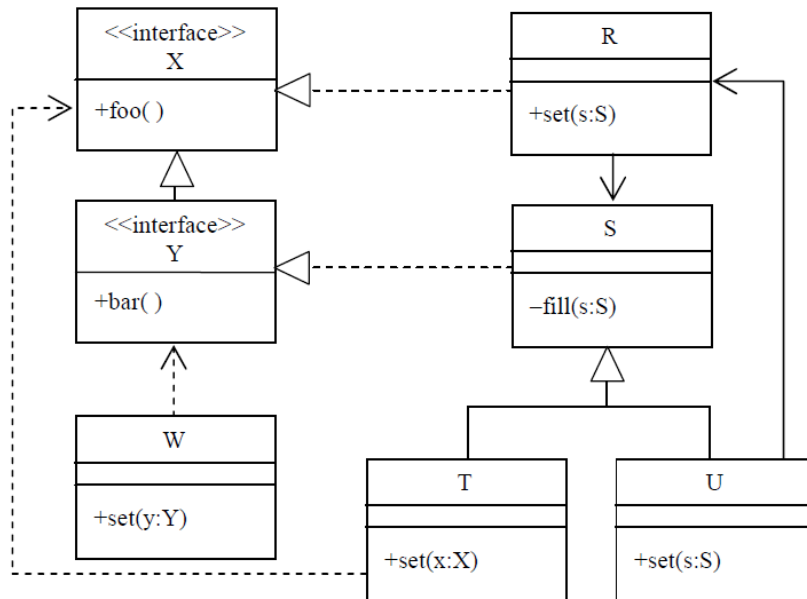


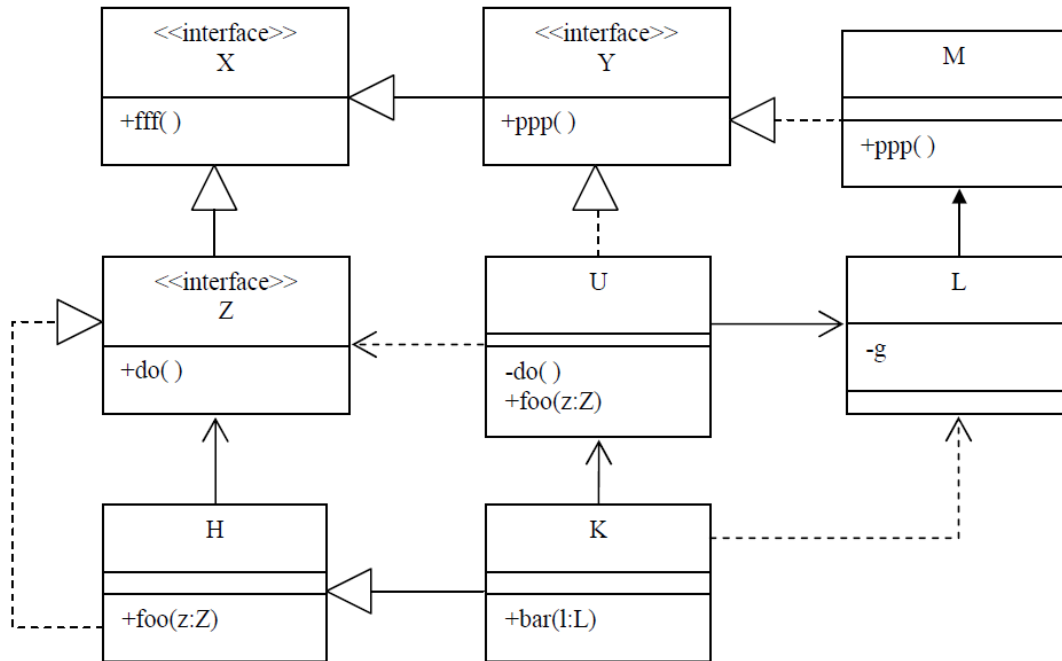
UML class diagram – A,B,C,D,E feladattípus, megoldások

