		(변수) = 인스탠스 ==> Heap 영역(에 만들어 진다.	Heap => gabag	je collector가 관리	하는 영역										
	static int b; static 필드(변수	e) = 클래스 ==> method Area														
	}															
*인스턴스(non-st	static) 필드															
			로컬번수	인스턴스 생성 !												
	class A{		A obj1 =	new A();		JVM stack			Heap							
변수를 만들라는			A obj2 =	new A();		main()			200							
	boolean v2;		A obj3 =	new A();		obj1	200		v1	v2						
	}					obj2	300									
			obj = local 변4			obj3	400		A의 인스턴스							
			매서드 흐름이	끝나면 사라짐												
	void m1() {					레퍼런스(인스턴	스 주소를 담는 것	= 레퍼런스)								
	}		new A(); = 인스	- 턴스												
	{}															
	obj1.m1();	call 했을때														
	obj1.v1(100);															
																_
*클래스(static) 플	필드(변수)															
	class A {	main{	method arad		1. Method Arad	에 클레스가 로딩	돌때 스태틱 변수가	생성.								
	static int v1; →	A obj;	A		+											
	static boolean v2;	A.v1 = 100; A.v2 = tree;	A class	v1 v2	+											
		A.vZ = tree;	JVM stack		+											
			JVM stack	Heap	1											
		,	1	1	1											
		}	1		1											
			-													
				+	4											
·클래스 로딩과 실	AIM															+
프네다 포요하 8																
							1									
	\$java Hello															
	sjava Hello 1. Hello.class 를 찾는다.															
,———	1. Hello.class 을 쏫는다. 2. Bytecode 검증			meth	od arad											
	2. Bytecode 원명 3. Method Area에 로딩한다.	→ .class	1 →		관리하기 쉽게											
	4. 스태릭 필드 생성		→	베서느고느 번수코드	정리해서											
	·· -4 ¬ = - 55					1										
	5. 스테링 분류 신역															
	5. 스태력 블록 실행 6. main() 호출 JVM으로 직접 4	실행할 경우만 실행	→ →	상속코드	메모리에											
	5. 스테릭 블록 실행 6. main() 호출 JVM으로 직접 설	실행할 경우만 실행														
	5. 스태릭 블록 실행 6. main() 호출 JVM으로 직접 설	실행할 경우만 실행			메모리에											
	6. main() 호출 JVM으로 직접 (실행할 경우만 실행			메모리에											
	5. 스테릭 블록 실행 6. main() 호출 JVM으로 직접 실 대략 필드 및 인스턴스 필드 생성	실행할 경우만 실행			메모리에											
*클래스 로딩, 스티	6. main() 호출 JVM으로 작정 (대략 필드 및 인스턴스 필드 생성	실행할 경우만 실행			메모리에											
*클래스 로딩, 스티	6. main() 호출 JVM으로 직접 (실행할 경우만 실행			메모리에											
*클래스 로딩, 스티	6. main() 호출 JVM으로 작정 (대략 필드 및 인스턴스 필드 생성		→ ·		메모리에											
*클래스 로딩, 스티	6. main() 호출 JVM으로 작정 (대략 필드 및 인스턴스 필드 생성	→ Exam0140.clas	→ .s	상속코드 	메모리에 배치 od arad											
*클래스 로딩, 스테	S. main() 호출 JVM으로 직접 4 대략 필드 및 언스런스 필드 생성 class Exam0140 Class A { static intv1 :		→ .s	상속코드 meth xam0140.class >	메모리에 배치 od arad		1. Exam0140.clas	ss								
*클래스 로딩, 스테	S. main() 호출 JVM으로 작권 6 대학 필드 및 인스턴스 필드 생성 class Exam0140	→ Exam0140.clas	→ S	상속코드 meth xam0140.class >	메모리에 배치 od arad 1/2번 v1 = 100 >> 5/6번		2. main() 호출									
*몸레스 로딩, 스티	E. main() 호출 JVM으로 작정 (대략 필드 및 언스런스 필드 생성 class Exam0140 data(: int v1 : int v2 : }	→ Exam0140.clas	→ S	상속코드 meth xam0140.class >' A class > 4번	메모리에 배치 ood arad											
*물래스 로딩, 스테	E. main() 호텔 JVM으로 작정 (대학 필드 및 언스턴스 필드 생성 class Exam0140 class A { static int v1 : int v2 : } main() (~ }	→ Exam0140.clas	→ S	상속코드 meth xam0140.class >' A class > 4번 JVM stack	메모리에 배치 od arad 1/2번 v1 = 100 >>5/6번 v2 Heap		2. main() 호출 3.main()의 로컬번	수 생성								
