	package com.eomcs.quiz.ex01.sol;			1 (2		3	4	7	6				
	package com.eomcs.quiz.exoi.soi,					9-)+	, 9+	9+ 36-1 = 35번 바ョ	일수 있음			
	// 출처: codefights.com			3 2	2	-								
	//			4 3		9-2=7	9-3=6	9-4=5	9-5=4					
	// 한번에 한 자리의 숫자만 바꿀 때			5 4	4		2	3	4	5				
	// 단 맨 앞의 숫자는 0이 될 수 없다			6 5	5	:	3	4	5	6				
	// OH)			7 6	6	4	1	5	6	7				
	// 10 => 17 번			3 7	7		5	6	7	8				
	//			9 8	В	(3	7	8	9				
	// [시간 복잡도]			9	9	- 1	7	8	9					
	// - O(k) : k는 10진수의 자릿수이[다.				8	3	9						
	public class Test05 {		8번	9번		9	9							
	public class lestes (HISHIE 幕 報 (i하기 때문에 31만	13 바보하다									
	<pre>public static void main(String[]</pre>	args) {	012012240	10171 1112 011012	B c qea.									
	System.out.println(countWaysToCh	nangeDigit(10) == 17);												
	}													
	static int countWaysToChangeDigi	t(int value) {												
	int answer = 0;													
	while (value > 0) { answer += 9 - value % 10;													
	value /= 10;													
	}													
	return answer;													
	}													
	}													
				101 3101										
			1의 개수는 갈지	만, 값만 달라짐.										
	양의 정소 v를 2지수로 표현했을 때 1 강을 2	같는 비트 개소를 정소이 무게라고 정이함 때	00001>01	> 0										
	// 같은 무게를 가지는 양의 정수 중에서 x와	갖는 비트 개수를 정수의 무게라고 정의할 때, 가장 가까운 양의 정수를 구하시오!	00001201											
	public class Test06 {		10>	00001010										
	public static void main(String[] args) {	001010) == 0b00001001); // 10 ==> 9 001000) == 0b11000100); // 200 ==> 196	9	00001001										
	System.out.println(closestIntSameBit(0b000	001010) == 0b00001001); // 10 ==> 9	12	00001100										
	System.out.printin(closestintsameBit(0b110	JU1000) == 0611000100); // 200 ==> 196	10	00000110										
	, ,			00010010										
	static int closestIntSameBit(int x) {													
	static int closestIntSameBit(int x) { // 이 메서드를 완성하시오!		6	7	8	9	10	11	12		18			
	// 이 메서드를 완성하시오!		6	7	8	9	10	11	12		18			
	// 이 메서드를 완성하시오!	환한다.	6	7	8	9	10	11	12		18			
	static int closestInttSameBit(int x) { # 이 에서드를 완성하시오! # x의 모든 비트가 0이거나 1이면 오류를 반throw new IllegalArgumentException("모든 I	원한다. 비트가 0 또는 1이다.");	6	7	8	9	10	11	12		18			
	// 이 메서드를 완성하시오!	용한다. 배트가 0 또는 1이다가;	6	7	8	9	10	11	12		18			
	// 이 메서드를 완성하시오!	환산대. 비트가 0 또는 1이다.가;	6	7	8	9	10	11	12		18			
	# 이 에서드를 완성하시오! # X의 모든 바트가 0이거나 1이면 오류를 반 throw new IllegalArgumentException("모든 !)	환한다. 배트가 0 또는 1에다.");	6	7	8	9	10	11	12		18			
	# 이 매서드를 완성하시오! # x일 모든 비밀가 001가は 101만 오류를 반 throw new illegal Argument Exception("모든 []]] x: 00001010	혼한다. 비트가 0 또는 101대가;	6	7	8	9	10	11	12		18			
	# 이 에서드를 완성하시오! # X의 모든 바트가 0이거나 1이면 오류를 반 throw new IllegalArgumentException("모든 !)	환한다. 비트가 0 또는 1이다."):	6	7	8	9	10	11	12		18			
	# 이 매서드를 완성하시오! # x일 모든 비밀가 001가は 101만 오류를 반 throw new illegal Argument Exception("모든 []]] x: 00001010		6	7	8	9	10	11	12		18			
	# 이 교육도를 관심하시오! # # 보고 모든 배문가 001개나 101만 오구를 반 throw new Illegal ArgumentException("모든 1)) # # 10 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	i=0	6	7	8	9	10	11	12		18			
	# 0 표 서도를 환경하시오! # ### 모든 배트가 00 기니 10년 오류를 반 throw new The pal-Argument Exception "모든 1 } #### ##############################		6	7	8	9	10	11	12		18			