2008.01.08 – 1. Feladat



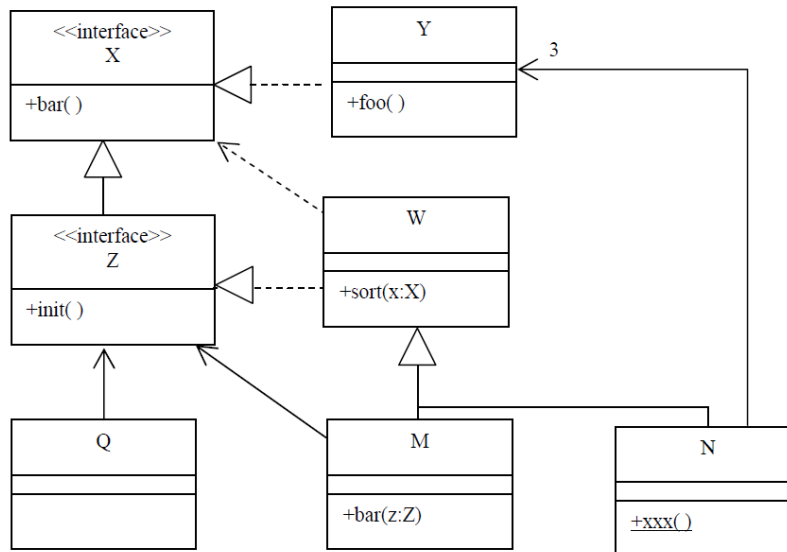
Válasz	Állítás	1.	2.	köv.
E	Y bárhol helyettesíthető W-vel, mert W az Y leszármazottja. <i>Y akkor lenne helyettesíthető W-vel, ha W leszármazottja lenne Y-nak, viszont ez nem igaz, így mind2 tagmondat hamis.</i>	-	-	
D	U bármikor lehet T.set(x:X) paramétere, mert U megvalósítja az X interfészt. <i>A T set metódusa X-t vár, vagy olyan osztályt/interfészt, ami azt örökli/implementálja. U (ha csak közvetetten is), de ennek eleget tesz, így mind2 tagmondat igaz, és a következtetés is igaz.</i>	+	+	+
B	R meghívhatja saját set(s:S) metódusából egy W set(y:Y) metódusát, mert S megvalósítja Y-t. <i>Az S nem ismeri W-t, így azon keresztül se hívható meg egy W set(y:Y) metódus így ez hamis. S megvalósítja az Y interfészt, ez triviális, igaz.</i>	-	+	
E	S fill(s:S) metódusa nem kaphat paraméterül T-t, mert a metódus protected. <i>Mivel T S leszármazottja, így kaphat T-t, tehát ez hamis. A metódus privát, nem protected, tehát ez is hamis.</i>	-	-	
A	T megvalósítja az X interfészt, mert T az R leszármazottja. <i>Bár közvetetten, de megvalósítja, így ez igaz. A T nem leszármazottja R-nek, még közvetetten se, így ez hamis.</i>	+	-	
B	T pontosan egy U-t tartalmazhat, mert csak egy közvetlen ősük van. <i>Nincs köztük asszociáció/aggregáció, így nem tartalmazhat T U-t, vagyis ez hamis. A 2. tagmondat persze triviálisan igaz.</i>	-	+	
E	T bárhol helyettesíthető U-val, mert egyforma az interfészük. <i>U nem leszármazottja T-nek, így ez hamis. Akkor azonos 2 osztály interfésze, ha „kivülről ugyanazokat a metódusokat látjuk”, azonos szignatúrákkal! Másképpen: ha példányosítanánk a 2 osztályt, akkor a példányok legördülő menüjében ugyanannyi, és ugyanazon metódusokat látnánk. Ez nem teljesül: set(x:X) vs. set(s:S), tehát ez hamis.</i>	-	-	
B	U meghívhatja S fill(s:S) metódusát, mert R asszociációban van S-sel. <i>Mivel a metódus privát, így U nem fér hozzá, tehát ez hamis. 2. tagmondat triviálisan igaz.</i>	-	+	

