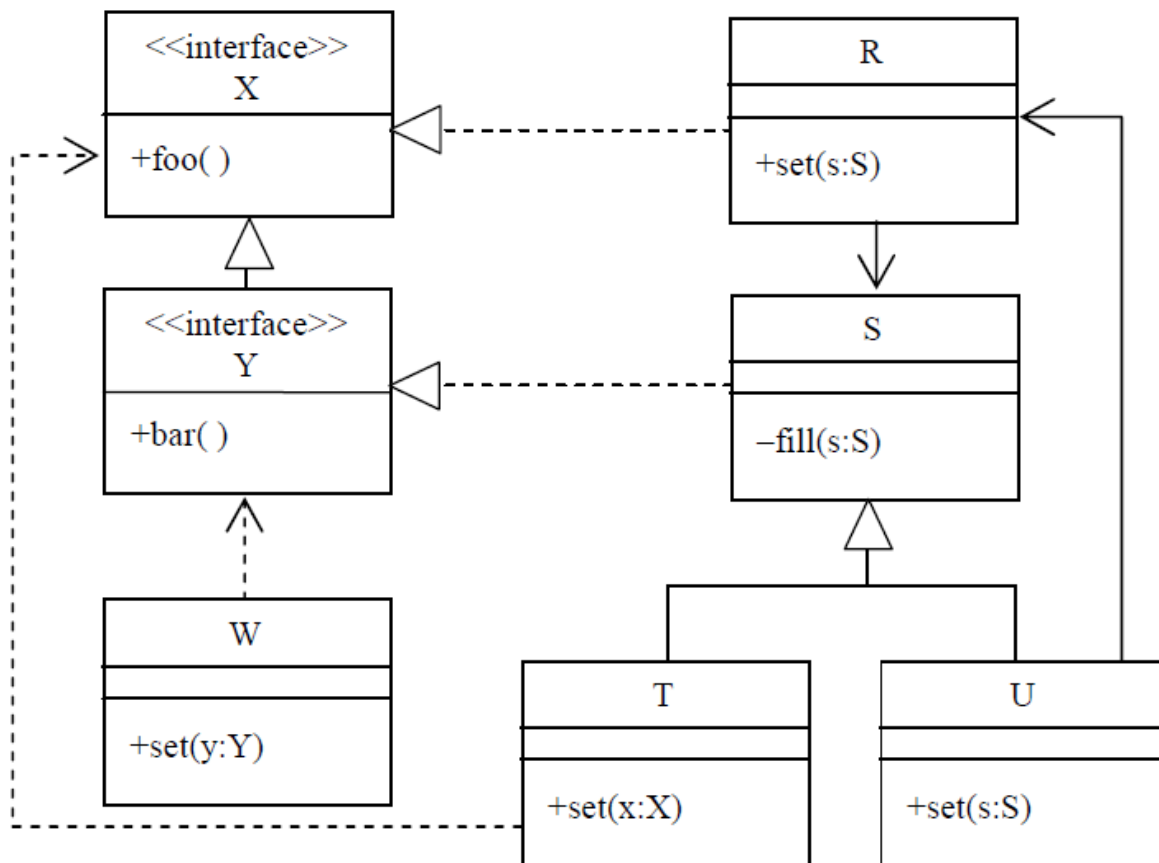


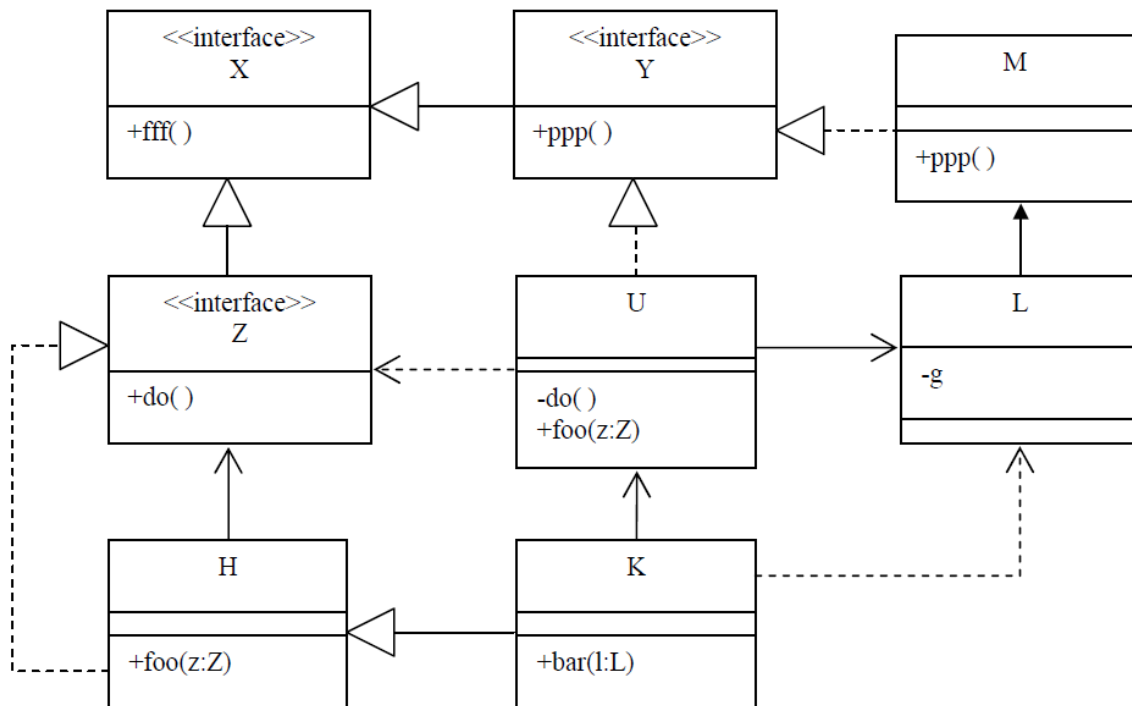
UML class diagram – A,B,C,D,E feladattípus, feladatok

2008.01.08 – 1. Feladat



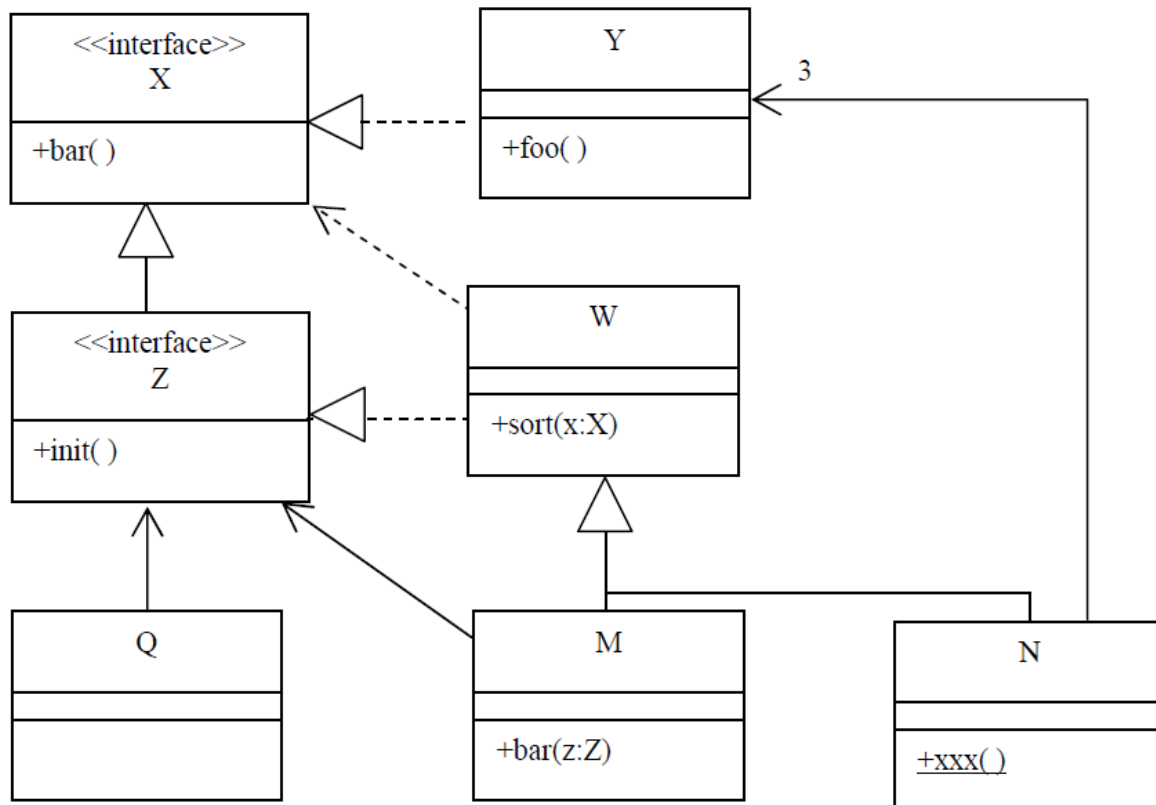
Válasz	Állítás	1.	2.	köv.
	Y bárhol helyettesíthető W -vel, mert W az Y leszármazottja.			
	U bármikor lehet T.set(x:X) paramétere, mert U megvalósítja az X interfészt.			
	R meghívhatja saját set(s:S) metódusából egy W set(y:Y) metódusát, mert S megvalósítja Y -t.			
	S fill(s:S) metódusa nem kaphat paraméterül T -t, mert a metódus protected.			
	T megvalósítja az X interfészt, mert T az R leszármazottja.			
	T pontosan egy U -t tartalmazhat, mert csak egy közvetlen ősök van.			
	T bárhol helyettesíthető U -val, mert egyforma az interfészük.			
	U meghívhatja S fill(s:S) metódusát, mert R asszociációban van S -sel.			

2008.01.15 – 1. Feladat



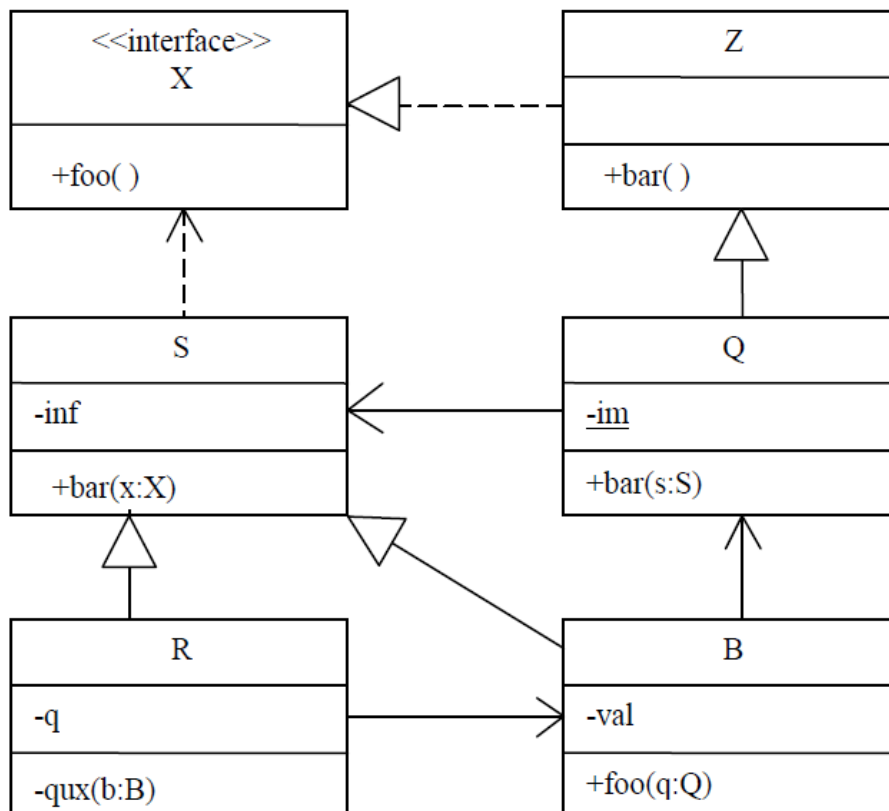
Válasz	Állítás	1.	2.	köv.
	H bárhol helyettesítheti U-t, mert mindketten megvalósítják az X interfészt.			
	H foo(z:Z) metódusa meghívható egy U-val, mert U megvalósítja az Y interfészt.			
	U nem hívhatja meg K bar(l:L) metódusát, mert K nem függ L-től.			
	K implementálja a Z interfészt, ezért K meghívhatja U do() metódusát.			
	K nem hozhat létre L objektumot, mert az L g attribútuma privát.			
	U foo(z:Z) metódusa nem hívhatja meg egy paraméterül kapott H foo(z:Z) metódusát, mert U nem implementálja a Z interfészt.			
	K bárhol helyettesíthető U-val, mert K az U leszármazottja.			
	L nem ismeri az Y interfészt, ezért L nem hívhatja meg M ppp() metódusát.			