*물래스 로딩, 스타	6. main() 호출 JVM으로 작업 6 대학 필드 및 인스턴스 필드 상성 class Exam0140 class A { static int v1 ; int v2 ; } main() (~) A v1 = 100;	→ Exam0140.clas	→ S	상속코드 meth xam0140.class >: A class > 4번 JVM stack main()	메모리에 배치 vod arad v12번 v1 = 100 >> 5/6번 v2 Heap V2 = 200> 12번		2. main() 호출 3.main()의 로컬번 4. A.class 를 메모	수 생성 리에 로딩								
*플레스 로딩, 스타 5번 7번	6. main() 호출 JVM으로 작정 (대략 필드 및 언스런스 필드 생성 class Exam0140 class A(static int v1 : int v2 :	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬번	수 생성 리에 로딩								
*물래스 로딩, 스타 5번 7번 10번	S. main() 호출 JVM으로 작업 6 대학 필드 및 인스턴스 필드 생성 class Exam0140 class A { static int v1; int v2; } main() (~) A v1 = 100; A p = new A(); p v2 = 200;	→ Exam0140.clas	→ S	상속코드 meth xam0140.class >: A class > 4번 JVM stack main()	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬번 4. A.class 를 메모	수 생성 리에 로딩								
*물래스 로딩, 스타 5번 7번 10번	6. main() 호출 JVM으로 작업 (대략 필드 및 언스템스 필드 생성 class Exam0140 class A { static int vt ; int vz ; } main() (~) A.vi = 100. A.vi = 100. p.v2 = 200. A.22 = new A(); p.v2 = 200.	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬번 4. A.class 를 메모 5. v1 스테틱 번수 6. A.v1 = 100 7. new A();	수 생성 리에 로딩								
*물레스 로딩, 스턴 5번 7번 10번 12번	S. main() 호출 JVM으로 작업 6 대학 필드 및 인스턴스 필드 생성 class Exam0140 class A { static int v1; int v2; } main() (~) A v1 = 100; A p = new A(); p v2 = 200;	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
*물래스 로딩, 스타 5번 7번 10번	6. main() 호출 JVM으로 작업 (대략 필드 및 언스템스 필드 생성 class Exam0140 class A { static int vt ; int vz ; } main() (~) A.vi = 100. A.vi = 100. p.v2 = 200. A.22 = new A(); p.v2 = 200.	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬번 4. A.class 를 메모 5. v1 스테틱 번수 6. A.v1 = 100 7. new A();	수 생성 리에 로딩								
*물래스 로딩, 스타 5번 7번 10번	6. main() 호출 JVM으로 작업 (대략 필드 및 언스템스 필드 생성 class Exam0140 class A { static int vt ; int vz ; } main() (~) A.vi = 100. A.vi = 100. p.v2 = 200. A.22 = new A(); p.v2 = 200.	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
*물래스 로딩, 스타 5번 7번 10번	6. main() 호출 JVM으로 작업 (대략 필드 및 언스템스 필드 생성 class Exam0140 class A { static int vt ; int vz ; } main() (~) A.vi = 100. A.vi = 100. p.v2 = 200. A.22 = new A(); p.v2 = 200.	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
"웹계스 로딩, 스타 S번 7번 10번 15번	6. main() 호출 JVM으로 작정 (대략 필드 및 언스템스 필드 생성 class Exam0140 Class A { static int vt ; int vz ; } main() (~) A vt = 1000. A vt = 1000. A vz = 2000. A v2 = 2000. A v2 = 300.	