2008.01.15 – 1. Feladat



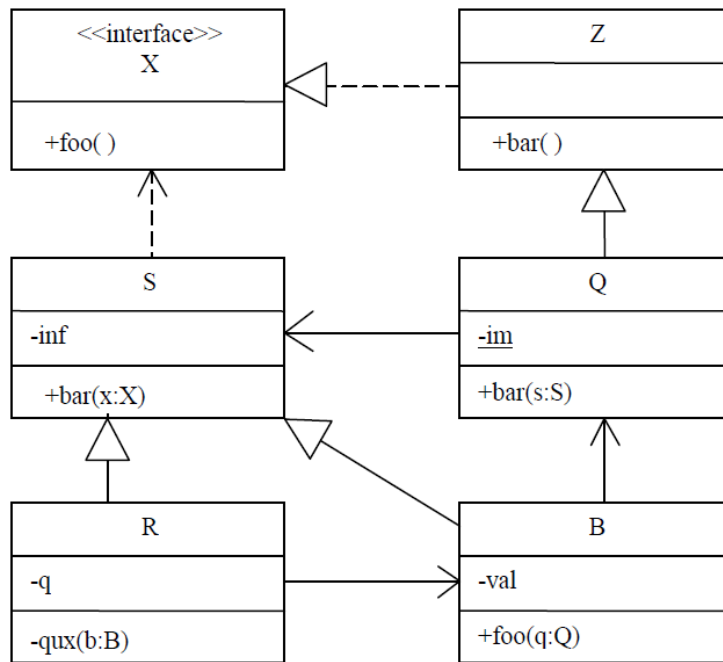
Válasz	Állítás	1.	2.	köv.
B	H bárhol helyettesítheti U-t, mert mindketten megvalósítják az X interfészt. <i>H nem leszármazottja U-nak, így ez hamis. Közvetetten mind2-en megvalósítják az X interfészt, így ez igaz.</i>	-	+	
B	H foo(z:Z) metódusa meghívható egy U-val, mert U megvalósítja az Y interfészt. <i>Z helyére Z interfészt megvalósító osztály kerülhetne. Mivel ez U-ra nem igaz, ez a tagmondat hamis. U megvalósítja az Y interfészt, ez triviális.</i>	-	+	
A	U nem hívhatja meg K bar(l:L) metódusát, mert K nem függ L-től. <i>U-nak fogalma nincs K-ról, így ez igaz. K függ L-től, ez triviális, így ez hamis.</i>	+	-	
A	K implementálja a Z interfészt, ezért K meghívhatja U do() metódusát. <i>Bár közvetetten, de implementálja („H-n keresztül), tehát ez igaz. U-nak a do() metódusa privát, így ahhoz csak maga az U fér hozzá, így ez hamis.</i>	+	-	
B	K nem hozhat létre L objektumot, mert az L g attribútuma privát. <i>K függ L-től, így létrehozhat L objektumot, tehát ez hamis. 2. tagmondat triviálisan igaz.</i>	-	+	
C	U foo(z:Z) metódusa nem hívhatja meg egy paraméterül kapott H foo(z:Z) metódusát, mert U nem implementálja a Z interfészt. <i>Bár H megvalósítja a Z interfészt, így kaphat az U foo(z:Z) metódus H paramétert, viszont azon is csak olyan metódusok hívhatók, amik a Z-ben megvannak, és mivel Z-ben nincs foo(z:Z) metódus, így ez igaz. A 2. tagmondat triviálisan igaz, persze a 2 tagmondatnak köze nincs egymáshoz, így a következtetés triviálisan hamis.</i>	+	+	-
E	K bárhol helyettesíthető U-val, mert K az U leszármazottja. <i>Az U nem leszármazottja K-nak, így nem helyettesíthető, vagyis ez hamis. A 2. tagmondat triviálisan hamis.</i>	-	-	
A	L nem ismeri az Y interfészt, ezért L nem hívhatja meg M ppp() metódusát. <i>(TODO) Jó kérdés, az a nyíl mi akart lenni, így ezt passzolom.</i>	+	-	

2008.01.22 – 1. Feladat



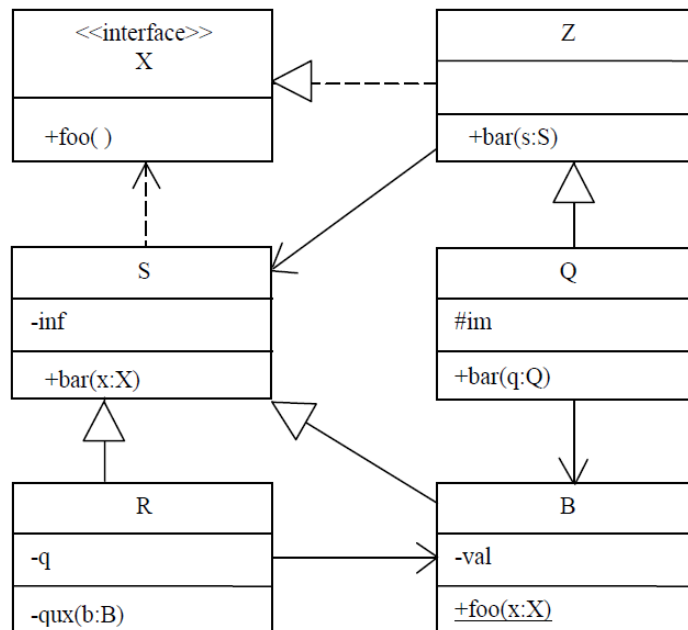
Válasz	Állítás	1.	2.	köv.
B	Y helyettesíthető W-vel, mert mindketten megvalósítják az X interfészt. <i>W nem leszarmazottja Y-nak, így ez hamis. Mindketten megvalósítják az X interfészt (W közvetten, „Z interfészen keresztül”), így ez igaz.</i>	-	+	
C	N meghívhatja Y foo() metódusát, mert N megvalósítja a Z interfészt. <i>Ott van köztük az asszociáció, tehát miért ne hívhatná (már a függés is elég lenne), így ez igaz. A 2. tagmondat triviálisan igaz. A 2 tagmondatnak viszont köze nincs egymáshoz, az N és Y közötti kapcsolatnak (ez esetben asszociáció) köszönhető, hogy N meghívhatja Y foo() metódusát.</i>	+	+	-
C	M bar(z:Z) metódusa kaphat paraméterül N objektumot, mert van közös ősök. <i>A Z interfészt implementálja N, így ez igaz. Az is triviálisan igaz, hogy van közös ősök, de ennek megint csak semmi köze az 1. tagmondatához.</i>	+	+	-
B	W nem helyettesíthető M-mel, mert W-nek nincs bar(z:Z) szignatúrájú metódusa. <i>De, helyettesíthető, mivel M leszarmazottja W-nek. 2. tagmondat triviálisan igaz.</i>	-	+	
E	Q helyettesíthető M-mel, mert mindkettő megvalósítja a Z interfészt. <i>Nope, M nem leszarmazottja Q-nak, így ez hamis. M megvalósítja a Z interfészt (W-n keresztül), de Q nem, így ez hamis.</i>	-	-	
B	N xxx() metódusa meghívható a W osztály sort(x:X) metódusából, mert az N.xxx() statikus. <i>A W-nek fogalma sincs N létezéséről, így ez hamis. A 2. tagmondat triviálisan igaz.</i>	-	+	
C	W sort(x:X) metódusa meghívhatja egy paraméterül kapott Y objektum bar() metódusát, mert W-nek is van ugyanilyen szignatúrájú metódusa. <i>Az X helyére kerülhet Y, mivel Y megvalósítja az X interfészt, és mivel a bar() metódus az X interfészhez tartozik, így ez igaz. 2. tagmondat is triviálisan igaz, hiszen W implementálja közvetten az X interfészt. A következtetés hamis, okát lásd fentebb.</i>	+	+	-
A	M-nek és N-nek különböző az interfésze, mert N nem valósítja meg X-t. <i>1. tagmondat triviálisan igaz (pl. M-nek van bar(z:Z) metódusa, míg N-nek nincs), így ez igaz. N (közvetten) megvalósítja az X interfészt, így ez igaz.</i>	+	-	