2008.01.22 – 1. Feladat



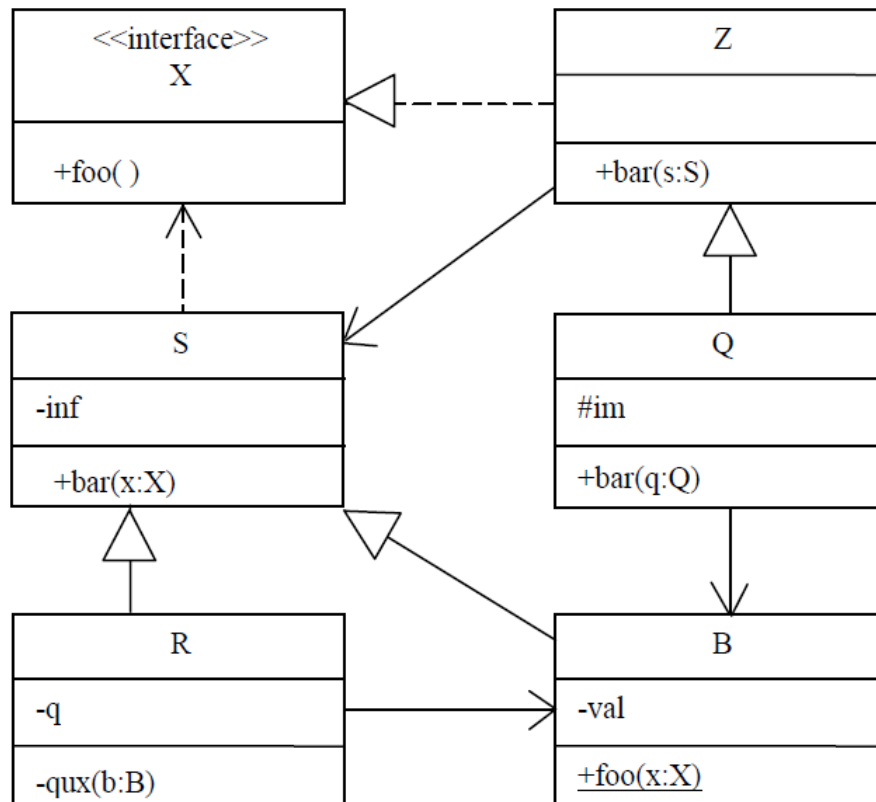
Válasz	Állítás	1.	2.	köv.
	Y helyettesíthető W -vel, mert mindketten megvalósítják az X interfészt.			
	N meghívhatja Y foo() metódusát, mert N megvalósítja a Z interfészt.			
	M bar(z:Z) metódusa kaphat paraméterül N objektumot, mert van közös ősök.			
	W nem helyettesíthető M -mel, mert W -nek nincs bar(z:Z) szignatúrájú metódusa.			
	Q helyettesíthető M -mel, mert mindkettő megvalósítja a Z interfészt.			
	N xxx() metódusa meghívható a W osztály sort(x:X) metódusából, mert az N.xxx() statikus.			
	W sort(x:X) metódusa meghívhatja egy paraméterül kapott Y objektum bar() metódusát, mert W -nek is van ugyanilyen szignatúrájú metódusa.			
	M -nek és N -nek különböző az interfésze, mert N nem valósítja meg X -t.			

2008.05.27 – 1. Feladat



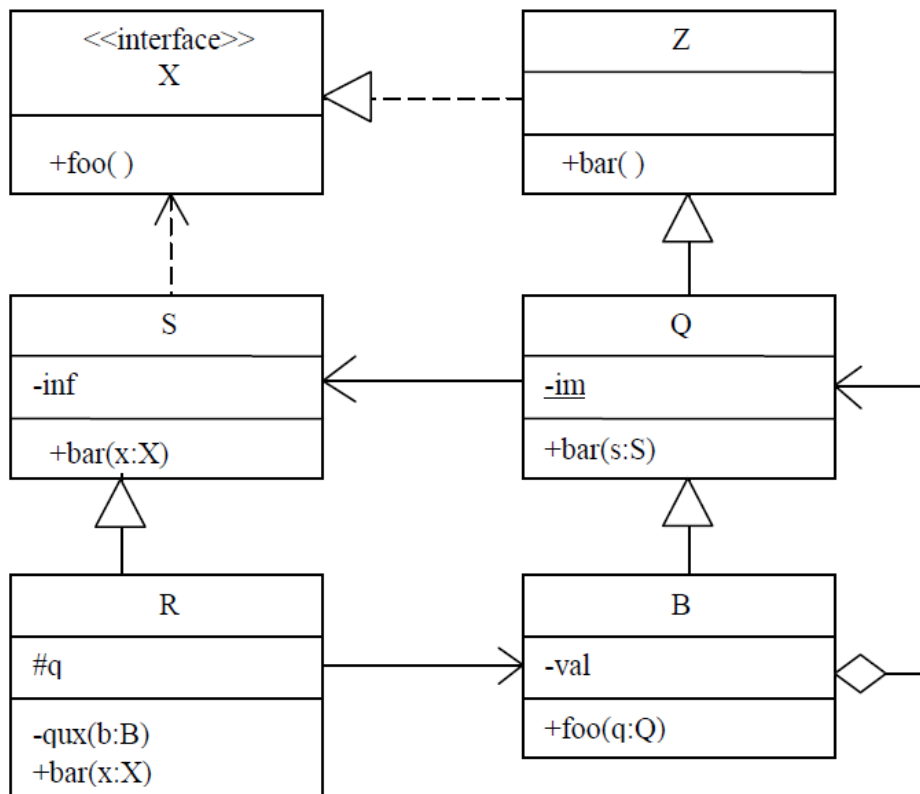
Válasz	Állítás	1.	2.	köv.
	S helyettesíthető Q-val, mert Q az S leszármazottja.			
	S helyettesíthető B-vel, mert B megvalósítja az X interfészt.			
	R átadható paraméterül Q bar(s:S) metódusának, mert Q és S interfésze megegyezik.			
	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát.			
	Q meghívhatja S bar(x:X) metódusát, mert mindketten megvalósítják az X interfészt.			
	B interfésze tartalmazza bar(s:S) metódust, mert a metódus statikus.			
	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum konstans.			
	B-nek nincs bar(x:X) metódusa, ezért nem függ az X interfésztől.			

2008.06.10 – 1. Feladat



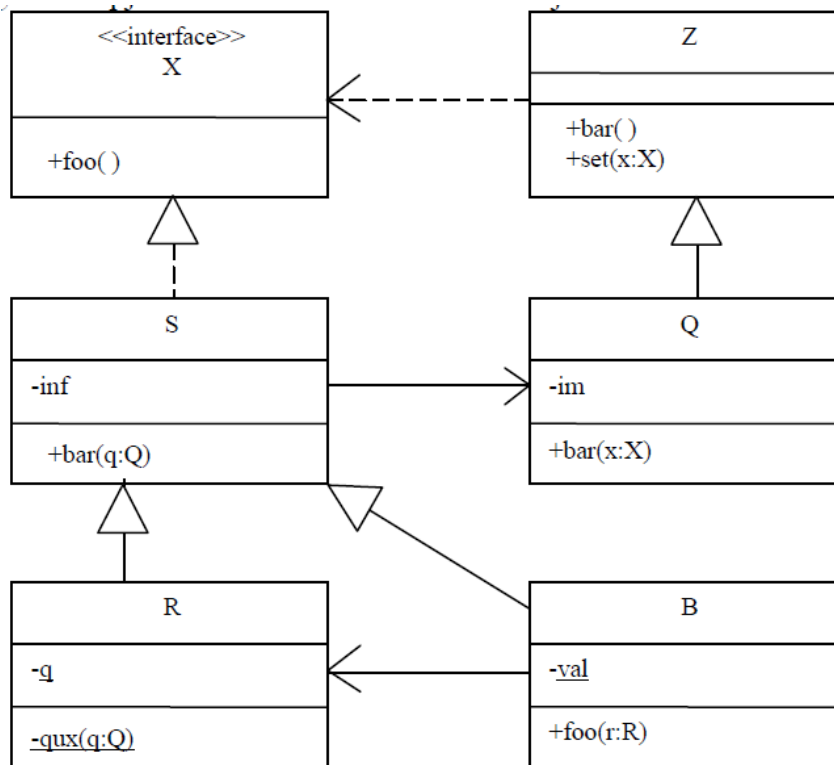
Válasz	Állítás	1.	2.	köv.
	R helyettesíthető S-sel, mert S az R leszármazottja.			
	R helyettesíthető B-vel, mert mindketten megvalósítják az X interfészt.			
	R átadható paraméterül Q bar(s:S) metódusának, mert Q és S interfésze megegyezik.			
	B foo(x:X) metódusa nem látja a val attribútum értékét, mert az attribútum nem statikus.			
	Q meghívhatja S bar(x:X) metódusát, mert mindketten megvalósítják az X interfészt.			
	B interfésze tartalmaz qux(b:B) metódust, mert B-nek van R-rel közös őse.			
	Q bar(s:S) metódusa nem módosíthatja az im attribútumot, mert az attribútum privát.			
	B meghívhatja R qux(b:B) metódusát, mert a metódus paramétere B osztályú.			

2008.06.17 – 1. Feladat



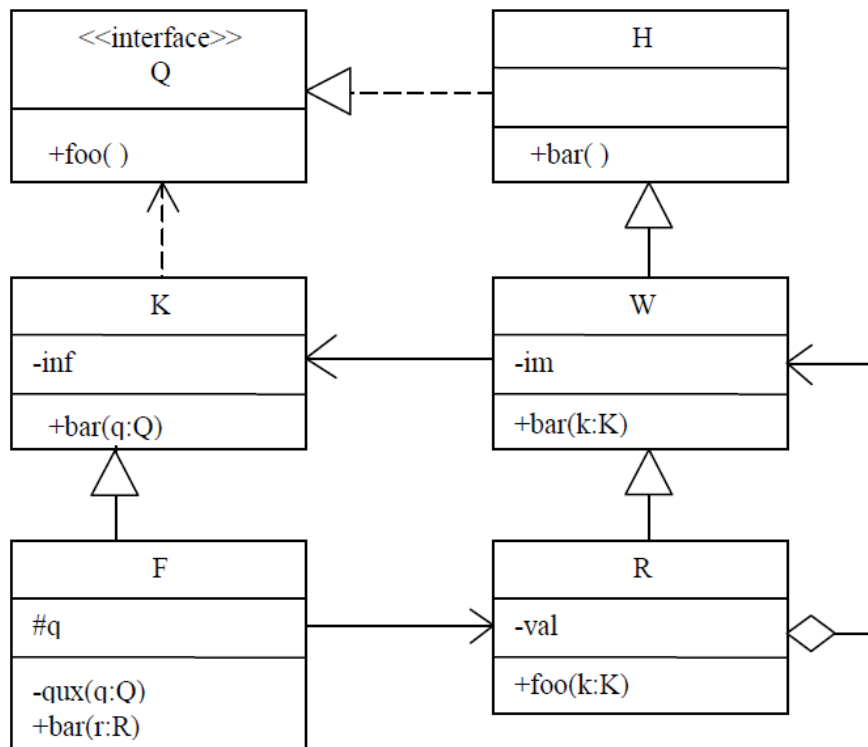
Válasz	Állítás	1.	2.	köv.
	R helyettesíthető B-vel, mert mindketten megvalósítják az X interfészt.			
	Z helyettesíthető B-vel, mert B a Z leszármazottja.			
	Q bar(s:S) metódusa nem módosíthatja Q im attribútumát, mert az attribútum statikus.			
	R qux(b:B) metódusa nem hívhat meg a paraméteren foo() metódust, mert B nem implementálja az X interfészt.			
	S bar(x:X) metódusa meghívhatja egy paraméterül kapott Z bar() metódusát, mert Z megvalósítja az X interfészt.			
	R qux(b:B) metódusa nem módosíthatja a q attribútumot, mert az attribútum privát.			
	B bar(s:S) metódusa nem kaphat paraméterül R objektumot, mert B nem ismeri az R osztályt.			
	S bar(x:X) metódusa kaphat paraméterül R objektumot, mert R függ X-től.			