→ Exam0140.clas Actass xam\$0140.Acta	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·빨래스 로딩, 스턴 5번 7번 10번 12번 15번	E. main() 호텔 JVM으로 작업 6 대학 필드 및 인스턴스 필드 생성 class Exam0140 class Exam0140 static int v1; int v2; j) main() (~) A v1 = 100; A p = new A(); p v2 = 200; A p2 = new A(); p 2 v2 = 300; Method Area	→ Exam0140.clas	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	메모리에 배치 V1=100>>5/6번 V2 Heap V2=300>12번 V2=300>13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 모딩, 스타 5번 7번 10번 15번	S. main() 호출 JVM으로 작업 (대략 필드 및 언스런스 필드 생성 class Exam0140 class A { static int v1 ; int v2 ; J	→ Exam0140.clas Actass xam\$0140.Acta	→ S	상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 모딩, 스타 5번 7번 10번 15번	E. main() 호텔 JVM으로 작업 6 대학 필드 및 인스턴스 필드 생성 class Exam0140 class Exam0140 static int v1; int v2; j) main() (~) A v1 = 100; A p = new A(); p v2 = 200; A p2 = new A(); p 2 v2 = 300; Method Area	→ Exam0140.clas A.etaes xam\$0140.Acla		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 포딩, 스턴 5번 7번 10번 12번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스템스 필드 생생 class Exam0140 class A { static int v1 ; int v2 ; Jh Av1 = 100; A p = new A(); p.v2 = 200; A p2 = new A(); p.v2 = 300; Method Area 1. Exam0140 class 실행 2. main() 행사도를 호출	→ Exam0140.clas Actass xam\$0140.Acta		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
"웹계스 모임, 스타 S번 7번 10번 13번 15번	S. main() 호출 JVM으로 작업 (대략 필드 및 언스턴스 필드 생성 class Exam0140 class A{ static int v1 : int v2 : } main() (~) Av1 = 100; Av2 = 200; Av2 = 200; Av2 = 300; Method Area 1. Exam0140.class 실행 2. main() 에서드를 호출 4.A.class	→ Exam0140.clas A.etaes xam\$0140.Acla		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 포팅, 스턴 5번 7번 10번 12번 15번	S. main() 호텔 JVM으로 작업 (대략 필드 및 언스턴스 필드 생성 class Exam0140 diass A { static int v1 : int v2 : j main() (~) A v1 = 100; A p = new A(); p. v2 = 200; A p2 = new A(); p. v2 = 300; Method Area 1. Exam0140.class 실행 2. main() 명 A 드를 호출 4.A. class 5. v1 스탠딩 변수 생경	→ Exam0140.clas A.etaes xam\$0140.Acla		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 로딩, 스타 S번 7번 10번 12번 15번	S. main() 호출 JVM으로 작업 (대략 필드 및 언스템스 필드 생성 class Exam0140 class A { static int vt ; int vz ; } main() (-) Avt = 100. Ay = new A(); pvz = 200. Ap 2 = new A(); pvz = 300. Method Area 1.Exam0140 class 실행 2. main() 매서드를 호출 4.A. class 5.vt 스테틱 변수 생성 6. Avt = 100. 실행 6. Avt = 100. 실행	→ Exam0140.clas A.etaes xam\$0140.Acla		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	08 보건에 해치 ood arad 1/2世 v1 = 100 > 568번 v2 = 300 > 13번 V2 = 300 > 13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 로딩, 스타 S번 7번 10번 12번 15번	S. main() 호텔 JVM으로 작업 (대략 필드 및 언스턴스 필드 생성 class Exam0140 diass A { static int v1 : int v2 : j main() (~) A v1 = 100; A p = new A(); p. v2 = 200; A p2 = new A(); p. v2 = 300; Method Area 1. Exam0140.class 실행 2. main() 명 A 드를 호출 4.A. class 5. v1 스탠딩 변수 생경	→ Exam0140.clas A-class - xam50140.Acla JVM stack 3. main()의 生音		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보건에 해치 제 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기 기		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·웹계스 포팅, 스타 5번 7번 10번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스런스 필드 생성 class Exam0140 Class A { static int