Homework #6 (200 pts)

Your job is to write a simplified run-length encoder and decoder in x86 assembly.

- Run-length encoding is a straightforward compression algorithm that works by outputting a sequence of <character, number> pairs representing the input.
  - For example, the sequence AAAAAABBC would be encoded as A6B2C1.
    - ‘A’ occurs 6 times, ‘B’ occurs 2 times, ‘C’ occurs 1 time...
- Run-length decoding decompresses a run-length encoded sequence of <character, number> pairs by outputting the original sequence.
  - For example, A6B2C1 would be decoded into AAAAAABBC.

For the run-length encoder:
- Read user input from standard input.
- Your program must output the encoded sequence to standard output.
- You may assume that a character will not repeat more than 9 times in a row.
  - You DO NOT have to error check for this.
  - I will test your code with valid input only.
- Your encoder must work for all visible ASCII characters:
  - upper / lower case letters, numbers, whitespace, operators, and punctuation.
- Submit a single Makefile that assembles and loads your source file.
  - Name the x86 assembly source file encoder.asm
  - Name the executable encoder.
- As always, your source file must assemble using the nasm assembler and execute on a 32-bit Linux computer.
- Examples: (red = input from stdin; blue = output to stdout)

UNIX> ./encoder
aaaaaabbc
a6b2c1

UNIX> ./encoder
...$$&&&((()))<<<<>>>>
.3$2&4(2)3<3>5

UNIX> ./encoder
MISS ISS IPP I
M1I1S2 11I1S2 2I1P2 5I1

UNIX> ./encoder
aaaaaaaaaa@@@@@@@@@*****&&&&^%^%^%^%%%%%%%======
a9@9*6&4^6%4=6
For the **run-length decoder**:

- Read user input from *standard input*.
- Your program must output the decoded sequence to *standard output*.
- You may assume that the encoded input sequence will *always* be correct:
  - The number will *always* be between 1 and 9 (inclusive).
  
    - You **DO NOT** have to error check for this.
  - I will test your code with *valid input only*.
- Your *decoder* must work for all visible ASCII characters:
  - upper / lower case letters, numbers, whitespace, operators, and punctuation.
- Submit a single *Makefile* that *assembles and loads* your source file.
  - Name the x86 assembly source file *decoder.asm*
  - Name the executable *decoder*.
- As always, your source file must assemble using the *nasm* assembler and execute on a 32-bit Linux computer.
- Examples: (red = input from stdin; blue = output to stdout)

```plaintext
UNIX> ./decoder
a6b2c1
aaaaabbc

UNIX> ./decoder
.3$2&4(2)3<3>5
...$$&$&$((&))<<<<>>>>

UNIX> ./decoder
M111S2 111S2 211P2 511
MISS ISS IPP I

UNIX> ./decoder
a9@9*6&4^6%4=6
aaaaaaaaaa@@@@@@ @@*****&&&^&&&^%%%%==

• You can also make use of the “pipe” command common in Linux shells:
  - *stdout* from left process is “piped” as *stdin* to right process:

```
UNIX> ./encoder | ./decoder
AAAAABBBS$%%%%!!!*****((((()))))==++++
AAAAABBBS$%%%%!!!*****((((()))))==++++
```

To summarize, you must submit the following **three** items:

1. *encoder.asm* – x86 assembly source for the run-length encoder
2. *decoder.asm* – x86 assembly source for the run-length decoder
3. *Makefile* – a single *Makefile* that can assemble / load both programs.