Asymptotic performance of mutating behaviors of the list class				
		Dorvation	Pun Time	
			0(1)	
		date-append (val)	0(1)*	
		datajaser + (kival)	O(n-k+1)*	
		data.pop()	0(1)	
		data vemore (val)	$O(n)^{\frac{1}{2}}$	
		data) 3= data2	Q(n²)	
		data.revered()		
		data.sortl	O(n(ogn)	
Adding Element to a list thre complicating factors for efficiency:				
			ncy:	
		1.) the addition of an element may regular us to reside dynamic across		
2.) shifting elements to make room				
	1	ef Insert(self, k, val):		
if selfn == selfcapacity: // not enough room selfresize (2* selfcapacity) // double capacity			[1] [1] [1] [1] [1] [1] [1] [1] [1] [1]	
			의 전투를 보고 있는 것이 있는 것이 없는 것이 없는 것이 없는 것이 없는 것이 되면 한 것이다. 그런 것이 없는 것이다. 그런 것이 사람들이 다른 것이다고 있다면 없는 것이다.	
		for; In range (selfn, k,-1)	: // shift right flost	
		self. A[j] = SelfA[j-1]		
_	selfA[k]: val		Ustore nemer element	
		selfn+=		
		0 1 1 1 1 1	11 11 1 0/3	
Pernoving: there is no improvement to removing by value, always $\Omega(n)$		nt to removing by value, always SCLM.		
		def vemore(self, val):		
		for k in range(selfn): if selfA[k]== val: // found match		
		for ! come !! call		
		for ; in range (k, selfn-1): // shift others to fill gay selfA[j]: selfA[j+1]		
M		selfA[sufn-1]= None 11 help garbage collection		
		Solfn -=	1 on less item	
		return		
		rose Valuetros		