v1 : int v2 : J	→ Exam0140.clas A.etaes xam\$0140.Acla		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 모인에 배치 od arad 1/2번 v1 = 100 >>56번 v2 Heap V2 = 200 > 13번 V2 = 300 > 13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 모딩, 스턴 7번 7번 10번 12번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스런스 필드 생성 class Exam0140 class A { static int v1 : int v2 : } main() (~) Av1 = 100; Ay = new A(); pv2 = 200; Ap2 = new A(); pv2 = 300; Method Area 1.Exam0140 class 실행 2. main() 행서도를 호출 4.A class 5v1 스탠딩 변수 생성 6. Av1 = 100; 실행 7. Ap = new A(); 10; pv2 = 200;	→ Exam0140.clas A-class - xam50140.Acla JVM stack 3. main()의 生音		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	08 보건에 해치 ood arad 1/2世 v1 = 100 > 568번 v2 = 300 > 13번 V2 = 300 > 13번		2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 모딩, 스턴 7번 7번 10번 12번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스런스 필드 생성 class Exam0140 Class A { static int v1 : int v2 : J	→ Exam0140.clas A-class - xam50140.Acla JVM stack 3. main()의 生音		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보고 이 배치	5 dd	2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 모딩, 스턴 7번 7번 10번 12번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스런스 필드 생성 class Exam0140 class A { static int v1 : int v2 : } main() (~) Av1 = 100; Ay = new A(); pv2 = 200; Ap2 = new A(); pv2 = 300; Method Area 1.Exam0140 class 실행 2. main() 행서도를 호출 4.A class 5v1 스탠딩 변수 생성 6. Av1 = 100; 실행 7. Ap = new A(); 10; pv2 = 200;	→ Exam0140.clas A.class xamS0140.Acis xamS0140.Acis JVM stack 3. main()의 至意		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 모인에 배치 od arad 1/2번 v1 = 100 >>56번 v2 Heap V2 = 200 > 13번 V2 = 300 > 13번	5 dd	2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 포팅, 스턴 5번 7번 10번 12번 15번	S. main() 호텔 JVM으로 작업 (대략 필드 및 언스턴스 필드 생성 class Exam0140 Class A { static int v1 ; int v2 ; j main() (~) A v1 = 100; A p = new A(); p. v2 = 200; A p2 = new A(); p. v2 = 300; Method Area 1. Exam0140.class 실행 2. main() 명 AC를 효율 4.A. class v3 A p4 = new A(); 10; 10; 10; 10	→ Exam0140.clas A-class - xam50140.Acla JVM stack 3. main()의 生音		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	0년 오건에 배치 od arad 172번 17 = 100 > >56번 17 = 20 > 12번 17 = 20 > 12번 17 = 20 > 13번 18 = 20 > 13번	5 dd	2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								
·텔레스 포팅, 스턴 5번 7번 10번 12번 15번	S. main() 호출 JVM으로 작경 (대략 필드 및 언스런스 필드 생성 class Exam0140 class A { static int v1 : int v2 : } main() (~) Av1 = 100; Ay = new A(); pv2 = 200; Ap2 = new A(); pv2 = 300; Method Area 1.Exam0140 class 실행 2. main() 행서도를 호출 4.A class 5v1 스탠딩 변수 생성 6. Av1 = 100; 실행 7. Ap = new A(); 10; pv2 = 200;	→ Exam0140.clas A.class xamS0140.Acis xamS0140.Acis JVM stack 3. main()의 至意		상속코드 Meth xam0140.class >' A class > 4번 JVM stack main() p = 111 > 9번	00 보고 이 배치	5 dd	2. main() 호출 3.main()의 로컬변 4. A.class 를 메모 5. v1 스테틱 변수 6. A.v1 = 100 7. new A(); 8.인스턴스 생성	수 생성 리에 로딩								