2009.01.06 – 1. Feladat



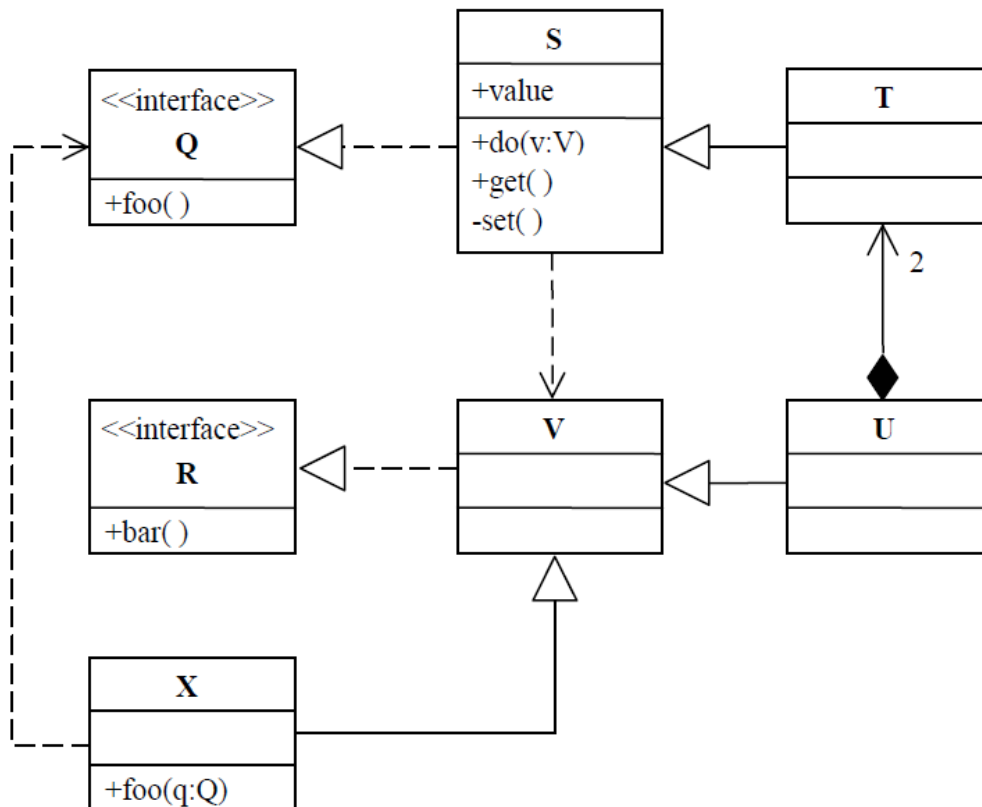
Válasz	Állítás	1.	2.	köv.
	Q bárhol helyettesíthető S -sel, mert S a Q leszármazottja.			
	R qux(q:Q) metódusa nem kaphat paraméterül Z objektumot, mert a metódus absztrakt.			
	B foo(r:R) metódusa nem hívhat meg a paraméterül kapott objektumon foo() metódust, mert R -nek nincs ilyen metódusa.			
	S bar(q:Q) metódusa nem módosíthatja az S inf attribútumát, mert az attribútum konstans.			
	S bar(q:Q) metódusa nem hívhatja meg a paraméterül kapott objektum bar() metódusát, mert a Q osztálynak nincs ilyen szignatúrájú metódusa.			
	Z set(x:X) metódusa nem kaphat paraméterül B objektumot, mert B megvalósítja az X interfészt.			
	B módosíthatja egy Q objektum im attribútumát, mert S függ Q -tól.			
	R és B bárhol felcserélhetők, mert közös az ősük.			

2009.01.13 – 1. Feladat



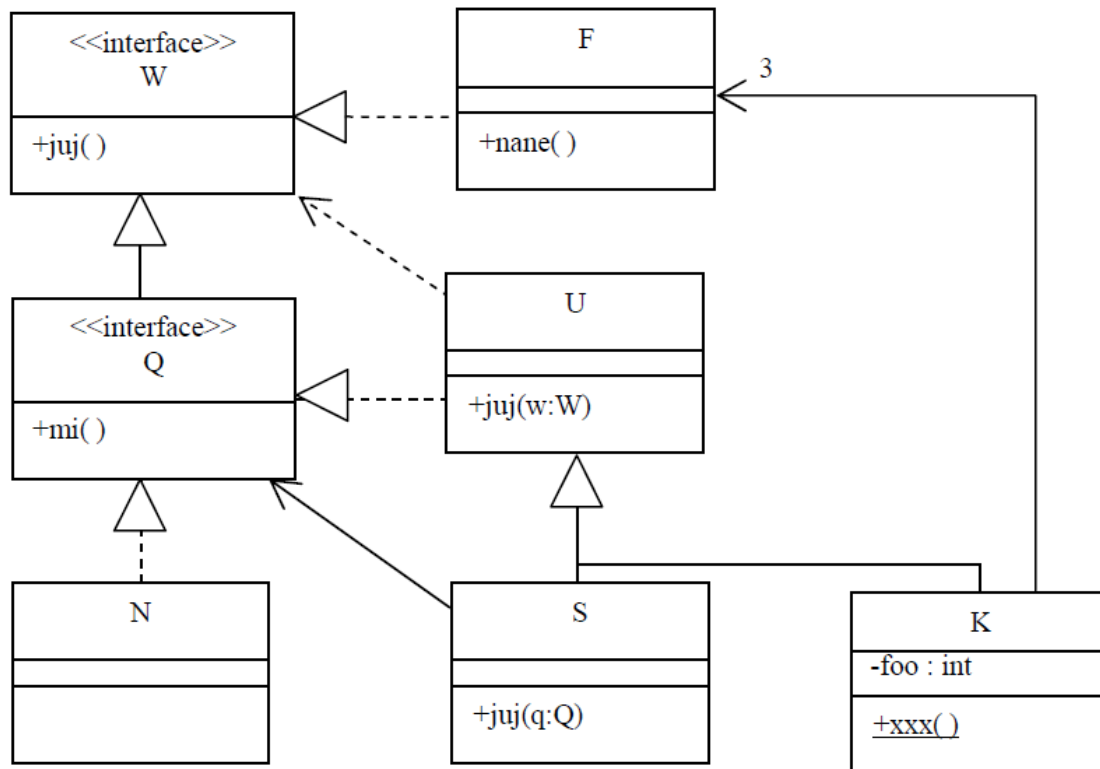
Válasz	Állítás	1.	2.	köv.
	H bármikor helyettesíthető R -rel, mert R a H leszármazottja.			
	W nem módosíthatja K inf attribútumát, mert az attribútum protected.			
	F bar(r:R) metódusa nem hívhatja meg a qux(q:Q) metódust, mert az utóbbi statikus.			
	F qux(q:Q) meghívhatja a bar(r:R) metódust a q paraméterrel, mert az R megvalósítja a Q interfészt.			
	Az ábrán szereplő összes egy-paraméteres bar metódus kaphat R objektumot paraméterül, mert R megvalósítja az összes említett metódust.			
	W bar(k:K) metódusa nem módosíthatja az osztály im attribútumát, mert az attribútum konstans.			
	W rendelkezik foo() szignatúrájú metódussal, mert függ K -től.			
	Egy F objektum meghívhatja saját magával mint paraméterrel egy R foo(k:K) metódusát, mert F a K leszármazottja.			

2009.01.27 – 1. Feladat



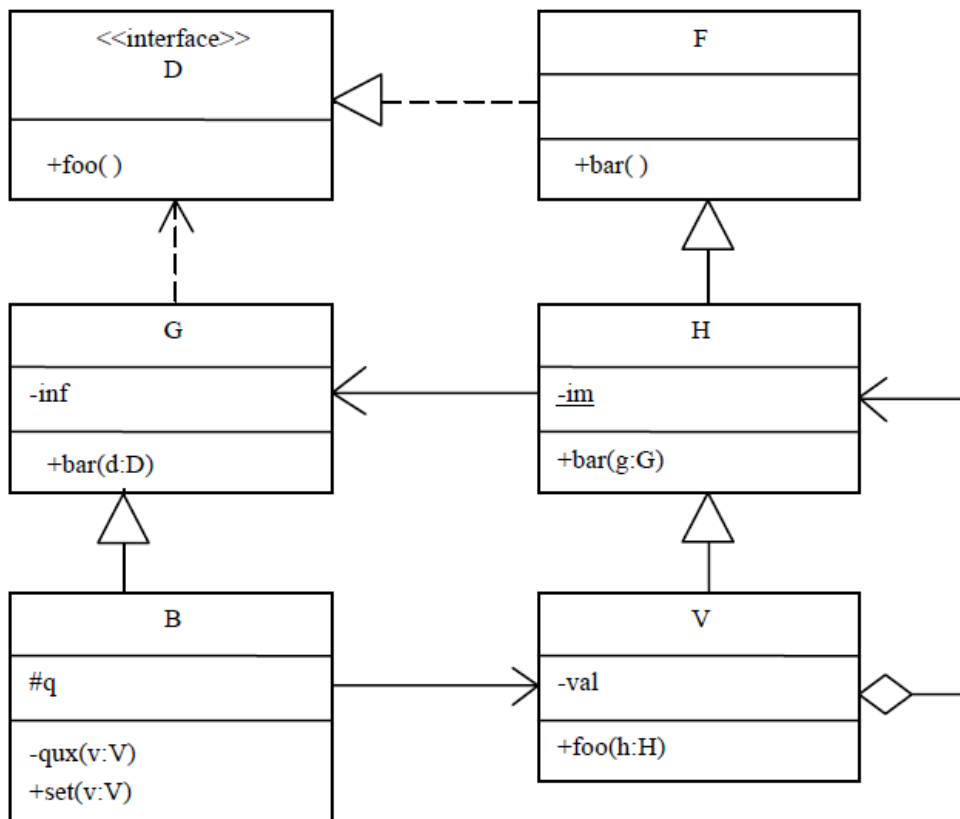
Válasz	Állítás	1.	2.	köv.
	S létrehozhat V osztályú objektumot, mert V függ az S -től.			
	X foo(q:Q) metódusa kaphat paraméterül T -t, mert T megvalósítja a Q interfészt.			
	X foo(q:Q) metódusa meghívhatja a paraméterül kapott S get() metódusát, mert S megvalósítja a Q interfészt.			
	T -ből legalább kétszer annyi példány van, mint U -ból, mert egy T példány nem tartozhat két U -hoz.			
	T meghívhatja U bar() metódusát, mert U -nak van bar () metódusa.			
	X meghívhatja egy Q interfészű objektum foo() metódusát, mert X implementálja Q -t.			
	V helyettesíthető U -val, mert mindketten megvalósítják az R interfészt.			
	S set() metódusa nem módosíthatja a value attribútumot, mert a láthatóságuk különböző.			

2009.05.28 – 1. Feladat



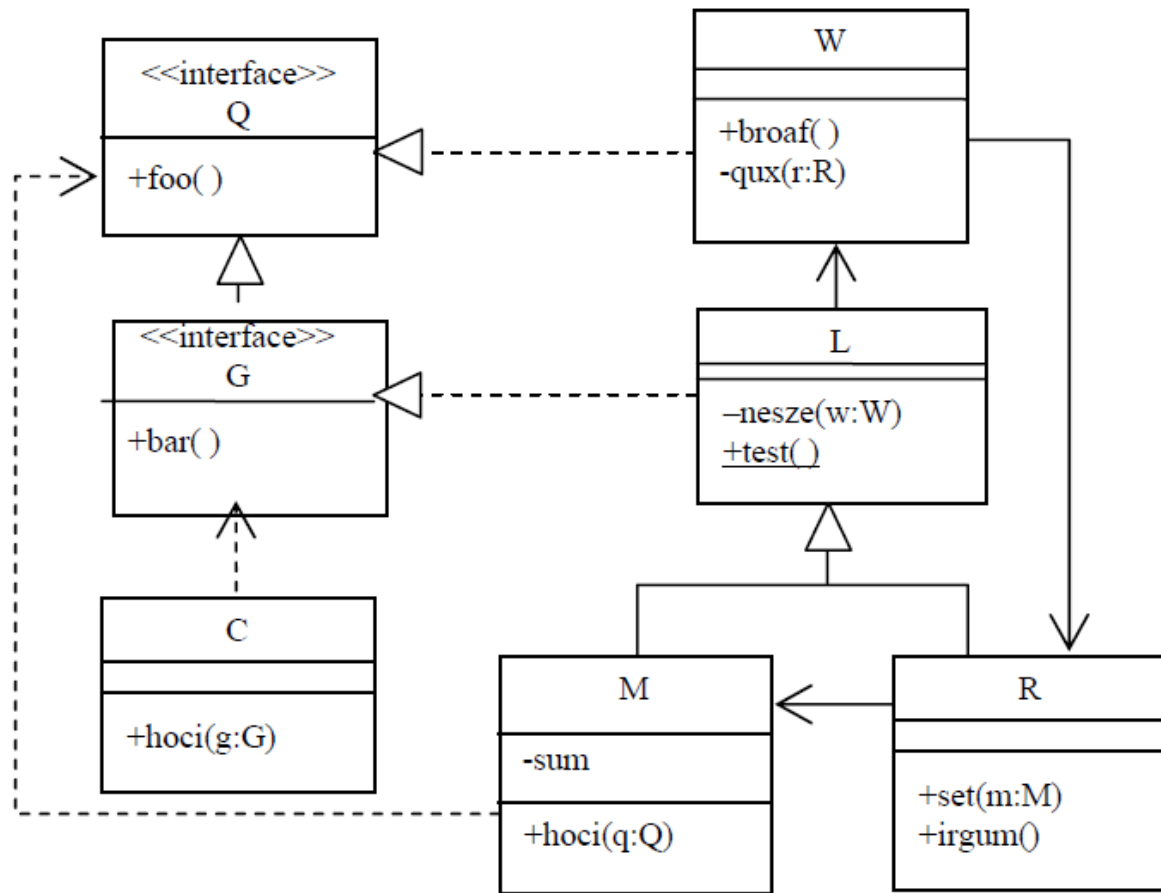
Válasz	Állítás	1.	2.	köv.
	K helyettesíthető S-sel, mert közös ősük U.			
	K juj(w:W) metódusa kaphat paraméterül S-t, mert K az F leszármazottja.			
	K xxx() metódusa módosíthatja bármely K objektum foo attribútumát, mert a metódus statikus.			
	F nem implementálja a juj() metódust, mert nem U leszármazottja.			
	S juj(q:Q) metódusában meghívható egy paraméterül kapott N objektum mi() metódusa, mert N megvalósítja a W interfészt.			
	S-nek nincs juj(w:W) metódusa, mert a juj(q:Q) metódusnak ugyanaz a szignatúrája.			
	F helyettesíthető U-val, mert K mindkettejük leszármazottja.			
	U juj(w:W) metódusából meghívhatjuk egy paraméterül kapott F nane() metódusát, mert F megvalósítja a Q interfészt.			

