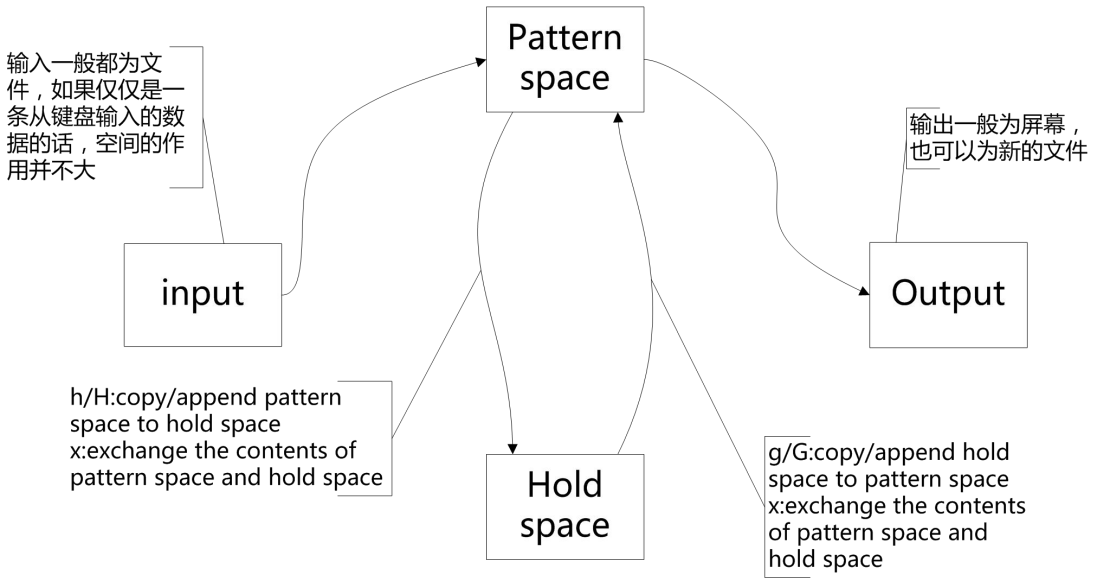
Pattern space 和 Hold space

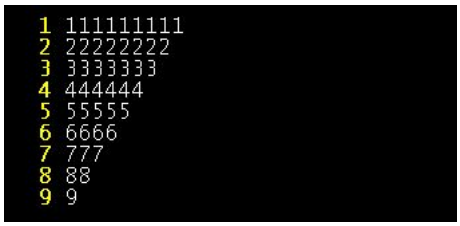
Sed 是对数据流进行操作的一个强大的工具，大家已经经常使用到 sed 对文本进行修改，替换。这里就不介绍 sed 的使用方法了，本文主要介绍 sed 中的 Pattern space 和 Hold space。理解了这两个概念有助于从本质上理解 sed 处理数据流的方式，尤其是像实现逆转、只输出奇数或偶数行等等功能，Pattern space 和 Hold space 即模式空间和保留空间（也可以称之为缓冲区），保留空间的初始为空，我们可以认为是一个空行。下面图示了 sed 中的输入流，输出流，模式空间以及保留空间：



先简单介绍下 sed 的一般工作模式（在没有利用到 Hold space 的时候），pattern space 每次从 input 中取一行数据到 pattern space 中，然后经过一些处理，将一行数据放入 output 中，然后 output 会输出到屏幕（默认为屏幕，也可以是文件）。记住，此时 pattern space 还储存着这一行数据，直到 input 再放入第二行数据取代第一行数据。

当 sed 的命令为 sed [-n] '[hHgGx]p' file 格式时，需要涉及到 hold space，其实我们可以简单的讲 hold space 理解为 sed 的一个缓冲区就好了，只是这个缓冲区不会直接进行输出，并且只有 pattern space 可以对其进行操作，放入或者拿出数据（如上图所示）。

下面解释下 [hHgGx] 这几个参数：（以 file 做例子，file 中的数据为：



)