A CADISS 1 1 2 2 1 1 2 2 2 1 2 2	
S (**) 스타인 연구 전용 10 (**2 - 200): (*** (***) 보이 전용 *** (***)	
6 M 스테트 변수 경험	
7 A S = rem A(S) 10 Ref 2 - 20 0 11 A Ref 2 - rem A(S) 11 전 2 - 20 0 12 A Ref 2 - rem A(S) 15 전 2 - 20 0 16 A Ref 2 - rem A(S) 16 전 2 - 20 0 17 A Ref 2 - rem A(S) 18 전 2 - 20 0 19 A Ref 2 - rem A(S) 19 A Ref 2 - rem A(
7 (A) = new A(); 10 (A) = 200 (A); 11 (A) = 200 (A); 15 (A) = 200 (A); 16 (A) = 200 (A); 17 (A) = 200 (A); 18 (A) = 200 (A); 19 (A) = 200 (A); 19 (A) = 200 (A); 19 (A) = 200 (A); 10 (A) = 200 (A); 10 (A) = 200 (A); 10 (A) = 200 (A); 11 (A) = 200 (A); 11 (A) = 200 (A); 11 (A) = 200 (A); 12 (A) = 200 (A); 13 (A) = 200 (A); 14 (A) = 200 (A); 15 (A) = 200 (A); 16 (A) = 200 (A); 17 (A) = 200 (A); 18 (A) = 200 (A); 18 (A) = 200 (A); 19 (A) = 200 (A); 19 (A) = 200 (A); 10 (A) = 200 (A); 11 (A) = 200 (A); 12 (A) = 200 (A); 13 (A) = 200 (A); 14 (A) = 200 (A); 15 (A) = 200 (A); 16 (A) = 200 (A); 17 (A) = 200 (A); 18 (A) = 200 (A); 19 (A) = 200 (A); 10 (A) = 200 (A); 11 (A) = 200 (A); 11 (A) = 200 (A); 12 (A) = 200 (A); 13 (A) = 200 (A); 14 (A) = 200 (A); 15 (A) = 200 (A); 16 (A) = 200 (A); 17 (A) = 200 (A); 18 (A) = 200 (A); 19 (A) = 200 (A); 10 (A) = 200	
15 [22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 = 30% 15 [22 22 22 22 = 30% 15 [22 22 22 22 = 30% 15 [22 22 22 22 22 = 30% 15 [22 22 22 22 22 22 22 = 30% 15 [22 22 22 22 22 22 22 22 22 22 22 22 22	
15 (24/26 300는 19 10 10 10 10 10 10 10 10 10 10 10 10 10	
Heap Note	
Method Area July stack Heap new Calculator	
Method Area JVM stack Heap new Calculator() Marin() args ct c2 result plus() (-) minus() (
Method Area	
Method Area JVM stack Heap nain() new Calculater() New C	
Example 20.class main() mew Calculater() minus() minus()	
Example 20.class main() mew Calculater() minus() minus()	
main() -)	
Calculator.class Part Par	
Fesult	
Fesult plus() (-)	
plus((-) minus()(-) 에서도는 Method Area에 있다.	
minus() (-) static世中と の最空意 3 章 7 地。 static void	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 호흡할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 호흡할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 요즘할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 요즘할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 요즘할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 요즘할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 호흡할 수 있음.	
static void m10 (system.out.printin("m10") instence 변수는 주소가 있어야 요즘말 수 있음.	
static void m10 (system.out.printin("n10") instence 번수는 주소가 있어야 호흡할 수 있음.	
(system.out.printfin("m1()") instence 변수는 주소가 있어야 효용할 수 있음.	
instence 번수는 주소가 있어야 호흡할 수 있음.	
void m2()	
A objf = new Alj;	
obj1.m2();	
인스런스와 생성자	
2. 인스템스 생성 후 그 주소를 가지고 기본 생성자를 오늘한다.	
Score s new Score ():	
3.번지수가 들어감. 1. Score 설계도에 따라 인스턴스 변수를 Heap에 지장	
인스런스 변수 생성과 생성자 호흡이 끝나면, 인스런스 주소를 리린한다.	