	# 이 교육도를 관심하시오! # # 보고 모든 배문가 001개나 101만 오구를 반 throw new Illegal ArgumentException("모든 1)) # # 10 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0	i = 0 int bit1 = (x >>1) &1; ==> 엔질 비트를 추출 int bit2 = (x>>(i+1) &1;	6	7	8	9	10	11	12		18			
	# 0 표 서도를 환경하시오! # ### 모든 배트가 00 기니 10년 오류를 반 throw new The pal-Argument Exception "모든 1 } #### ##############################	i= 0 int bit1 = (x>>i) &1; ==> 앤질 비트를 추출 int bit2 = (x>(i+1))&1; 두 비트교육을 위한 마스크 쥬비	6	7	8	9	10	11	12		18			
	# 이 행사도를 환경하시오! # # # # # # # # # # # # # # # # # # #	i=0 int bit1=(x>>i)&1; ==> 현광 비트를 추출 int bit2=(x>>(i+1)&1; 두 비르교환을 위한 마스크 준비 00000001	6	7	8	9	10	11	12		18			
	# 이 변체도를 환경하시오! # ### 보는 바라와 ###################################	i= 0 int bit = (x>>i) &1; =>> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교환을 위한 마스크 준비 00000001	6	7	8	9	10	11	12		18			
	# 이 에서드를 완성하시오! # 보고 모든 배드가 이에게나 1이면 오루를 반 throw new Hegal Argument Exception ("모든 1) x:00001010	i=0 int bit1=(x>>i)&1; ==> 현광 비트를 추출 int bit2=(x>>(i+1)&1; 두 비르교환을 위한 마스크 준비 00000001	6	7	8	9	10	11	12		18			
	### AND THE PROPERTY OF THE PR	i=0 int bit1 = (x>>i) &1; ==> 앤끌 비트를 추출 int bit2 = (x>>(i+1)&1; 두 비트교론을 위한 마스크 준비 00000001 < 1 < <ii>00000011 < mask</ii>	6	7	8	9	10	11	12		18			
	# 0 個 M C E 型金砂 M 21 # 22 2 2 1 1 2 7 0 0 7 1 1 1 1 1 1 2 2 元 書 世 throw new llegal/argumenti-ixception(**2 e) **********************************	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	9	10	11	12		18			
	### 기업 #	i=0 int bit1 = (x>>i) &1; ==> 앤끌 비트를 추출 int bit2 = (x>>(i+1)&1; 두 비트교론을 위한 마스크 준비 00000001 < 1 < <ii>00000011 < mask</ii>	6	7	8	9	10	11	12		18			
	# 0 個 M C E 型金砂 M 21 # 22 2 2 1 1 2 7 0 0 7 1 1 1 1 1 1 2 2 元 書 世 throw new llegal/argumenti-ixception(**2 e) **********************************	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	9	10	11	12		18			
	### 기업 #	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	9	10	11	12		18			
ex02.Test01	### 기업 #	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>•</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	•	10	11	12		18			
ex02.Test01	### 기업	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>•</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	•	10	11	12		18			
ex02.Test01	### 기업 #	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>•</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	•	10	11	12		18			
ex02.Tes101	### ### ### ### ### ### ### ### ### ##	i=0 int bit1 = (x>>i) &1; ==> 연광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000011 <1< <i 00000011="" <="" <1<<i="" ansk="" td="" x:00001010="" xor:00000011<=""><td>6</td><td>7</td><td>8</td><td>•</td><td>10</td><td>11</td><td>12</td><td></td><td>18</td><td></td><td></td><td></td></i>	6	7	8	•	10	11	12		18			
ex02.Test01	### AND THE PROPERTY OF THE PR	i= 0 int bit1 = (x>>i) &1; ==> 엔질 비트를 추출 int bit2 = (x>>(i+1)&1; 두 비트교론을 위한 마스크 준비 00000001 < 1 < <ii>1 00000001</ii>	6	7	8	•	10	11	12		18			
ex02.Test01	### NOTE ###	i= 0 int bit = (x>>i) &1; ==> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000001	6	7	8	•	10	11	12		18			