2009.06.11 – 1. Feladat



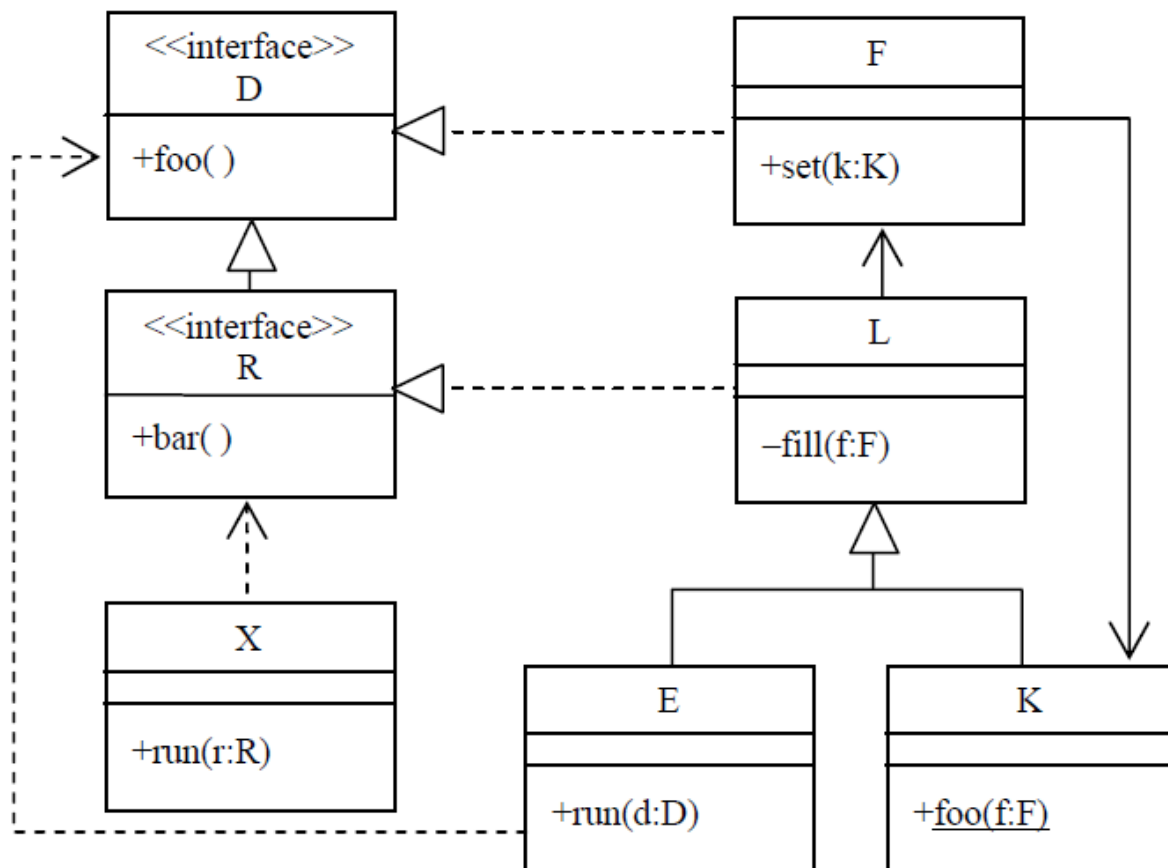
Válasz	Állítás	1.	2.	köv.
	G bar(d:D) metódusa kaphat paraméterül B objektumot, mert G a B leszármazottja.			
	H bar(g:G) metódusa kaphat paraméterül V objektumot, mert V megvalósítja a D interfészt.			
	B qux(v:V) metódusa módosíthatja a paraméter val attribútumát, mert mind a metódus, mind az attribútum privát.			
	H bar(g:G) metódusa nem módosíthatja az im attribútumot, mert az attribútum konstans.			
	B objektum nem hívhatja meg egy V objektum foo() metódusát, mert V -nek nincs ilyen szignatúrájú metódusa.			
	G bar(d:D) metódusa meghívhatja egy paraméterül kapott F objektum bar() metódusát, mert a két metódus azonos szignatúrájú.			
	B set(v:V) metódusa nem módosíthatja a q attribútumot, mert a láthatóságuk különböző.			
	B -nek van foo() szignatúrájú metódusa, mert B megvalósítja a D interfészt.			

2009.06.18 – 1. Feladat



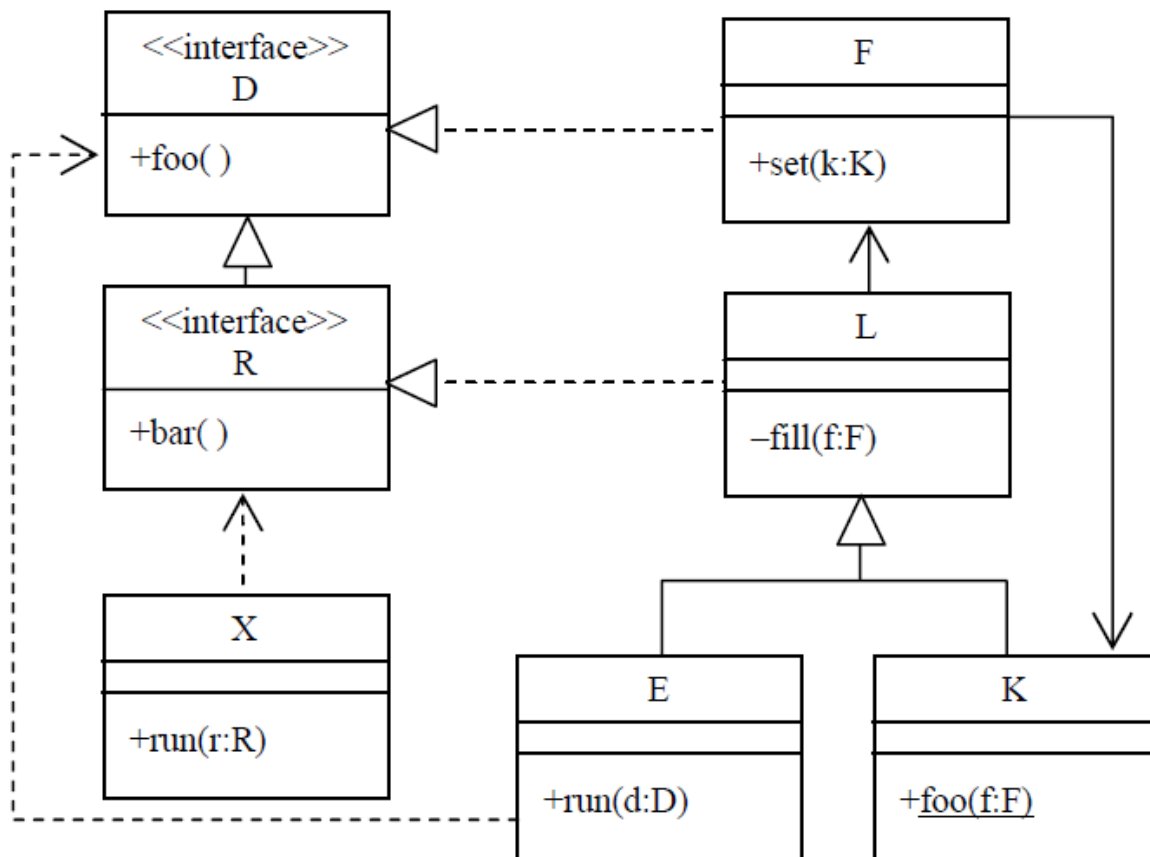
Válasz	Állítás	1.	2.	köv.
	M hoci(q:Q) függvénye meghívhatja egy paraméterül kapott W broaf() metódusát, mert a broaf metódus statikus.			
	R set(m:M) metódusa kaphat paraméterül L objektumot, mert M az L leszármazottja.			
	L nesze(w:W) metódusa meghívhatja a paraméterül kapott objektum qux(r:R) metódusát, mert mindkét metódus privát.			
	W bárhol helyettesíthető L -lel, mert mindketten megvalósítják a Q interfészt.			
	R -nek nincs foo() szignatúrájú metódusa, mert nem valósítja meg a G interfészt.			
	C hoci(g:G) metódusa kaphat paraméterül M objektumot, mert M hoci(q:Q) metódusa is kaphat paraméterül C -t.			
	L nesze(w:W) metódusa nem hívhatja meg a test() metódust, mert a test() statikus.			
	W qux(r:R) metódusából bármikor meghívható a paraméter irgum() metódusa, mert a két osztály nem függ egymástól.			

2010.01.05 (A) – 1. Feladat



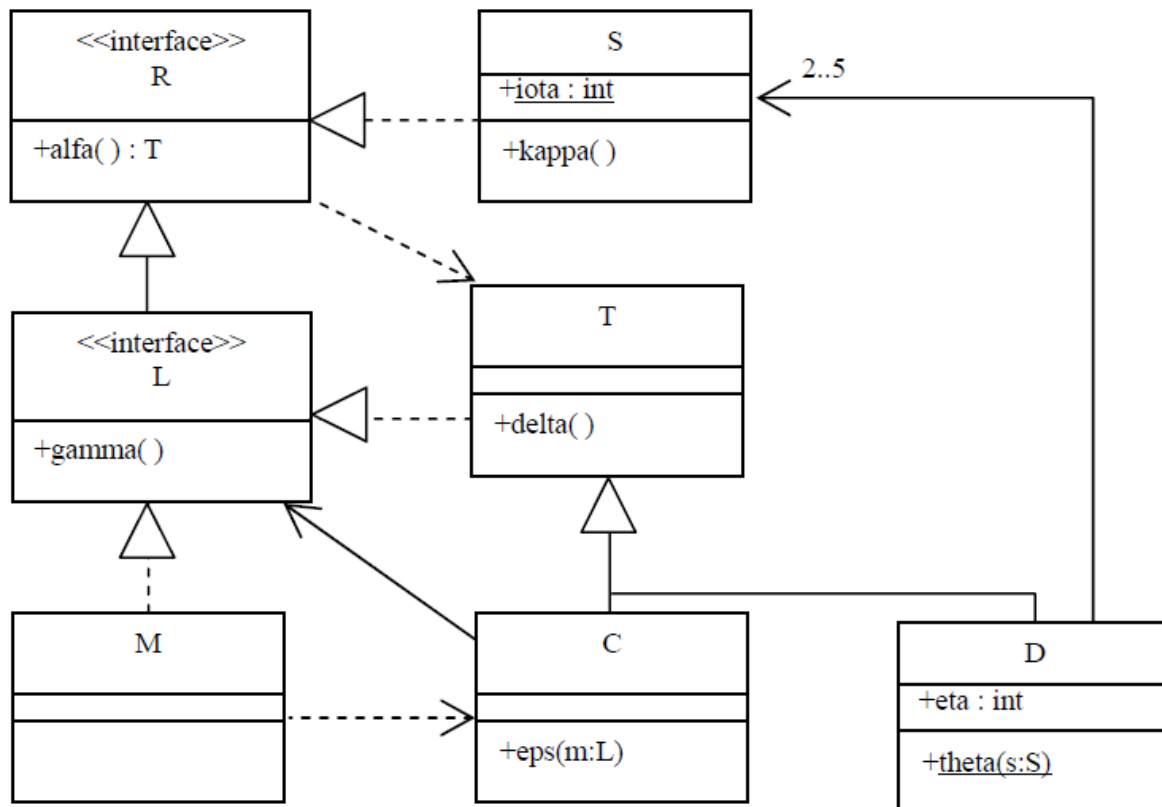
Válasz	Állítás	1.	2.	köv.
	E bárhol helyettesíthető K-val, mert van közös ősük.			
	L bárhol helyettesíthető F-fel, mert mindketten megvalósítják a D interfészt.			
	L nem helyettesíthető E-vel, mert L-nek van privát metódusa.			
	F set(k:K) metódusa meghívhatja egy paraméterül kapott K fill(f:F) metódusát, mert K függ F-től.			
	X run(r:R) metódusa kaphat paraméterül F osztályú objektumot, mert X függ R-től.			
	K-nak nincs foo() szignatúrájú metódusa, mert K-t nem lehet példányosítani.			
	X run(r:R) metódusa nem kaphat paraméterül K objektumot, mert K-nak van statikus metódusa.			
	K foo(f:F) metódusa nem hívhatja meg a paraméter foo() metódusát, mert az utóbbi metódus nem statikus.			