Score s1 = new Score s1 200 name kor eng math sum aver	
200	
Score s1 = new Score("8248", 100, 100, ···); s1 200 name kor eng math sum aver 200 8248 100 100 90 290 96. ···	
Score s1 = new Score("음길등", 100, 100,); s1 200 name kor eng math sum aver 200 음길등 100 100 90 290 96	
Score s2 = new Score("유편한", 100, 100,); s1 300 name kor eng math sum aver	
Score s2 = new Score("유급순", 100, 100,); s1 300 name kor eng math sum aver 300 유급순 100 100 90 290 96	
Score s2 = new Score("유권순", 100, 100,); s1 300 name kor eng math sum aver 300 유권순 100 100 90 290 96 인스턴스를 사용하기 전에 유요한 값으로 초기회시켜주는 것.	
인스턴스를 사용하기 전에 유효한 강으로 초기화시켜주는 것.	
인스턴스를 사용하기 전에 유효한 같으로 초기화시켜주는 것. 1 Exam0691 java	
인스템스를 사용하기 전에 유효한 강으로 초기화시켜주는것. 1 Exam0691 Java 2 main() 후호 1 () 유리는 나전에 크겠스 문의	
인스템스를 사용하기 전에 유효한 강으로 초기화시켜주는것. 1 Exam0691 Java 2 main() 후호 1 () 유리는 나전에 크겠스 문의	
인스템스를 사용하기 전에 유효한 강으로 초기회사가주는 것. 1 Examble91 java	
인스템스를 사용하기 전에 유효한 강으로 초기화시켜주는것. 1 Exam0691 Java 2 main() 후호 1 () 유리는 나전에 크겠스 문의	
인스턴스를 사용하기 전에 유효한 강으로 출기용시켜 주는 것. 1	
인스턴스를 사용하기 전에 유효한 같으로 초기회사가주는 것. 1	
인스턴스를 사용하기 전에 유효한 같으로 휴기용시켜주는것. 1	
인스턴스를 사용하기 전에 유효한 강으로 출기하시켜주는것. 1	
인스런스를 사용하기 전략 유효한 같으로 출기용시켜주는 것. 1	
인스턴스를 사용하기 전에 유효한 같으로 휴기화시켜주는것. 1	
인스런스를 사용하기 전략 유효한 같으로 출기용시켜주는 것. 1	
인스템스를 사용하기 전에 유효한 같으로 초기용시키주는 것. 1	
인스턴스를 사용하기 전에 유효한 같으로 초기용시켜주는 것. 1 Exam0691,lava 2 main() 호를 3 Aclass a 4 Aclass a 5 Aclass a = 7 6 B.class b 7 B.class b 8 B.class b	
인스턴스를 사용하기 전에 유효한 같으로 초기용시켜 주는 것. 1	

class A { A() { [■从] }	· ·	
1	이시쉬스 추가하 볼륨 => 컴퓨입러가 모든 생성파의 안 부부에 불사하다.	
	\ \(\) 인스턴스 초기화 블록 → 합파일러가 모든 생성자의 앞 부분에 복사한다. } >> 이러 생성자에 공룡으로 들어갈 코드가 있다면 인스턴스 초기화 블록에 담아두라.	
, A(m+a) { [복사]		
}		
A(String s) { [복사]		
}		
,		
스태틱 초기화 블럭은 할		
스대막 소기와 발탁은 말	에서 보는 사람들이 되었다. 사람들이 보고 있는 사람들이 되었다. 사람들이 보고 있는 사람들이 되었다. 사람들이 되었다. 사람들이 보고 있는 사람들이 되었다.	
인스턴스 조기와 들썩은	CM가 영영사 보으로 내용한다.	
연수 조기화 운상 = 스배	목 불막이 함쳐지고, 인스템스 불막은 생성자와 함쳐진다.	
기능을 확장하는 방법 : 1. 기존클레스를 변경		
Car	>기존 소스를 변경하면 기존의 Car를 사용해서 만든 프로그램에도 영향	
model	>심각한 오류가 발생할 수 있다.	
maker	>쓰지 않는 코드가 누려되는 문제가 생긴다.	
capacity		
car()		
car(~생성자)		
run()		
sunroof()		
auto()		
Car(선루프 생성자 추가)		
>기존 코드에 새 기능을 위한 코드 추가<		

-									

-									

1									
1									
l									
l									
l									
l									

	1									
	l									
	l									
	1									
	1									
	l									
	l									
	1									
	l									