ex02.Test01	### AND THE PROPERTY OF THE PR	i= 0 int bit = (x>>i) &1; ==> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000001	6	7	8	•	10	11	12		18			
ex02.Test01	### NOTE ###	i= 0 int bit = (x>>i) &1; ==> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000001	6	7	8	•	10	11	12		18			
ex02.Test01	### 2	i= 0 int bit = (x>>i) &1; ==> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000001	6	7	8	•	10	11	12		18			
ex02.Test01	### 기계 ### #### #######################	i= 0 int bit = (x>>i) &1; ==> 현광 비트를 추출 int bit2 = (x>>(i+1))&1; 두 비트교론을 위한 마스크 준비 00000001	6	7	8	•	10	11	12		18			

	// 이 메서드를 완성하시오! return answer:									
	letum answer,									
	}									
ex01.Test07										
https://en.wikipe	dia_org/wiki/Trailing_zero									
	Factorial [edit]									
	The number of trailing zeros in the decimal representation	of n!, the factoria	al of a non-nec	gative integer n, is simply the multiplicity of the prime factor 5 in	nl. This can be determined with this special ca	se of de Polignac's formu	la: ^[1]			
	$f(n) = \sum_{i=1}^k \left\lfloor rac{n}{5^i} ight floor = \left\lfloor rac{n}{5} ight floor + \left\lfloor rac{n}{5^2} ight floor + \left\lfloor rac{n}{5^3} ight floor + \cdots +$	$+ \left \frac{n}{-\nu} \right $,								
		L5^ J								
	where k must be chosen such that									
	$5^{k+1} > n$,									
	more precisely									
	$5^k \le n < 5^{k+1}$,									
	$k = \lfloor \log_5 n floor$,									
	and $\lfloor a \rfloor$ denotes the floor function applied to a . For $n=0$									
	0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4	l, 4, 4, 6, (seque	ence A027868 i	n the OEIS).						
	For example, 5 ³ > 32, and therefore 32! = 2631308369336									
	$\left\lfloor \frac{32}{5} \right\rfloor + \left\lfloor \frac{32}{5^2} \right\rfloor = 6 + 1 = 7$									
		4								
	zeros. If $n < 5$, the inequality is satisfied by $k = 0$; in that of									
	The formula actually counts the number of factors 5 in n!,	but since there a	re at least as m	nany factors 2, this is equivalent to the number of factors 10, ea	h of which gives one more trailing zero.					
	Defining									
	$q_i = \left\lfloor rac{n}{5^i} ight floor,$									
	the following recurrence relation holds:									
	$q_0 = n,$									
	$q_{i+1} = \left\lfloor rac{q_i}{5} ight floor.$									
		of the summation	n which can b	be stopped as soon as q_i reaches zero. The condition $5^{k+1} > n$ is	equivalent to $\alpha_+ = 0$					
	I was can be used to simplify the companion of the terms		ing trimen con b	Total production of the condition of the	equitations to 4 k+1					
ex02.Test04	동비수열									
	108 108									
	0 1 2 3	4								
	0 1 2 3 2 6 18 54	162								
	x3 x3 x3	х3								
	1번항목과 2번 항목의 비율									
	1번항목과 0번 항목의 비율									
	2번항목과 3번 항목의 비율									
	2번항목과 1번 항목의 비율									
	0번과 1번의 비율을 가지고 나머지 값을 구한다.									
ex01.Test08		1								
			5:0101	0101 x 0011						
	10^2 10^1 10^0 20 x 7 = 140		3:0011							
	128 x 7 2 x 70 = 140			2 ⁰ 0001x0011 = 1 * 3 = 3						
	128 7x10^0 = 56	-	1	2^4 0000x0110 = 0 * 3 = 0	0과 곱하면 0이기 때문에 버린다.					