2010.01.05 (B) – 1. Feladat



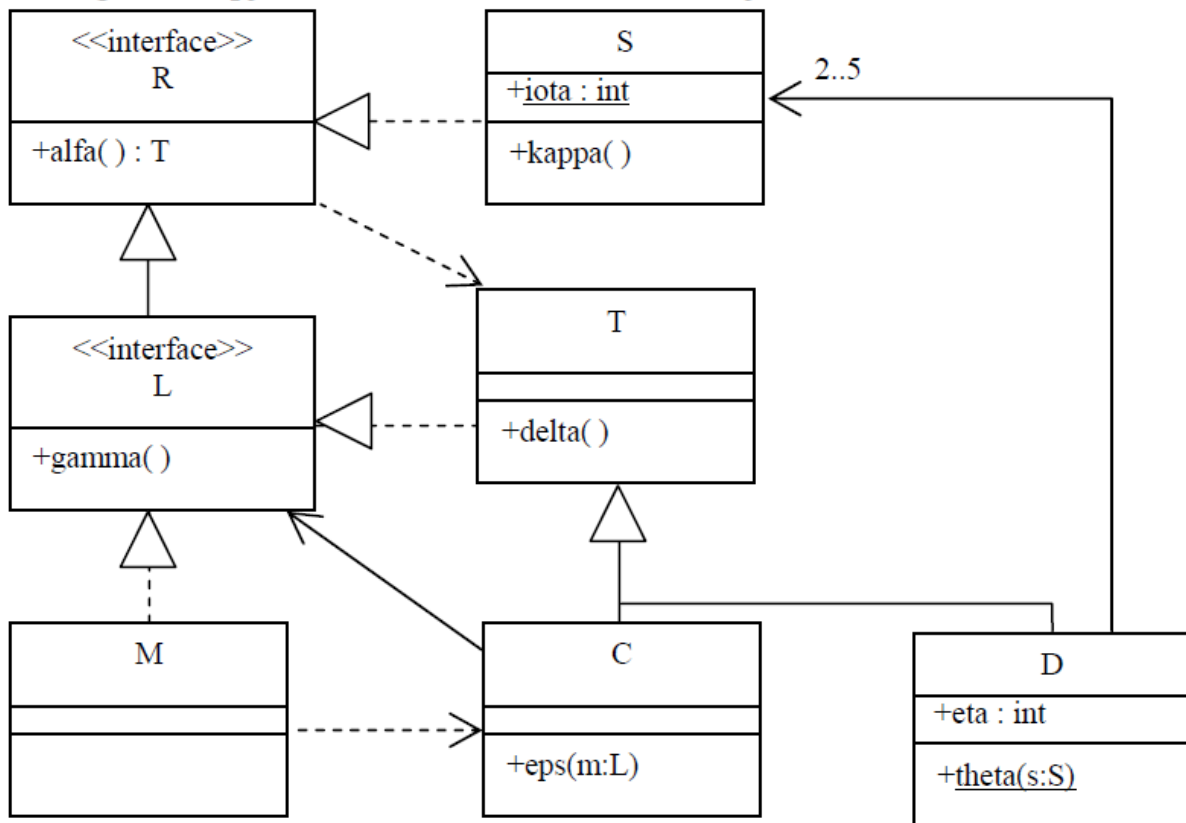
Válasz	Állítás	1.	2.	köv.
	X run(r:R) metódusa kaphat paraméterül F osztályú objektumot, mert X függ R -től.			
	K -nak nincs foo() szignatúrájú metódusa, mert K -t nem lehet példányosítani.			
	L bárhol helyettesíthető F -fel, mert mindketten megvalósítják az R interfészt.			
	L nem helyettesíthető E -vel, mert L -nek van privát metódusa.			
	X run(r:R) metódusa nem kaphat paraméterül K objektumot, mert K -nak van statikus metódusa.			
	K foo(f:F) metódusa nem hívhatja meg a paraméter foo() metódusát, mert az utóbbi metódus nem statikus.			
	E bárhol helyettesíthető K -val, mert van közös ősük.			
	F set(k:K) metódusa nem hívhatja meg egy paraméterül kapott K fill(f:F) metódusát, mert K függ F -től.			

2010.01.12 (A) – 1. Feladat



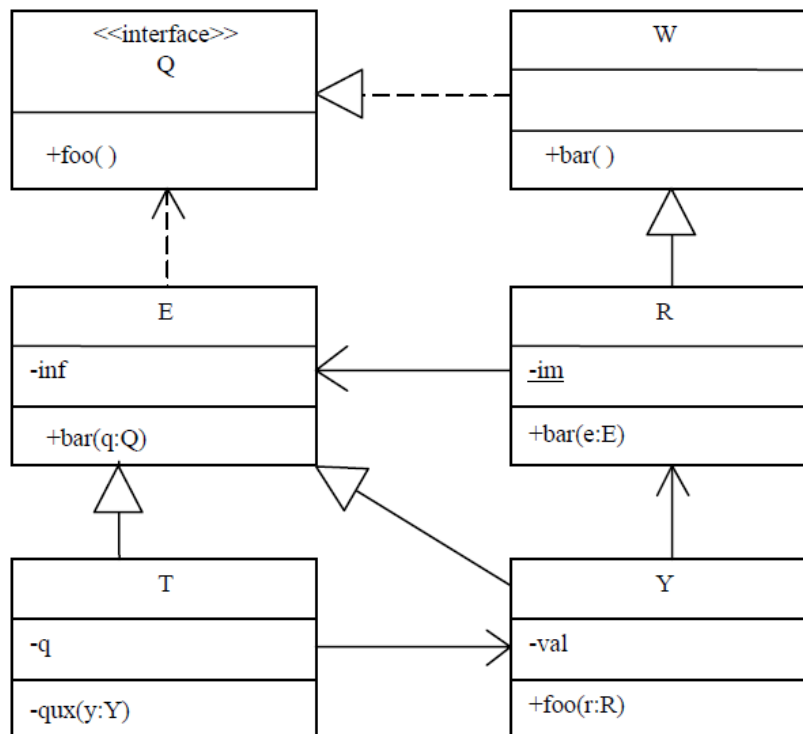
Válasz	Állítás	1.	2.	köv.
	M bárhol helyettesíthető C -vel, mert mindketten megvalósítják az R interfészt.			
	C eps(m:L) metódusa nem hívhatja meg a paraméter gamma() metódusát, mert az utóbbi metódus protected láthatóságú.			
	D theta(s:S) metódusa módosíthatja a paraméter iota attribútumát, mert a theta(s:S) statikus.			
	T osztálynak nincs alfa():T szignatúrájú metódusa, mert T nem valósítja meg az R interfészt.			
	M alfa():T metódusa visszaadhat C objektumot, mert C függ M -től.			
	D theta(s:S) metódusa kaphat paraméterül C objektumot, mert S és C is megvalósítja az R interfészt.			
	S nem valósítja meg az alfa():T szignatúrájú metódust, mert S nem függ T -től.			
	D theta(s:S) metódusa legfeljebb 5-ször hívható meg, mert D objektum legfeljebb 5 S -sel állhat asszociációban.			

2010.01.12 (B) – 1. Feladat



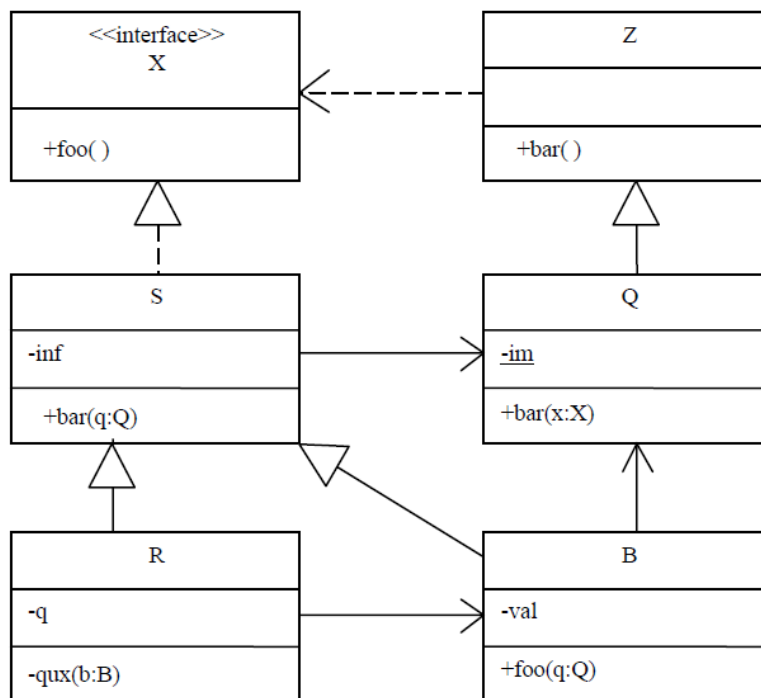
Válasz	Állítás	1.	2.	köv.
	M alfa():T metódusa visszaadhat C objektumot, mert C függ M -től.			
	D theta(s:S) metódusa nem kaphat paraméterül C objektumot, mert S és C is megvalósítja az R interfészt.			
	C eps(m:L) metódusa nem hívhatja meg a paraméter gamma() metódusát, mert az utóbbi metódus protected láthatóságú.			
	D theta(s:S) metódusa nem módosíthatja a paraméter iota attribútumát, mert a theta(s:S) statikus.			
	S nem valósítja meg az alfa():T szignatúrájú metódust, mert S nem függ T -től.			
	D theta(s:S) metódusa legfeljebb 5-ször hívható meg, mert D objektum legfeljebb 5 S -sel állhat asszociációban.			
	M bárhol helyettesíthető C -vel, mert mindketten megvalósítják az R interfészt.			
	T osztálynak van alfa():T szignatúrájú metódusa, mert T megvalósítja az R interfészt.			

2010.01.26 – 1. Feladat



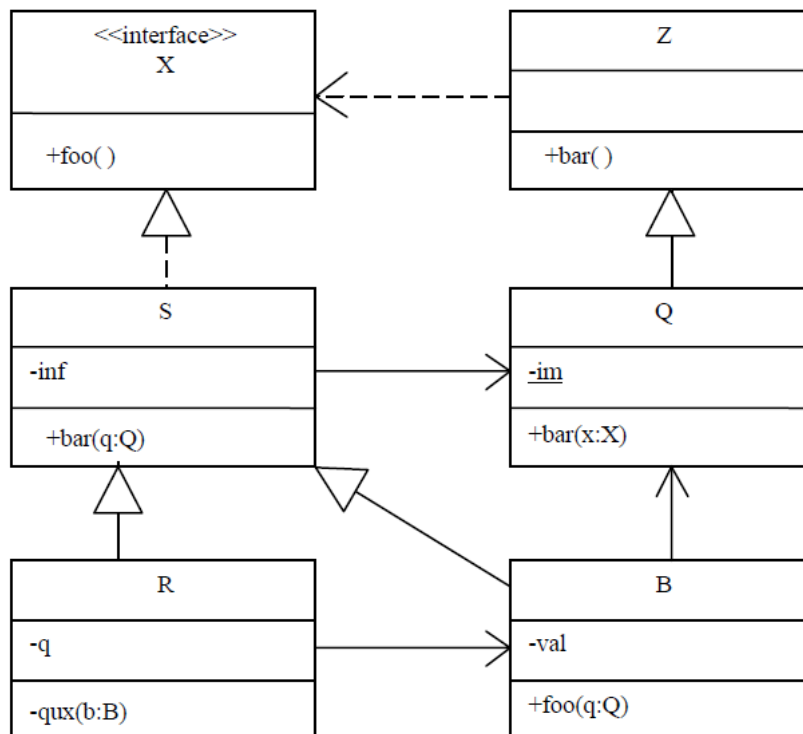
Válasz	Állítás	1.	2.	köv.
	Y bar(q:Q) metódusa kaphat paraméterül R objektumot, mert Y függ R -től.			
	T qux(y:Y) metódusa módosíthatja a paraméter val attribútumát, mert a metódus privát.			
	E bárhol helyettesíthető R -rel, mert azonos az interfészük.			
	Y foo(r:R) metódusa nem módosíthatja a paraméter im attribútumát, mert az attribútum statikus.			
	E bar(q:Q) metódusa kaphat E objektumot paraméterül, mert az E megvalósítja a Q interfészt.			
	E bar(q:Q) metódusa nem hívhatja meg egy paraméterül kapott W foo() metódusát, mert W -nek nincs ilyen szignatúrájú metódusa.			
	R bar(e:E) metódusa nem kaphat paraméterül Y objektumot, mert az Y-R asszociációban csak Y hívhatja R -t.			
	R nem valósítja meg a Q interfészt, mert van olyan szignatúrájú metódusa, ami nem szerepel a Q metódusai között.			