2008.05.27 – 1. Feladat



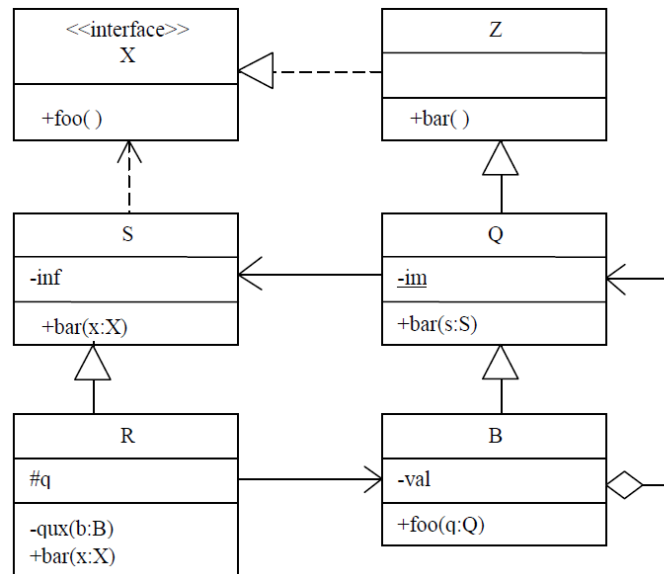
Válasz	Állítás	1.	2.	köv.
E	S helyettesíthető Q-val, mert Q az S leszármazottja. <i>Q nem leszármazottja S-nek, így ez hamis. A 2. tagmondat se talált be.</i>	-	-	
A	S helyettesíthető B-vel, mert B megvalósítja az X interfészt. <i>B leszármazottja S-nek, így ez igaz. Nem, nem valósítja meg.</i>	+	-	
A	R átadható paraméterül Q bar(s:S) metódusának, mert Q és S interfésze megegyezik. <i>R S-nek a leszármazottja, így a bar(s:S) metódus kaphat paraméterül R-t is, tehát ez igaz. A 2 osztály interfésze nagyon nem egyezik (Q-nak van bar(), és foo() metódusa is, míg ez S-ről nem mondható el), így ez hamis.</i>	+	-	
B	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát. <i>Még szép, hogy látja, ez hamis. A 2. tagmondat meg triviálisan igaz.</i>	-	+	
A	Q meghívhatja S bar(x:X) metódusát, mert mindketten megvalósítják az X interfészt. <i>Köztük van asszociáció (egy függés is elég lenne), így ez igaz. Csak a Q valósítja meg (közvetetten) az X interfészt, S nem, így ez hamis.</i>	+	-	
E	B interfésze tartalmazza bar(s:S) metódust, mert a metódus statikus. <i>B nem leszármazottja Q-nek, így nem szerepel az interfészében a metódus, tehát ez hamis. A 2. tagmondat triviálisan hamis.</i>	-	-	
A	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum konstans. <i>Q csak örökl Z-nek bar() metódusát (mert nem írja felül, ha megtenné, szerepele az osztályban a metódus külön), így a bar() metódus nem módosíthatja az im attribútumot, ezért lesz igaz. A 2. tagmondat egy baromság, hamis.</i>	+	-	
E	B-nek nincs bar(x:X) metódusa, ezért nem függ az X interfésztől. <i>B S-nek a leszármazottja, és a metódus public, így ez hamis. Van neki bar(x:X) metódusa, pont ezért függ tőle, tehát ez is hamis.</i>	-	-	

