Definition of how YAML objects should be formatted. Version 2. Written 1st April 2019

References to sub-sequences can be done with their names.

The ID of regimens and sequences, which is defined by the FarmBot when it receives it, will be written to internal storage automatically.

Example of a YAML file:

```
""""
CSV: my_map.csv
other_files: ["yaml_file1", "yaml_file2"]  # This field is optional, the program will always look in its own file first.
PIN_1 : water_pin # optional labels for pins
scale : "1=10cm" # option to set the scale of coordinates
default_speed : 50 # option to define a default speed
default_z : 0 # option to define a default z-coordinate value
default_x_offset : 0 # option to define a default x-offset value
default_y_offset : 0 # option to define a default y-offset value
default_z_offset : 0 # option to define a default z-offset value

# The program assumes every object with a "start_time" field is a Farm Event
my_event:
  start_time: date_time
  repeat_event: {every: <<default 1>>, unit = "minutes/hours/days/weeks/months/years", until: <<date_time>>} #optional
  schedule: [{group: [optional], type: [optional], days: [], times: [], actions: <<list of actions or name of sequence>>}]
# Each list of actions can be iterated over a group or plant type, as defined in the CSV file
# The existance of a schedule field means each list of actions is repeated at the set days and times
# The list can be defined as a range. For example, "1,10,2" means every second number starting from 1 and smaller than 10.
# and "23:00,0:00,0:10" means every 10 minutes, starting from 23:00 and ending before 0:00

my_event:
  start_time: date_time
  group: ["extra_water_group"]
  type: ["radish"]
  schedule: [{group: [optional], type: [optional], every: 4, unit: "minutes/hours/days/weeks/months/years", actions: []}]
```

# If an item in the schedule list does not have group and doess not have type, it defaults to the option defined above it
# This Farm Event also showcases an alternative way of defining schedules

```
my_event:
  start_time: date_time
  group: ["extra_water_group"]
  type: ["radish"]
  actions: [<<actions or name of sequence>>]
# If the object also has a "schedule" field, the "action" field will be ignored

# The program assumes every object without a "start_time" field but with a "schedule" field is a regimen
my_regimen:
  schedule: [{group: [optional], type: [optional], every: 4, unit: "minutes/hours/days/weeks/months/years", actions: []}]
  color: "gray/green/blue/yellow/orange/purple/red" #optional

# The program assumes every object without a "start_time" and without a "schedule" is a sequence
my_sequence:
  color: #optional
  actions: []
# Individual actions in a list can also be a string that refers to the name of a sequence
"""
```

The following are the keywords and formats we defined for single FarmBot commands such as "Move Absolute" and "Write Pin".

```
 MOVE_REL: {x: 0, y: 0, z: <<defualt 0 or as defined>>, speed: <<defualt 50 or as defined>>}
 MOVE_ABS: {x: 0, y: 0, z: <<defualt 0 or as defined>>, x_off: <<default 0 or as defined>>, y_off: <<default 0 or as defined>>, z_off: <<default 0 or as defined>>, speed: }
 IF : {cond: <<see below>>, then: , else: }

 cond: "x = 0 AND y > 0 OR PIN_1 < 0 OR sensor_pin = UNKNOWN"

 FIND_HOME : [x,y,z]
 MESSAGE : {text: <<note: {{x/y/z}} is a special tag>>, type: "success/warning/busy/error/info"}
 WAIT : "time in milliseconds"
```

```
READ_PIN : {pin : "<<PIN_# or a name you chose>>", label: <<optional, default is pin name>>mode : "D/A"}
WRITE_PIN : {pin : "<<PIN_# or a name you chose>>", value: "ON/OFF", mode : "D"}
WRITE_PIN : {pin : "<<PIN_# or a name you chose>>", value: "0 to 1023", mode : "A"}

TO_SELF # Go to the current plant in a loop
TO_PLANT  # go to a plant with this (unique) name, this saves typing in coordinates by hand if you have a special plant.
```

e.g. we can have:

```
my_sequence:
  group: ["water_group"]
  actions: [WAIT:"1000", TO_SELF, "water_plant", TO_PLANT:"plant_name"]
```