	12 7x10 = 140 1x7x10^2 = 700	-		2 ² 0001x1100 = 4 * 3 = 12 2 ² 0000x11000 = 0 * 3 = 0	0과 곱하면 0이기 때문에 버린다.					
	700+140+56=896	1		2000X11000 - 0 3 - 0 15	· 보이면 아이기 때문에 어딘다.					
	1557 145755 555			.5						
	128		5:0101	0101 x 0011						
	x 7		3:0011							
	896		1	210 1 0011 = 0 0 1 1	27 7 7 7 10 20 21 10 10 10 10 10 10 10 10 10 10 10 10 10					
		1	1	2 ⁴ 0 * 0011 = 0 0 0 0 0 2 ⁴ 4 * 0011 = 1 1 0 0	0과 곱하면 0이기 때문에 버린다.					
			1	2 ² 2 4 * 0011 = 1100 2 ⁴ 3 0 * 0011 = 0000	0과 골하면 0이기 때문에 버린다.					
	128 x 7			1 1 1 1 (15)						
	10-2 10^1 10^0		비트 연산자가							
	8*7 8*7		5:0101	2^3 2^2 2^1 2^0 이진수의 특징						
	20 * 7 2 * 70		3:0011	0101 x 0011						
	100* 7 1 * 700		1	2^0 자리계산 1 * 0011 = 0 0 1 1 1 * 0011						

	_		
			<mark>24 자리기찬</mark> 0 *0011 = 0000
			2*2 꼬리계산 4*0011 = 1100 (4배를 만드는 것이 원목으로 2배로 이용한것과 같다.)
			<u>200,001 = 0000</u>
			1111(15)
		비트 연산자가	J8078 7
ex01.Test09			
exu1.1estu9	원을 이루는 값에서 서로 마주치는 숫자.		
	전을 이무슨 없네지 지도 마무지는 <u>1</u> 지.		
	1 2 3 4	4 5	6
	1 2 3 4 2 5 8 1	4 5 1 14 1	7
	3		
	12 11 10 9	5 6	3 4 5 6 7 8 9 10 11 12
	11	5 6	
	10	6	8 9 10 11 12
	9	-	
	8 4	1	3+(n/2)
ex02.Test05			
	분수 비교 상대편 값을 곱해도 등호는 동일하다.		기장 크다라고 가정
			0 1 2
	6 x 3 x 5 5	2 x 3 x 6	5 2 5
	6	3	6 3 4
	3*5	2*6	6'2>3'5308?
	3*5 15	12	
	ex01, quiz11		
	*문자열에서 알파벳 개수 구하기, 중복 제거	for(int i = 'a'; i < 'z'; i++) {	
		//파라미터로 주어진 문자열에서	문자를 반복해서 꺼낸다.
	"cabca"	for(int x = 0; x <s.length(); td="" x++)<=""><td></td></s.length();>	
	abcdefghi 일치하면 count 111xxxxx	char c = s.charAt(x); // 문자일이 if (c == i) { //알파벳의 코드값과	지 한국도 그로 전혀 가면나. 루마이 국 군이에 2년에
	EAGLE COURT ITTAXAAA	result++ ; //개수를 한개 증가 시	20.
	result		2가 없다. 왜? (종복 카운트)는 여지 않는다.
	ex01, quiz12	4	Ⅲ6고 24 82
	반지름(rc)	_ /	
	(X6, Yc)	a=v	Cz $\sqrt{2}$ $\sqrt{12}$
		/ 0 - ,	$C = \sqrt{\lambda^2 + b^2}$ $C^2 = 0^2 + b^2$
	Ve Ve		2 3 6 5
	(×0,1/0)		C = O + D
	,	6	
	(× a , Ye)	N	
		b = x	
	(Xc. 16)		
	YC		
	,		
	b:110-x01		
	a=17n-ycl		
	C=01-b2		
	public cass Test11		package com.eomcs.quiz.ex01;
	public cass result		portuge control moneyer and t
	public static void main(String[] args) {		// 多月: codefights.com
	System.out.println(differentSymbolsNaive("cabca")) == 3);	
	}		#소문자 알파벳의 문자열이 주이질 때, 서로 다른 알파벳의 계수는?
	문자 코드값 'a'		W .