2008.06.10 – 1. Feladat



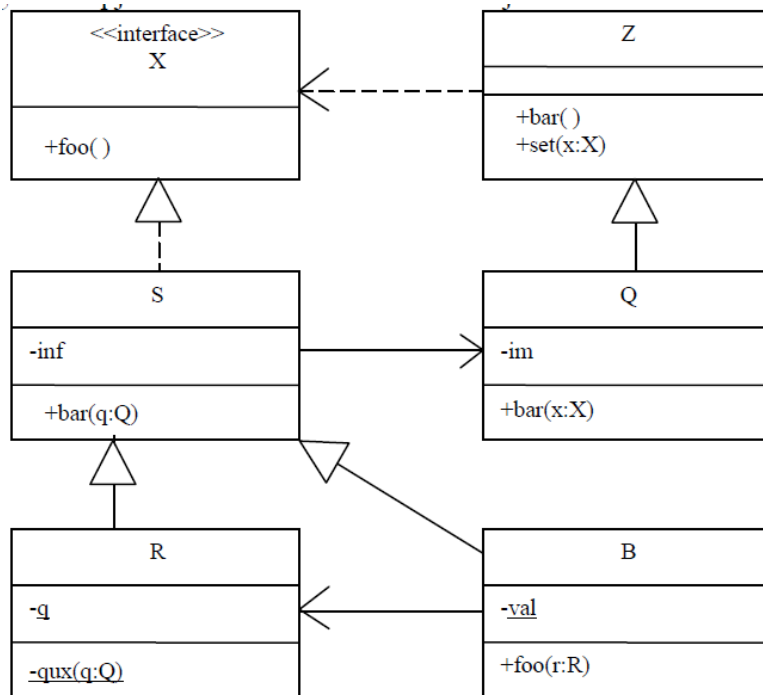
Válasz	Állítás	1.	2.	köv.
E	R helyettesíthető S-sel, mert S az R leszármazottja. <i>Az S nem R leszármazottja, ezért ez hamis (S-t lehetne helyettesíteni R-rel). Nope, Az R az S-nek leszármazottja.</i>	-	-	
E	R helyettesíthető B-vel, mert mindketten megvalósítják az X interfészt. <i>B nem leszármazottja R-nek, így ez hamis. Szomorú, de egyikse valósítja meg az X interfészt, csak függenek tőle.</i>	-	-	
A	R átadható paraméterül Q bar(s:S) metódusának, mert Q és S interfésze megegyezik. <i>Az R az S-nek leszármazottja, így ez igaz. 2 tagmondat triviálisan hamis.</i>	+	-	
D	B foo(x:X) metódusa nem látja a val attribútum értékét, mert az attribútum nem statikus. <i>Igaz, mivel az osztály statikus metódusa nem is módosíthatja saját magának a példány attribútumát. 2 tagmondat triviálisan igaz, és a következtetés szintén.</i>	+	+	+
A	Q meghívhatja S bar(x:X) metódusát, mert mindketten megvalósítják az X interfészt. <i>Mivel Z és S között asszociáció van, Q meg öröklé Z-t, és a bar(x:X) metódus is public, így Q eléri a metódust, tehát ez igaz (imho szerintem egy függés is elég lenne ahhoz, hogy a tagmondat igaz legyen...). S nem valósítja meg X-t, triviálisan hamis.</i>	+	-	
B	B interfésze tartalmaz qux(b:B) metódust, mert B-nek van R-rel közös őse. <i>Nope, több sebből is vérzik a dolog (B nem öröklé R-t, a metódus privát). A 2. tagmondat triviálisan igaz.</i>	-	+	
A	Q bar(s:S) metódusa nem módosíthatja az im attribútumot, mert az attribútum privát. <i>Igaz, mivel Q nem override-olja a bar(s:S) metódust, Z-nek meg fogalma sincs Q-ról. Az attribútum protected, nem privát, így ez triviálisan hamis.</i>	+	-	
B	B meghívhatja R qux(b:B) metódusát, mert a metódus paramétere B osztályú. <i>Nem nyert, a B nem ismeri R-t (ráadásul privát a metódus...). 2. tagmondat triviálisan igaz.</i>	-	+	