Man sed 中的解释为：

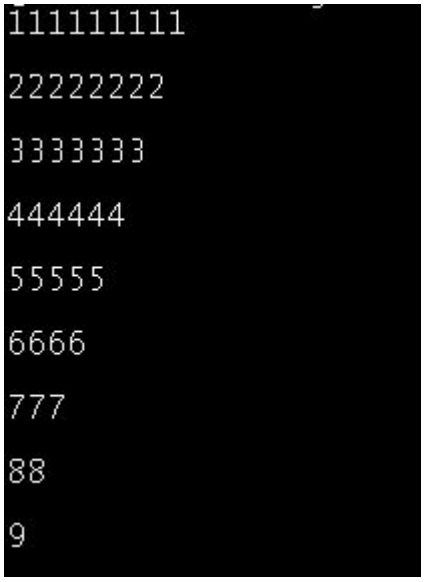
h/H:copy/append pattern space to hold space（复制或者追加模式空间的数据至保留空间）;

g/G:copy/append hold space to pattern space（复制或者追加保留空间的数据至模式空间）;

x:exchange the contents of pattern space and hold space（交换模式空间和保留空间的数据）;

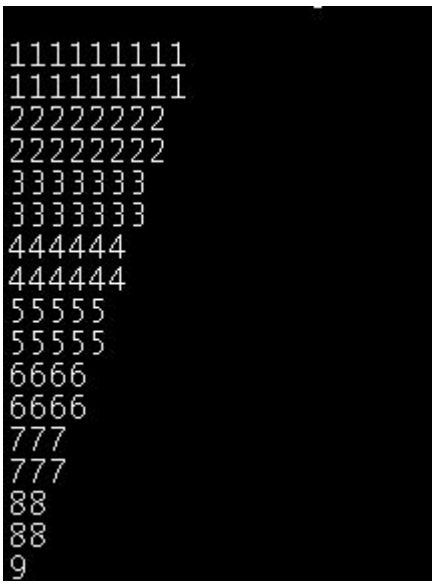
一般会将 h/H 和 g/G 以及 x 联合起来用，可以达到栈和队列的使用目的。

G：sed 'G' file



因为 hold space 的初始为一个空行，并且始终没有修改其中的数据，一直保持的是空行，所以在每次执行 G 命令时，会将空行追加到每行数据之后。

x：sed 'x;G' file



因为开始 x 命令将 hold space 的空行和 pattern space 中的 111111111 交换，然后 G 命令又将 111111111 追加到 pattern apace 的空行之后，然后输出了

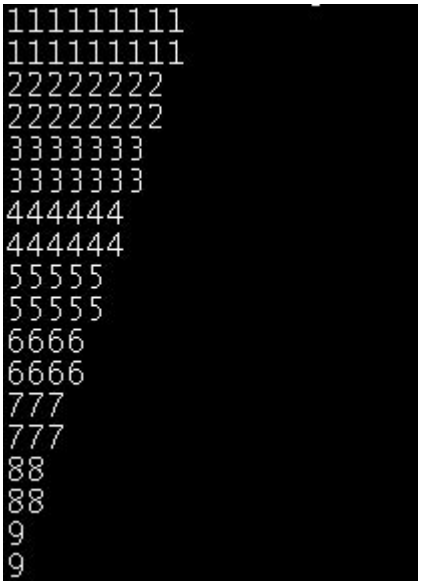
,此时 hold space 中依旧为 111111111 ,接着 pattern space 中进入 22222222，x 命令将 hold space 的 111111111 和 pattern space 中的 22222222 交换，然后 G 命令又将 22222222 追加到 pattern apace 的 111111111 之后，然后输出了

，依次类推，就输出了结果。可是看到过程如表格所示：

命令	sed 'x;G' file				
	hold space		pattern space		
执行命令过程	执行前	执行后	执行前	执行后	输出
执行 x 命令	空行	111111111	111111111	空行	无输出
执行 G 命令	111111111	111111111	空行	空行 111111111	