2011.01.04 (A) – 1. Feladat



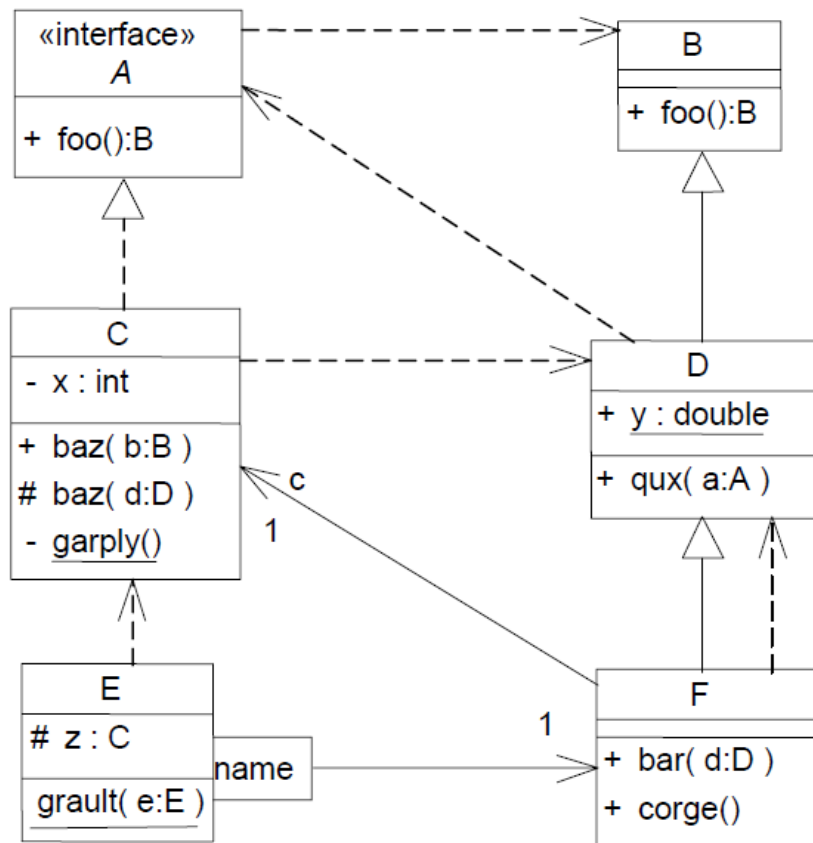
Válasz	Állítás	1.	2.	köv.
	Q helyettesíthető S -sel, mert S a Q leszármazottja.			
	S helyettesíthető B -vel, mert B megvalósítja az X interfészt.			
	R átadható paraméterül Q bar(x:X) metódusának, mert Q és S interfésze megegyezik.			
	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát.			
	B interfésze tartalmazza a bar(x:X) metódust, mert a metódus statikus.			
	Q meghívhatja S bar(q:Q) metódusát, mert mindketten megvalósítják az X interfészt.			
	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum konstans.			
	Q nem implementálja a foo() metódust, ezért nem függ az X interfésztől.			

2011.01.04 (B) – 1. Feladat



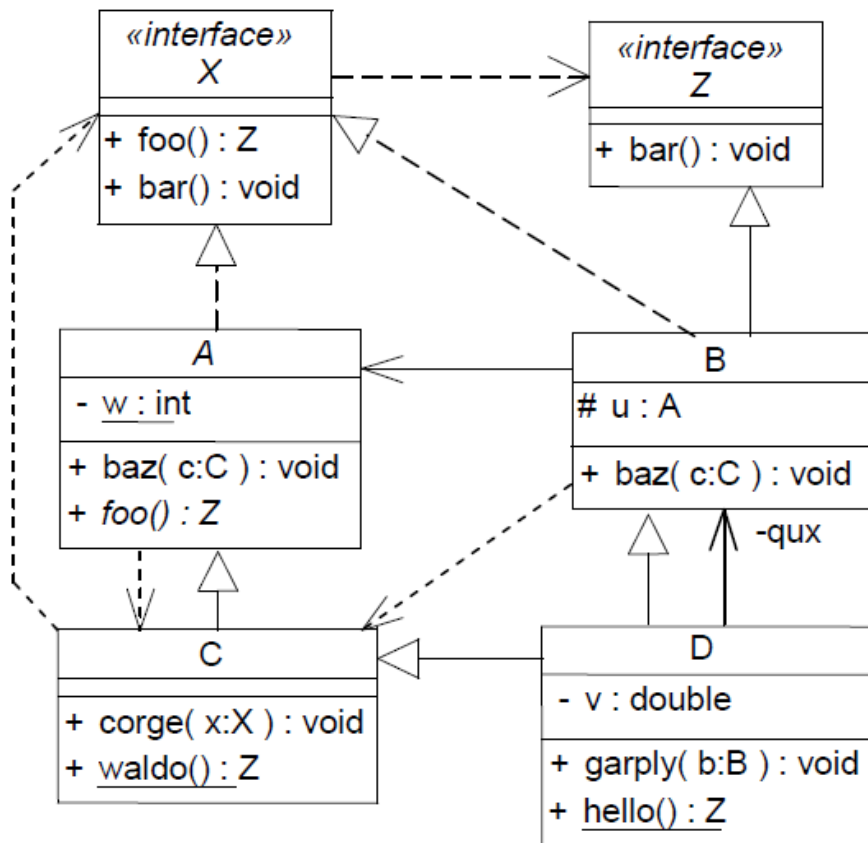
Válasz	Állítás	1.	2.	köv.
	R helyettesíthető B-vel, mert R függ B-től.			
	Q meghívhatja S bar(q:Q) metódusát, mert mindketten megvalósítják az X interfészt.			
	Q helyettesíthető S-sel, mert S a Q leszármazottja.			
	B interfésze tartalmazza a bar(x:X) metódust, mert a metódus statikus.			
	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát.			
	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum statikus.			
	Q nem implementálja a foo() metódust, ezért nem függ az X interfésztől.			
	B átvadható paraméterül Q bar(x:X) metódusának, mert Q és S interfésze megegyezik.			

2011.01.18 – 1. Feladat



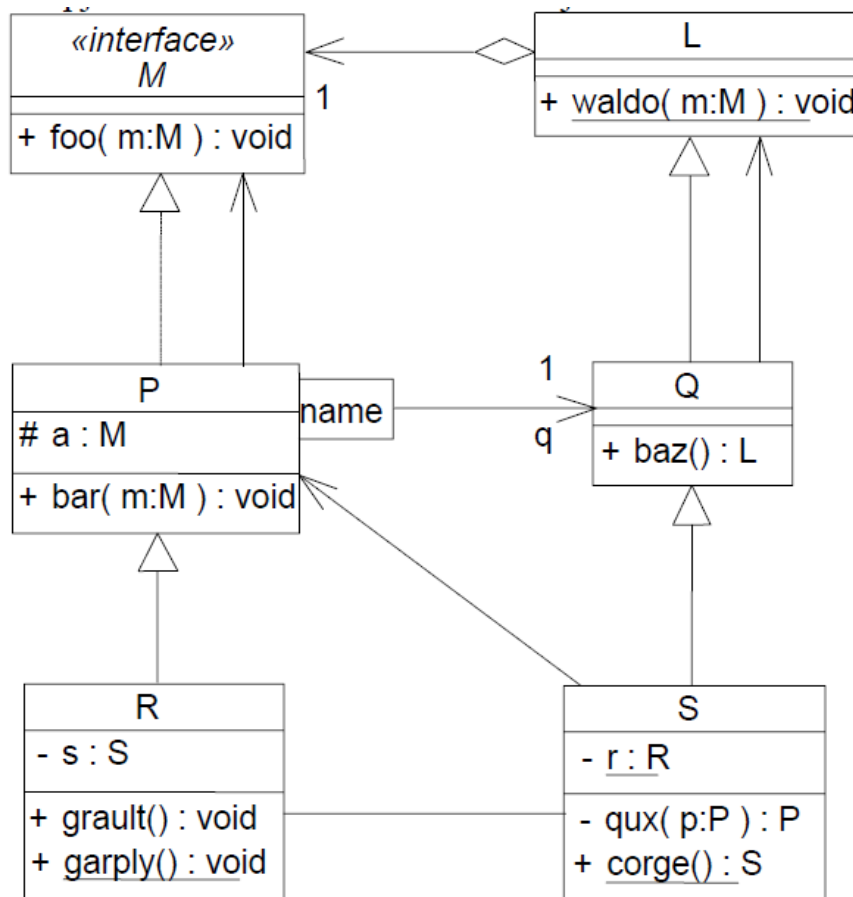
Válasz	Állítás	1.	2.	köv.
	D implementálja az A interfészt, mert A és B interfésze megegyezik.			
	F corge() metódusa nem módosíthatja D y attribútumát, mert D y attribútuma statikus.			
	E grault(e:E) metódusa nem hívhatja meg e z attribútumának baz(b:B) metódusát, mert a grault() statikus.			
	Egy F objektum pontosan egy E objektumot ismer, mert egy E objektum pontosan egy F objektumot ismer.			
	C foo() metódusa példányosíthat B típusú objektumot, ezért C függ B-től.			
	F bar(d:D) metódusából nem hívható meg c baz(b:B) metódusa, mert C baz(b:B) metódusa nem kaphat paraméterül D típusú objektumot.			
	C baz(d:D) metódusa nem hívhatja meg C garply() metódusát, mert C baz (d:D) metódusa nem statikus.			
	D qux(a:A) metódusa nem hívhatja meg egy paraméterül kapott C típusú objektum baz(b:B) metódusát, mert A nem helyettesíthető C-vel.			

2011.05.24 – 1. Feladat



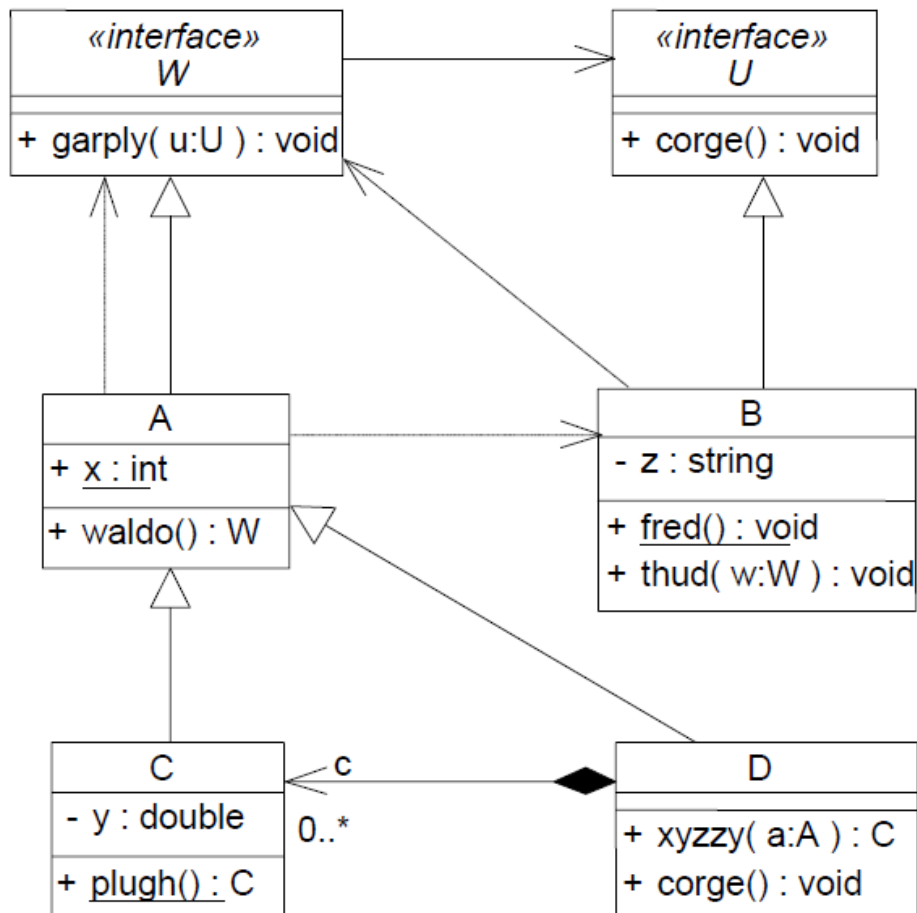
Válasz	Állítás	1.	2.	köv.
	D garply metódusa nem módosíthatja a b paraméter u attribútumát, mert protected attribútumhoz csak privát és protected metódusok férhetnek hozzá.			
	C corge metódusa kaphat paraméterül D típusú objektumot, ezért a metódus meghívhatja a kapott objektum garply metódusát.			
	D garply metódusa kaphat paraméterül A típusú objektumot, mert A és B interfésze megegyezik.			
	C waldo metódusa virtuális, ezért a B osztály baz függvénye egy paraméterül kapott D típusú objektumon meghívhatja a waldo metódust.			
	A baz metódusa nem módosíthatja A w attribútumát, mert A baz metódusa nem statikus.			
	C -nek van bar metódusa, ezért C implementálja a Z interfészt.			
	D hello metódusa nem módosíthatja D v attribútumát, mert D v attribútuma privát.			
	B baz metódusa nem hívhatja meg B u attribútumának foo metódusát, mert az A osztály foo metódusa absztrakt.			