2008.06.17 – 1. Feladat



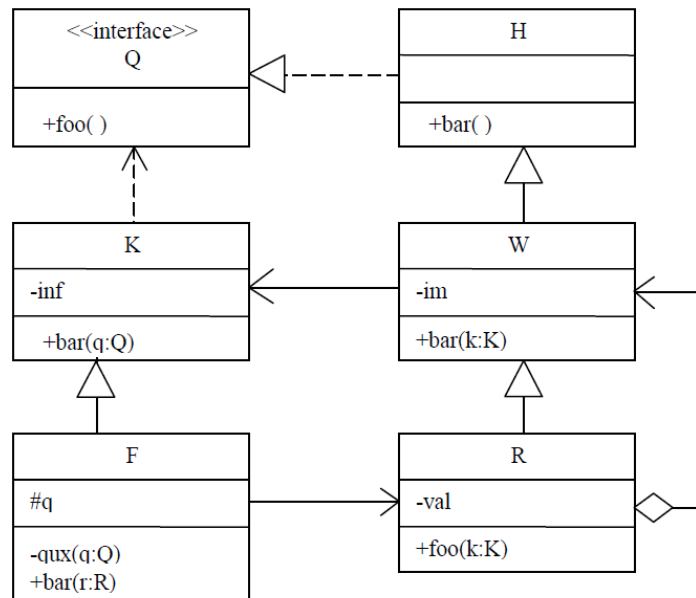
Válasz	Állítás	1.	2.	köv.
E	R helyettesíthető B-vel, mert mindketten megvalósítják az X interfészt. <i>B nem leszármazottja R-nek, így ez hamis. R nem valósítja meg X-t, így ez hamis.</i>	-	-	
D	Z helyettesíthető B-vel, mert B a Z leszármazottja. <i>Nincs hozzáfűzni való, az állítás „magyarázza magát”.</i>	+	+	+
B	Q bar(s:S) metódusa nem módosíthatja Q im attribútumát, mert az attribútum statikus. <i>De, módosíthatja (példány metódus módosíthat bármilyen láthatóságú attribútumot, és az se baj, ha static), tehát ez hamis. 2. tagmondat triviálisan igaz.</i>	-	+	
E	R qux(b:B) metódusa nem hívhat meg a paraméteren foo() metódust, mert B nem implementálja az X interfészt. <i>Van köztük asszociáció B felé irányítva, illetve a foo() metódus public, így semmi akadálya, tehát hamis. Implementálja (igaz, csak közvetten), tehát hamis.</i>	-	-	
B	S bar(x:X) metódusa meghívhatja egy paraméterül kapott Z bar() metódusát, mert Z megvalósítja az X interfészt. <i>X-t megvalósítja Z, így kaphat paraméterül Z-t, viszont csak olyan metódusokat lehet hívni, ami X-ben is van. Ez a bar()-ra nem igaz, ez nincs benne X-ben, így hamis. 2. tagmondat triviálisan igaz.</i>	-	+	
E	R qux(b:B) metódusa nem módosíthatja a q attribútumot, mert az attribútum privát. <i>Egy példány metódus láthatóságtól függetlenül módosíthat tetszőleges láthatóságú attribútumot, és még az se baj, ha static, így ez hamis. Az attribútum protected, nem privát, így ez is hamis.</i>	-	-	
B	B bar(s:S) metódusa nem kaphat paraméterül R objektumot, mert B nem ismeri az R osztályt. <i>R az S-nek leszármazottja, így kaphat, tehát ez hamis. 2. tagmondat triviálisan igaz (csak R ismeri B-t, de fordítva nem igaz!).</i>	-	+	
B	S bar(x:X) metódusa kaphat paraméterül R objektumot, mert R függ X-től. <i>R nem valósítja meg az X interfészt, így ez hamis. Mivel örökli S-t, aki függ X-től, így ő is függeni fog tőle, így ez igaz.</i>	-	+	