	비교		// (0) "cabca" ==> 3
	a' 문자 i값 0부터		
	문자코드 증가 > 문자로 출력		Given a string, find the number of different characters in it.
			control is seeing, and site internation of still control in the co

					for "cabca" output should be 3						
	문자의 값을 꺼낼때	l≒ char			ior cabca output silouid be 3						
	조사의 값을 깨달따 x번째 운자를 꺼냄.	_ Cital			linnutl etring e						
	▲근쎄 군사를 끼덤.				[input] string s						
					a string of lowercase latin letters						
					[output] integer						
					*/						
					II .						
					// [시간 복잡도]						
1											
					// - ?						
					II .						
1					public class Test11 {						
1											
					static int differentSymbolsNaive(String s) {						
					int requit = 0:						
1					int result = 0;						
1											
1					#알파벳 반복문을 돌린다.						
1											
1											
1					//for(int i = 0; i < 26; i++) {						
1					fordisk i = left i = left i + 1 f						
1					for(int i = 'a'; i < 'z'; i++) {						
1											
1					//파라미터로 주어진 문자열에서 문자를 반복해서	꺼낸다.					
1					for(int x = 0; x <s.length(); td="" x++)="" {<=""><td></td><td></td><td></td><td></td><td></td><td></td></s.length();>						
1					char c = s.charAt(x); // 문자열에서 인덱스 X의 문	사를 꺼낸다.					
1					if (c==i){//알파벳의 코드값과 문자의 코드 값이	SECTOR					
1					II (6 I) { // జ파컷의 고느값과 군사의 고느 값이	E90					
1					result++ ; //개수를 한게 증가 시킨다.						
1					break;// 다음 항목은 검사할 필요가 없다. 왜? [중략	ł 카운트]는 하지 않는다.					
1					}						
1					}						
					i						
1					//문자 코드값과 문자를 출력해 본다.						
					//System.out.printf("1\$%x == %1\$c\n", 'a' + i);						
1					//첫번째 값을 10진수로 ==> 두번째를 문자로						
					#이 메서드를 완성하시오!						
1					return result;						
					return result;						
(·					}						
1					}						
1											
1											
au01 auin14							 				
ex01, quiz14											
1											
1	123 -	> 12 -3	>1	->1							
		3	2								
	//나누기 10 //										
1	" - T > 10 II	S4 1 21 10		20.2							
1				30+2							
				320+1							
	1의 자리			320+1							
	1의 자리 10의 자리			320+1							
	10의 자리			320+1							
				320+1							
	10의 자리			320+1							
	10의 자리			320+1							
	10의 자리			320+1							
	10의 자리			320+1							
	10의 자리 100의 자리	TI # OKOTI		320+1							
	10의 자리	지 // 일정인지 /		320+1							
	10의 자리 100의 자리	지 // 일정인지 /		320+1							
	10의 자리 100의 자리	지 // 일정인지 /		320+1							
	10의 자리 100의 자리	지 // 일정인지 /		320+1							
ex01, quiz17	10의 자리 100의 자리 제곱으로 중가하는			320+1 321	624						
ex01, quiz17	10의 자리 100의 자리 제곱으로 중가하는			320+1	921						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 #주어진 수 보다 등	큰 수 중에서 한 개	이상의 0으로 끝	320+1 321	W24						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 #주어진 수 보다 등	면 수 중에서 한 개 main(String[] arg	이상의 0 으로 끝	320+1 321 나는 가장 작은 수를 구리	NS1						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제곱으로 증가하는 # 주어진 수 보다 등 public static void System out.println(를 수 중에서 한 개 main(String[] are (nearestRoundNu	이상의 0으로 끝 gs) { mber(122) == 13	320+1 321 나는 가장 작은 수를 구리 이);	921						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제곱으로 증가하는 # 주어진 수 보다 등 public static void System out.println(큰 수 중에서 한 개 main(String[] are (nearestRoundNu	이상의 0으로 끝 gs) { mber(122) == 13	320+1 321 나는 가장 작은 수를 구리 이);	921						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 #주어진 수 보다 등	큰 수 중에서 한 개 main(String[] are (nearestRoundNu	이상의 0으로 끝 gs) { mber(122) == 13	320+1 321 나는 가장 작은 수를 구리 이);	H21						
ex01, quiz17	10의 자리 100의 자리 M곱으로 증가하는 #주어진 수 보다 등 public static void System_out_printin(}	큰 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur	이상의 0으로 끝(gs) { mber(122) == 19 mber(99722) ==	320+1 321 321 나는 가장 작은 수를 구리 0); 99730);	921						