2011.06.07 – 1. Feladat



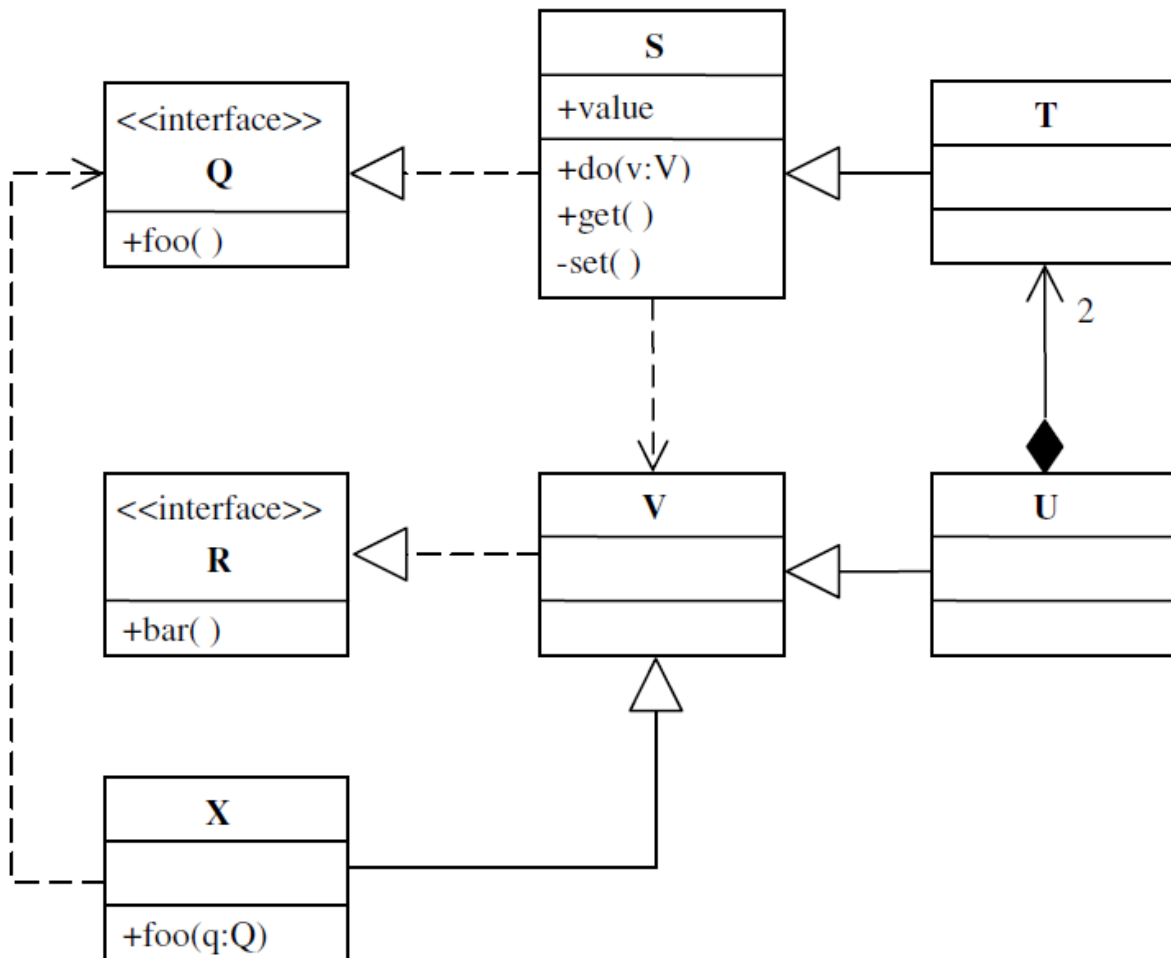
Válasz	Állítás	1.	2.	köv.
	R garply metódusa meghívhatja az s attribútum corge metódusát, mert a garply és a corge metódus is statikus.			
	P nem hívhat Q osztályon waldo függvényt, mert L waldo függvénye nem virtuális.			
	Q függ M -től, mert Q ismeri P -t.			
	Az M típus közvetlenül nem példányosítható, ezért R gault metódusa nem hívhatja meg P a attribútumának foo metódusát.			
	Egy P típusú objektum pontosan egy Q típusú objektumot ismer, ezért P függ Q -től.			
	S az L leszármazottja, ezért Q baz függvénye példányosíthat S típusú objektumot.			
	S qux függvénye nem módosíthatja az r attribútum s attribútumát, mert R s attribútuma privát.			
	S ismeri M -et, mert az S osztály L őse függ M -től.			

2011.06.14 – 1. Feladat



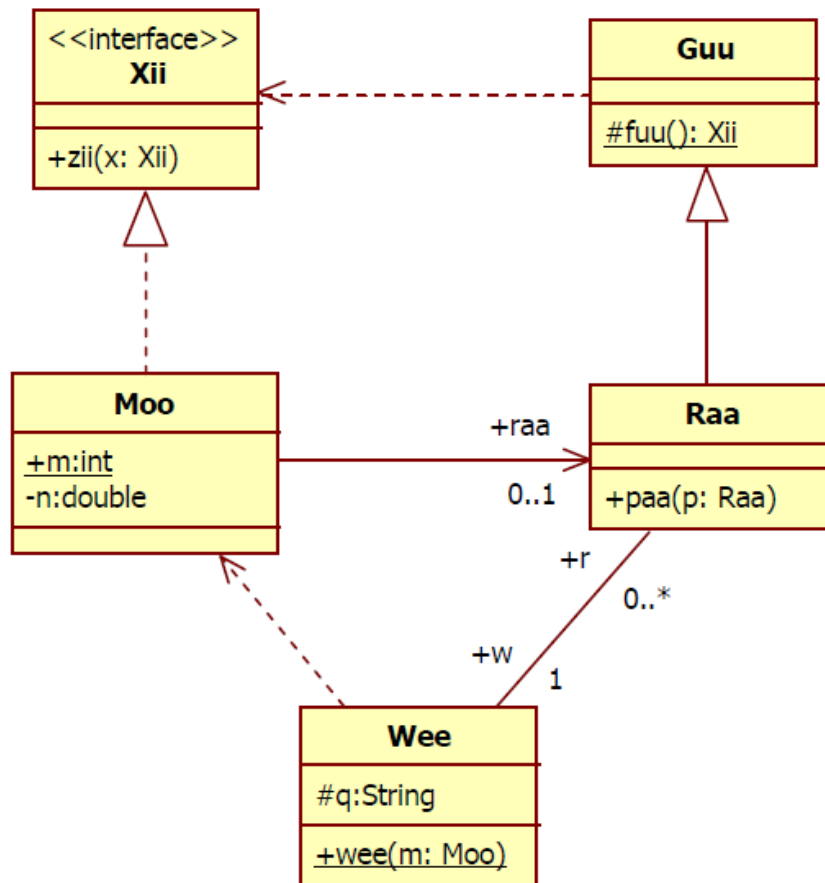
Válasz	Állítás	1.	2.	köv.
	U interfésze részhalmaza D interfészének, ezért D megvalósítja az U interfészt.			
	D xyzy függvénye visszaadhatja eredményként a paraméterként kapott a objektumot, mert C az A leszármazottja.			
	A waldo függvénye nem példányosíthat B típusú objektumot, mert B nem implementálja a W interfészt.			
	C plugh függvénye nem módosíthatja A x attribútumát, mert A x attribútuma protected.			
	Egy C objektum sok D objektumot tartalmaz, ezért C ismeri D -t.			
	B fred függvénye nem módosíthatja a z attribútum értékét, mert B z attribútuma nem protected.			
	B thud függvénye meghívhatja egy paraméterül kapott C típusú objektum plugh függvényét, mert C plugh függvénye virtuális.			
	C nem függ U -tól, mert C A őszosztálya sem függ U -tól.			

2013.06.18 – 5. feladat



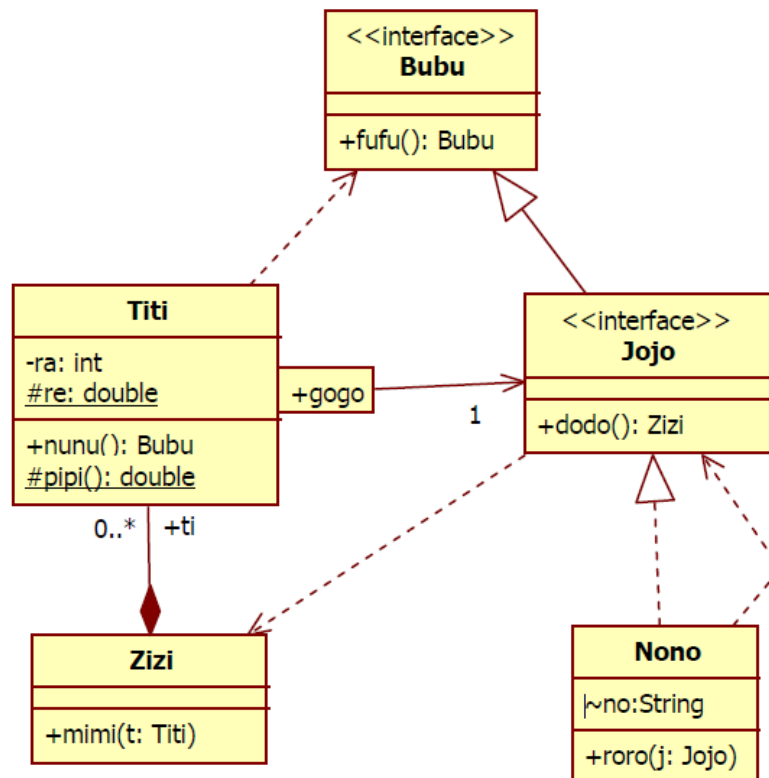
Válasz	Állítás	1.	2.	köv.
	T létrehozhat U osztályú objektumot, mert S létrehozhat V-t és T az S-nek, U a V-nek leszármazottja.			
	X foo(q:Q) metódusa kaphat paraméterül T-t, mert T-nek is van foo() metódusa.			
	X foo(q:Q) metódusa meghívhatja a paraméterül kapott S get() metódusát, mert S megvalósítja a Q interfészt.			
	V törlésekor törölni kell két T-t is, mert egy U-nak két T komponense van és U a V leszármazottja.			
	T nem függ U-tól, mert T nem függ V-től sem.			
	X meghívhatja egy Q interfészű objektum foo() metódusát, mert X implementálja Q-t.			
	X bar() metódusából meghívhatjuk egy Q interfészű objektum foo() metódusát, mert X foo(q:Q) metódusából is hívhatjuk egy Q interfészű objektum foo() metódusát.			
	S set() metódusa nem módosíthatja a value attribútumot, mert a láthatóságuk különböző.			

2014.01.14 – 1. Feladat



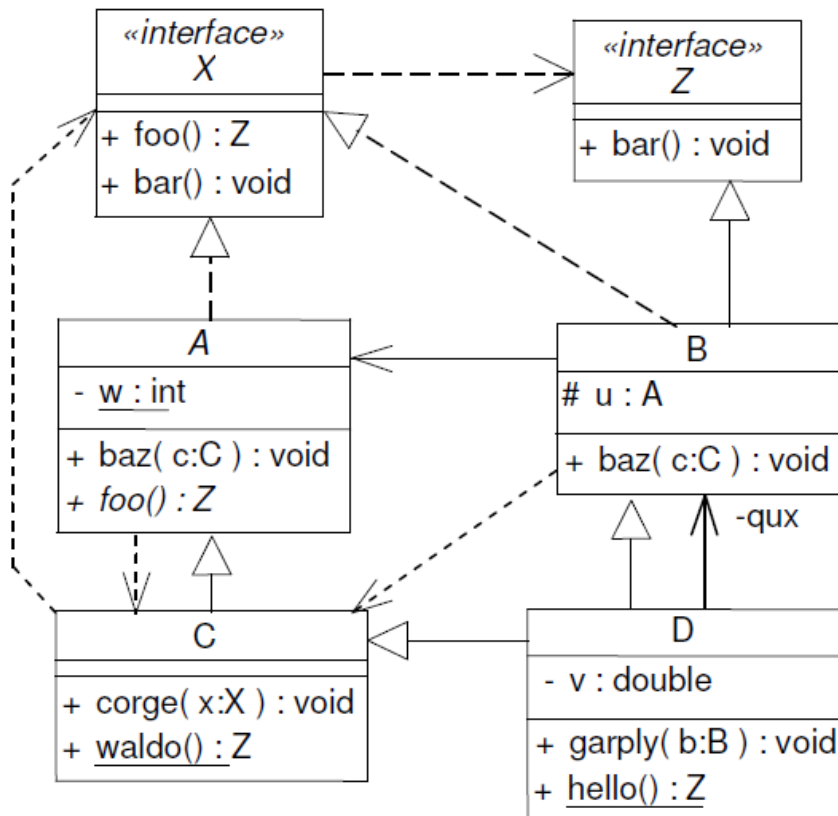
Válasz	Állítás	1.	2.	köv.
	Raa nem függ Xii -től, mert Guu nem implementálja a Xii interfészt.			
	Raa paa(p:Raa) függvénye kaphat paraméterül Guu objektumot, mert függvényparaméterként Raa helyettesíthető Guu -val.			
	Raa paa(p:Raa) függvénye nem módosíthatja Moo m attribútumát, mert Moo m attribútuma statikus.			
	Moo zii(x:Xii) függvénye nem szorozhatja össze az m és n attribútumok értékeit, mert az n attribútum privát.			
	Raa paa(p:Raa) függvénye nem hívhatja meg a fuu():Xii függvényt, mert az sértené a Pauli-elvet.			
	Wee wee(m:Moo) függvénye nem módosíthatja a q attribútumot, mert q package láthatóságú.			
	Wee wee(m:Moo) függvénye meghívhatja az m paraméter zii(x:Xii) függvényét, mert az nem sérti a Demeter-törvényt.			
	Moo és Xii interfésze megegyezik, mert Moo nem definiál újabb függvényt.			

