Stage 1 - Store ALL numbers in the string

```
Read String:
    Do
        Check if RDRF flag is set
        While RDRF flag is not set
            Check if RDRF set
        Load byte from serial to accumulator A
        If ‘end of string character’
            Branch to Read Done
        Store byte in accumulator A into reserved memory for string pointed to by X
        Increment register X by 1
        Branch to Read String

Read Done:

Set register X to point to the start of the string
Set register Y to point to a new memory location for numbers

Extract Number:
        Load A with value pointed by X
        Compare byte in A to ‘end of string character’
        If same
            Branch to Stage 2
        Compare byte in A to hex value $30
        If less than $30
            Increment X
            branch to Extract Number
        Compare byte in A to hex value $39
        If greater than $39
            Increment X
            Branch to Extract Number
        Store byte in A to memory reserved for numbers pointed by Y
        Increment Y
        Increment X
        Branch to Extract Number
```

Stage 2- Map numbers to 7-seg

```
Map_Start:
Load register X with location of the ‘number’ string
Load accumulator A with the current number (pointed to by X)
Compare current number with NULL to see if there are any numbers left
If NULL
    branch to Stage 3

Compare the current number with value $30
If same
    Load register A with $3F
    Branch to Store_Value
Compare the current number with value $31
If same
    Load register A with $06
    Branch to Store_Value
Compare the current number with value $32
If same
    Load register A with $5B
    Branch to Store_Value
Compare the current number with value $33
If same
    Load register A with $4F
    Branch to Store_Value
```

```
Compare the current number with value $34
If same
    Load register A with $66
    Branch to Store_Value
Compare the current number with value $35
If same
    Load register A with $6D
    Branch to Store_Value
Compare the current number with value $36
If same
    Load register A with $7D
    Branch to Store_Value
Compare the current number with value $37
If same
    Load register A with $07
    Branch to Store_Value
Compare the current number with value $38
If same
    Load register A with $7F
    Branch to Store_Value
Compare the current number with value $39
If same
    Load register A with $6F
    Branch to Store_Value

Store_Value:
Store the new value in accumulator A in the location pointed to by X
Increment X

Branch to Map_Start
```

Stage 3 - Displaying the numbers

```
Scrolling:
    Check if the current number string is less than 4 characters
    If true
        branch to Load less than 4
    If false
        branch to Load 4
```

```
Load less than 4:
    Load accumulator A to the length of ‘number string’

    Load head of number string to index register X

    Load head of display string to index register Y

    Load the value from number string to the display string

    Decrement X to next position

    Increment Y to next position

    Check If the accumulator A has become 0

    If true,
        Branch to Display Function
    If false,
        Branch to Load less than 4

Load 4:
    Load accumulator A to the length of ‘number string’

    Load head of number string to index register X

    Load head of display string to index register Y

    Check if the current number should be from the tail of the string

    If true,
        Load tail of number string to index register X

    Load the number from number string to the display string

    Decrement X position

    Increment Y position

    Check If the accumulator A has become 0

    If true,
        Branch to Display Function
    If false,
        Branch to Load 4
```

Display Function:

Display the numbers in display string onto the 7-segment

Delay the display for 0.5 seconds

Branch to **Scrolling**