ex01, quiz17	10의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 큰 public state void System out println(System.out println) }	큰 수 중에서 한 개 main(String[] arg nearestRoundNui nearestRoundNui	이상의 0으로 끝! gs) { mber(122) == 13 mber(99722) ==	320+1 321 321 나는 가장 작은 수를 구리 0); 99730);	第23						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 # 주어진 수 보다 큰 public static void System_out_println(System_out_println() }	로 수 중에서 한 개 main(String[] arg inearestRoundNur inearestRoundNur	이상의 0으로 끝(gs) { mber(122) == 13 mber(99722) ==	320+1 321 321 121 321 1321 1321 1321 1321 1	M21						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 # 주어진 수 보다 큰 public static void System_out_println(System_out_println() }	로 수 중에서 한 개 main(String[] arg inearestRoundNur inearestRoundNur	이상의 0으로 끝(gs) { mber(122) == 13 mber(99722) ==	320+1 321 321 121 321 1321 1321 1321 1321 1	\$21						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 #주어진 수 보다 큰 public static void System.out.printin() } <	= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	5923						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 # 주어진 수 보다 큰 public static void System_out_println(System_out_println(S) }	= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) { mber(122) == 13 mber(99722) ==	320+1 321 나는 가장 작은 수를 구리 이): 99730):	\$121						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 라 System out printin(System out printin(= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	951						
ex01, quiz17	10의 자리 100의 자리 제곱으로 증가하는 #주어진 수 보다 큰 public static void System.out.printin() } <	= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	\$21						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 라 System out printin(System out printin(= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	921						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 라 System out printin(System out printin(= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	921						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 라 System out printin(System out printin(= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	921						
ex01, quiz17	10의 자리 100의 자리 100의 자리 제공으로 증가하는 #주어진 수 보다 라 System out printin(System out printin(= 수 중에서 한 개 main(String[] arg nearestRoundNur nearestRoundNur soundNumber(int v.	이상의 0으로 끝(gs) (mber(122) == 13 mber(99722) == alue) { Value의 끝 자리:	320+1 321 나는 가장 작은 수를 구리 이): 99730):	9424 24						

	만약 찾지 못했다면 -1을 리턴하라!			// maxDivisor(i // - left: 왼쪽 수	nt left, int right, int o	fivisor)							
	public class Test18x {			// - right: 오른국	즉 수. 오른쪽 수는 1	방상 왼쪽 수 보다 길	거나 커야 한다.						
				// - divisor: 나 5	는 수								
	public static void main(String[] args) { System.out.println(maxDivisor(2, 17, 4));			// 例) // maxDivisor(2	17 4\=>16								
	System.out.printin(maxDivisor(2, 17, 4)); }			// IIIaxDivisoi(., 17, 4) -> 10								
	static int maxDivisor(int left, int right, int divisor) { for (int i = right; i >= left; i) {	//큰 수를 찾는 것	선이기 때문에 반복	문을 돌릴 때 큰 수	에서 작은 수로 돌	된다.							
	if(i % divisor == 0) {	//주어진 수가 나	누는 수의 배수인	지검사									
	return i;												
	}												
	return -1;												
	}												
	}												
ex01, quiz24													
	1 2 3 v v	4 5	5										
	v v v	v	040										
	V		3^3										
ex02. test11													
	c2 = a2 + b2	*X () TI III () (**	서 가장 큰 수를 찾	느ㅋ									
		가장 큰 세번의 :	서 가상 큰 수들 죗 길이를 알아내야?	_ X. }.									
	c												
а													
	b												
	b and a second												

	1									
	l									
	l									
	1									
	1									
	l									
	l									
	1									
	l									

-									

	1									
	1									
	l									
	l									
	l									
	l									
	l									
	l									

	1									
	l									
	l									
	1									
	1									
	l									
	l									
	1									
	l									

I										