Test Data | Expected Output |
| Int main | Reads in file and incorporates all the functions to convert word (from processingWord) into GCode | Float ProcessingWord  Float StoreData | GCode |
| Processingword | Identifies and returns a word | Float x0, float yo, float X, | Float word |
| StoreData | Stores the txt file into an array that can be read by C program | Int x, y, z | Struct DataSet grid[1027] |

*Extend table as required. Note that ‘Function’ includes main()*

# Flowchart(s)