2009.01.06 – 1. Feladat



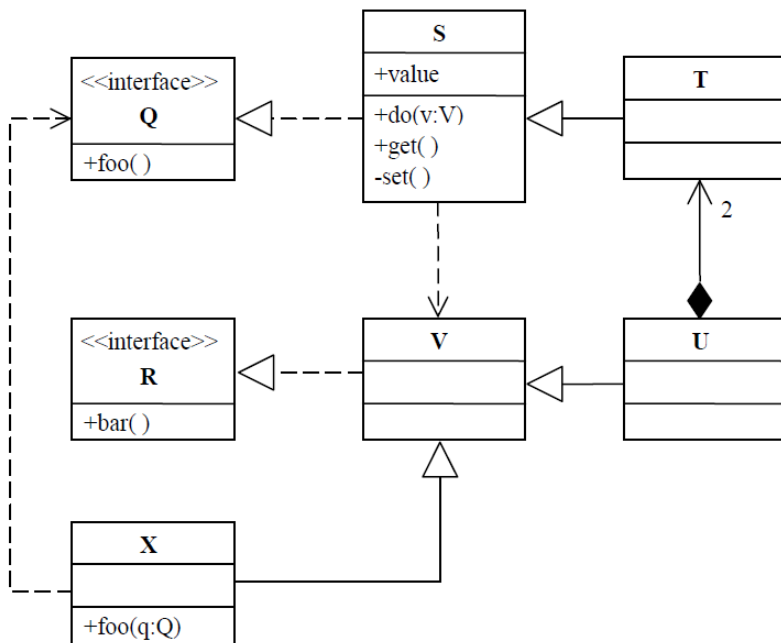
Válasz	Állítás	1.	2.	köv.
E	Q bárhol helyettesíthető S-sel, mert S a Q leszármazottja. <i>S nem leszármazottja Q-nek, tehát hamis. Nem nyert, S nem leszármazottja Q-nak.</i>	-	-	
A	R qux(q:Q) metódusa nem kaphat paraméterül Z objektumot, mert a metódus absztrakt. <i>Z nem leszármazottja Q-nak, így ez igaz. Nem abstract, hanem static, hamis.</i>	+	-	
E	B foo(r:R) metódusa nem hívhat meg a paraméterül kapott objektumon foo() metódust, mert R-nek nincs ilyen metódusa. <i>R-nek van foo()-ja (implementálja közvetetten X-t), és publikus, így meghívhatja, tehát hamis mind2 állítás.</i>	-	-	
E	S bar(q:Q) metódusa nem módosíthatja az S inf attribútumát, mert az attribútum konstans. <i>1. tagmondat triviálisan hamis, a 2. meg pusztán baromság, hamis az is.</i>	-	-	
E	S bar(q:Q) metódusa nem hívhatja meg a paraméterül kapott objektum bar() metódusát, mert a Q osztálynak nincs ilyen szignatúrájú metódusa. <i>S asszociál Q-val, és a metódus public, így semmi akadálya, tehát hamis. 2. tagmondat triviálisan hamis.</i>	-	-	
B	Z set(x:X) metódusa nem kaphat paraméterül B objektumot, mert B megvalósítja az X interfészt. <i>B megvalósítja az X interfészt, így kaphat a metódus B-t, tehát hamis. B (közvetetten) megvalósítja az X interfészt, szóval ez igaz.</i>	-	+	
B	B módosíthatja egy Q objektum im attribútumát, mert S függ Q-tól. <i>Az attribútum privát, így ez hamis. Mivel az asszociáció egyben függés is, így ez igaz.</i>	-	+	
B	R és B bárhol felcserélhetők, mert közös az ősük. <i>Egyik se leszármazottja a másiknak, így ez hamis. A 2. tagmondat triviálisan igaz.</i>	-	+	