2014.01.21 – 1. Feladat



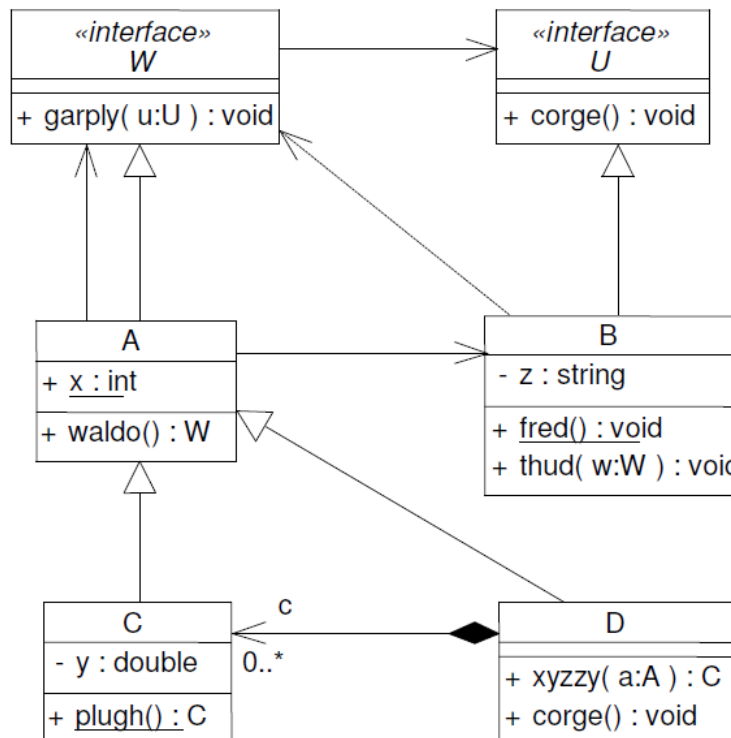
Válasz	Állítás	1.	2.	köv.
	Zizi mimi(t:Titi) függvénye nem hívhatja meg a t paraméter nunu() függvénye által visszaadott Jojo interfész objektum dodo() függvényét, mert az sértené a Demeter-törvényt.			
	Titi pipi() függvénye nem szorozhatja össze a ra és re attribútum értékét, mert privát változóhoz csak privát függvény férhet hozzá.			
	Zizi a ti.nunu() hívás eredményén hívhat roro(j:Jojo) metódust, mert Nono implementálja a Bubu interfészt.			
	Egy Titi objektum csak egy Jojo interfészű objektumot ismer, mert közöttük lévő asszociáció Jojo oldalán 1-es számosság szerepel.			
	Nono roro(j:Jojo) függvénye kaphat paraméterül Bubu interfészű objektumot, mert Jojo implementálja a Bubu interfészt.			
	Nono dodo() függvénye nem hozhat létre Zizi objektumot, mert Nono nem ismeri Zizi -t.			
	Zizi függ Bubu -tól, mert Titi függ Bubu -tól.			
	Nono dodo() függvénye nem módosíthatja a no attribútum értékét, mert Javában a String típus immutábilis.			

2014.05.27 – 1. Feladat



Válasz	Állítás	1.	2.	köv.
	B baz metódusa nem hívhatja meg B u attribútumának foo metódusát, mert az A osztály foo metódusa absztrakt.			
	C corge metódusa kaphat paraméterül D típusú objektumot, ezért a metódus meghívhatja a kapott objektum garply metódusát.			
	C waldo metódusa virtuális, ezért a B osztály baz függvénye egy paraméterül kapott D típusú objektumon meghívhatja a waldo metódust.			
	A baz metódusa nem módosíthatja A w attribútumát, mert A baz metódusa nem statikus.			
	C -nek van bar metódusa, ezért C implementálja a Z interfészt.			
	D garply metódusa kaphat paraméterül A típusú objektumot, mert A és B interfésze megegyezik.			
	D hello metódusa nem módosíthatja D v attribútumát, mert D v attribútuma privát.			
	D garply metódusa nem módosíthatja a b paraméter u attribútumát, mert protected attribútumhoz csak privát és protected metódusok férhetnek hozzá.			

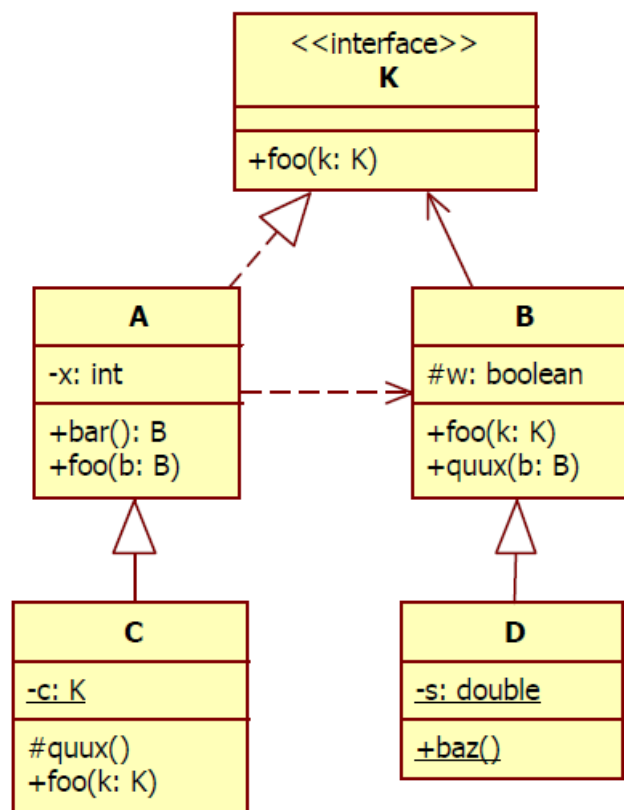
2014.06.03 – 1. Feladat



Válasz	Állítás	1.	2.	köv.
	D törlésekor legalább egy C objektumot is törölni kell, mert D tartalmaz C -t.			
	U interfésze részhalmaza D interfészének, ezért D megvalósítja az U interfészt.			
	C nem függ U -tól, mert C ősszánya (A) sem függ U -tól.			
	A waldo függvénye nem példányosíthat B típusú objektumot, mert B nem implementálja a W interfészt.			
	C plugh függvénye nem módosíthatja A x attribútumát, mert A x attribútuma protected.			
	D xyzy függvénye visszaadhatja eredményként a paraméterként kapott a objektumot, mert C az A leszármazottja.			
	B fred függvénye nem módosíthatja a z attribútum értékét, mert B z attribútuma nem protected.			
	B thud függvénye meghívhatja egy paraméterül kapott C típusú objektum plugh függvényét, mert C plugh függvénye virtuális.			

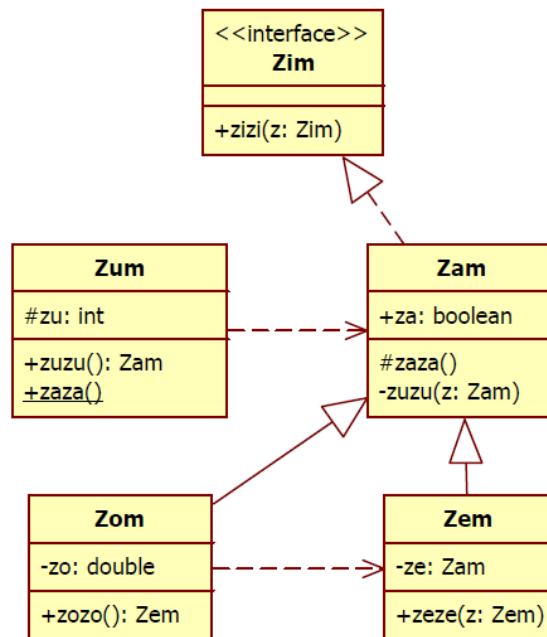
2015.01.06 – 4. Feladat

Az alábbi UML2 diagram alapján - a kulcs felhasználásával - jellemezze az állításokat!



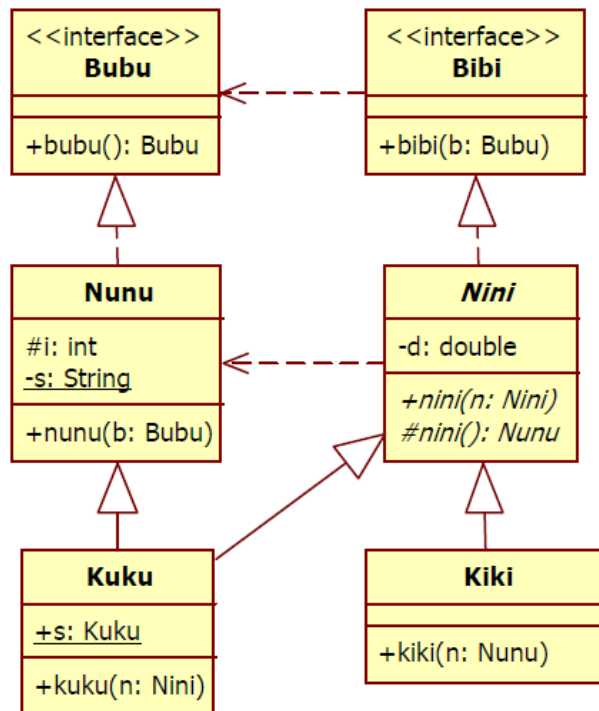
Válasz	Állítás	1.	2.	köv.
	C egyetlen függvénye sem módosíthatja egy paraméterül kapott B típusú objektum w attribútumát, mert C nem függ B-től.			
	D baz függvénye nem módosíthatja B w attribútumát, mert w statikus.			
	Van olyan foo függvény, amely nem kaphat paraméterül B típusú objektumot, mert B nem implementálja a K interfészt.			
	A bar függvénye nem példányosíthat D típusú objektumot, mert D nem függ A-tól.			
	C quux függvénye nem módosíthatja a c attribútum értékét, mert quux nem privát.			
	D foo függvénye nem hívhatja meg a paraméterül kapott C objektum foo(k:K) függvényét, mert D nem ismeri C-t.			
	Minden D-ben deklarált függvény módosíthatja az s attribútumot, mert s statikus.			
	C és D interfésze különbözik, mert D nem implementálja a K interfészt.			

2015.01.13 – 4. Feladat



Válasz	Állítás	1.	2.	köv.
	Van olyan zaza függvény, amely nem módosíthatja a za attribútumot, mert za nem statikus.			
	Zum zuzu függvénye nem példányosíthat Zem objektumot, mert Zem nem függ Zum -tól.			
	Zom zozo függvénye példányosíthat Zom objektumot, mert Zom a Zem leszármazottja.			
	Zum zaza függvénye nem módosíthatja a zu attribútum értékét, mert zu privát.			
	Zom zizi függvénye nem hívhatja meg Zem zizi függvényét, mert egyiknek sincs zizi függvénye.			
	Zem zeze függvénye nem kaphat paraméterül Zom objektumot, mert Zom függ Zem -től.			
	Zam zuzu függvénye meghívhatja a paraméterül kapott Zom típusú objektum zaza függvényét, mert zaza absztrakt.			
	Zam zaza függvénye nem hívhat meg minden zuzu függvényt, mert Zam nem ismeri Zum -ot.			

2015.01.20 – 4. Feladat



Válasz	Állítás	1.	2.	köv.
	Nini bibi függvénye nem kaphat paraméterül Kuku objektumot, mert Bibi nem ismeri Kuku -t.			
	Van olyan s attribútum, amelyet nem módosíthat Kiki kiki függvénye, mert nem minden s attribútum publikus.			
	Kuku implementálja a Bubu és Bibi interfészeket, mert Java-ban megengedett az interfészek többszörös öröklése.			
	Nunu nunu függvénye nem példányosíthat Kiki objektumot, mert Kiki függ Nunu -tól.			
	Kuku nem példányosítható, mert nem implementálja az absztrakt nini függvényeket.			
	Kuku és Kiki interfésze megegyezik, mert mindkettő megvalósítja a Bibi interfészt.			
	Kiki kiki függvénye nem kaphat paraméterül Kuku objektumot, mert ha kapna, az sértené a Liskov elvet.			
	Kuku bubu függvénye nem módosíthatja az i attribútum értékét, mert i statikus.			

asd

asd