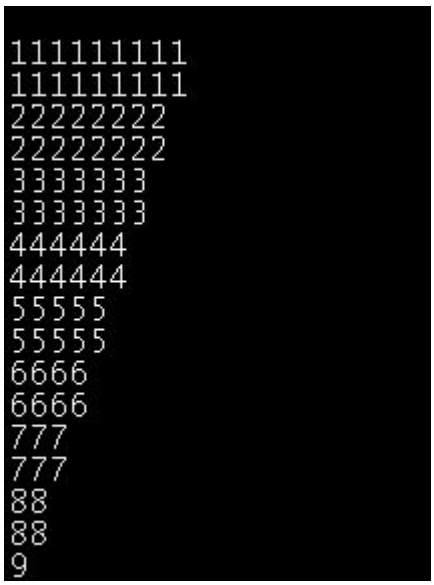
执行 x 命令	111111111	22222222	22222222	111111111	无输出
执行 G 命令	22222222	22222222	111111111	111111111 22222222	111111111 22222222
执行 x 命令	22222222	3333333	3333333	22222222	无输出
执行 G 命令	3333333	3333333	22222222	22222222 3333333	22222222 3333333
执行 x 命令	.....				
大家应该能思考到为什么最后只有一个 9					

h：sed 'h;G' file

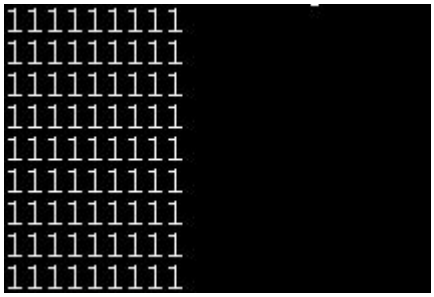


命令进行操作的思路，大家可以参照 x 命令的方法进行一步步的演示。

H：sed 'H;x' file



g：sed '1h;g;x' file



下面解释下'1h;g;x'中“1“的含义，表示只有第一行执行 h 命令，将 111111111 复制到 hold space 中去，之后每次执行 g 命令的时候，都是从 hold space 中将 111111111 复制出来，覆盖掉了 pattern space 中的数据，所以结果显示为打印了 9 行 111111111。

而$表示只有最后一行执行这个命令。

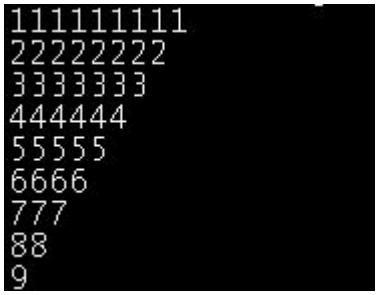
在 1 或$和命令中添加 “！“，表示只有第一行或者最后一行不执行这个命令。

有关于更改多的命令 n/N、d/D 等，请大家自己查阅 man sed 或者其他资料。在遇到复杂的命令的时候，希望大家可以手动进行画图来执行每个命令，这样就比较清楚了。

下面有 4 个例子来解释下选项 n 的作用：

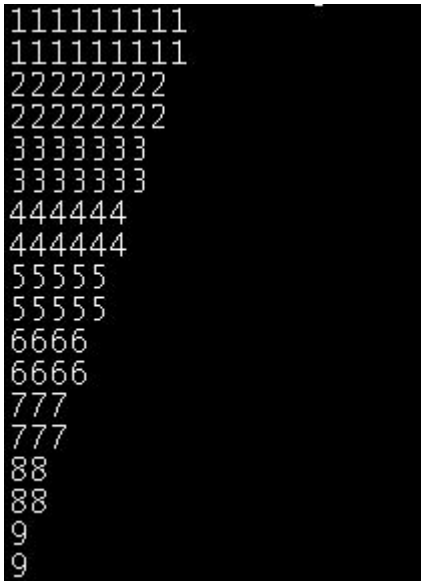
1. sed '' file，这句命令会将 output 中的数据进行输出：

逆序文件：sed '1!G;h;$!d' file



2. sed -n 'p' file，这句命令虽然显示的结果和上一个一样，但机制是不同的，这句命令是要求输出 pattern space 中的数据；如果大家觉得这个说法有点模糊，再继续看下面的例子。

3. sed 'p' file，先看结果：



这条命令输出了两遍，为什么呢？因为不仅将每次 output 中的数据进行了输出，接着又将 pattern space 中的数据再要求输出一次。

4. sed -n '' file，大家应该能猜到这个输出什么结果了，那就是空，因为选项-n 指明了要讲 pattern space 中的数据进行输出，而缺少了 p 命令，所以不能将 pattern space 中的数据进行输出。

逆序文件：sed '1!G;h;$!d' file

读一次相当于循环一次，重复执行' '中的命令操作！！