2009.01.13 – 1. Feladat



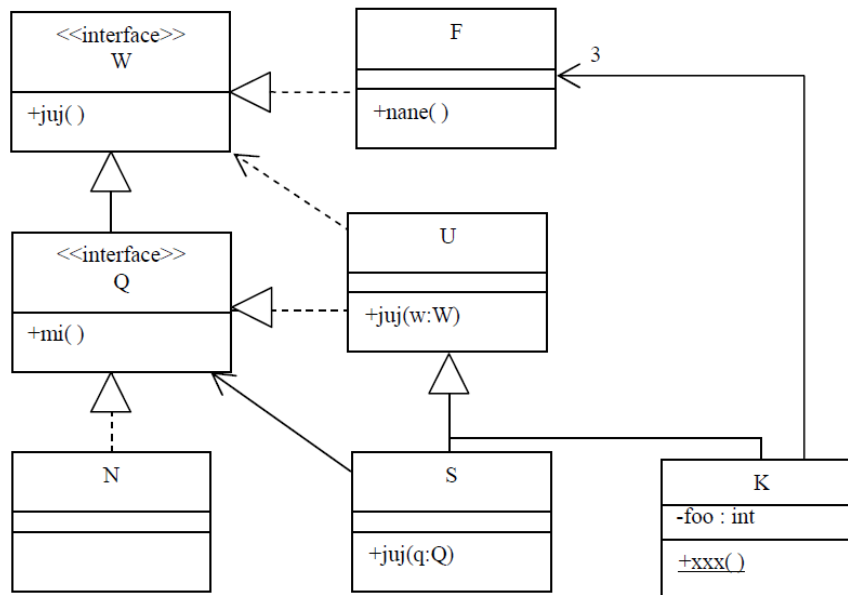
Válasz	Állítás	1.	2.	köv.
D	H bármikor helyettesíthető R-rel, mert R a H leszármazottja. <i>Az állítás megmagyarázza magát. Ámen.</i>	+	+	+
A	W nem módosíthatja K inf attribútumát, mert az attribútum protected. <i>Az attribútum privát, így ez igaz, a 2. tagmondat meg triviálisan hamis.</i>	+	-	
E	F bar(r:R) metódusa nem hívhatja meg a qux(q:Q) metódust, mert az utóbbi statikus. <i>Bármilyen láthatóságú példány metódus bármilyen láthatóságú metódust meghívhat [osztályon belül], így ez hamis. 2. tagmondat triviálisan hamis.</i>	-	-	
B	F qux(q:Q) meghívhatja a bar(r:R) metódust a q paraméterrel, mert az R megvalósítja a Q interfészt. <i>Hamis, nem írható R helyére Q (csak Q helyére írható R). R (közvetetten) megvalósítja Q-t, így ez igaz.</i>	-	+	
E	Az ábrán szereplő összes egy-paraméteres bar metódus kaphat R objektumot paraméterül, mert R megvalósítja az összes említett metódust. <i>Csak azok a 1 paramétere bar metódusok kaphatnak R-t, amelyek olyan típust várnak, amit R örököl/implementál. A 3 közül a bar(k:K) kilóg, így ez hamis. R-nek csak bar(k:K) metódusa van, így a többi nincs megvalósítva, hamis (ha jól értem, erre gondolnak, de nem biztos).</i>	-	-	
E	W bar(k:K) metódusa nem módosíthatja az osztály im attribútumát, mert az attribútum konstans. <i>Triviálisan hamis. A 2. tagmondat meg még mindig baromság.</i>	-	-	
C	W rendelkezik foo() szignatúrájú metódussal, mert függ K-tól. <i>Ja, mert implementálja a Q-t. És igen, függ K-tól, csak a kettőnek nincs köze egymáshoz.</i>	+	+	-
D	Egy F objektum meghívhatja saját magával mint paraméterrel egy R foo(k:K) metódusát, mert F a K leszármazottja. <i>(TODO) Az 1. tagmondatot nem értem...a foo(k:K) kaphat paraméterül F-t? Ez a kérdés, csak nyakatekertén? De amúgy a 2. tagmondat persze triviálisan igaz, és ez esetben a következtetés is igaz.</i>	+	+	+

2009.01.27 – 1. Feladat



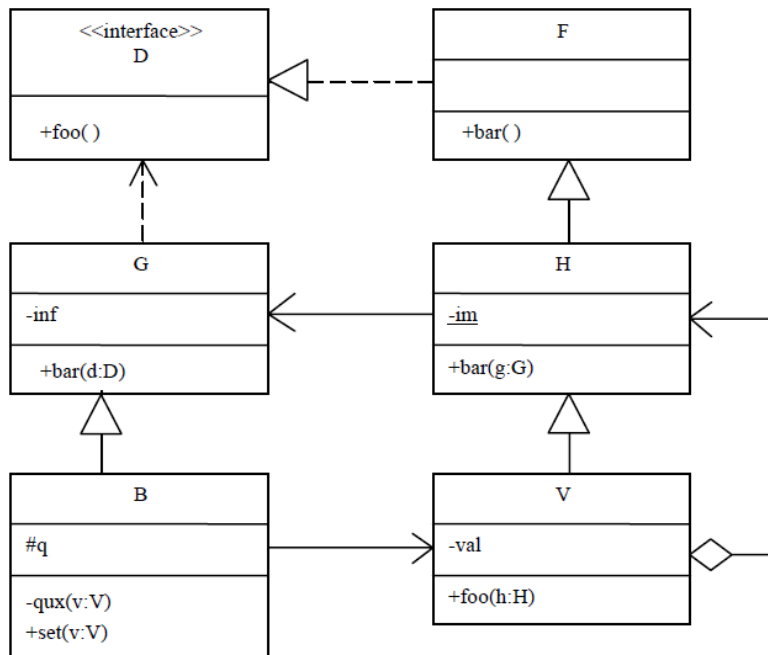
Válasz	Állítás	1.	2.	köv.
A	S létrehozhat V osztályú objektumot, mert V függ az S-től. <i>Mivel S függ V-től, így ez igaz. Nem nyert, S függ V-től, és nem fordítva.</i>	+	-	
D	X foo(q:Q) metódusa kaphat paraméterül T-t, mert T megvalósítja a Q interfészt. <i>Az állítás magyarázza magát.</i>	+	+	+
B	X foo(q:Q) metódusa meghívhatja a paraméterül kapott S get() metódusát, mert S megvalósítja a Q interfészt. <i>Bár Q helyére kerülhet S (mert S megvalósítja a Q interfészt, ld. 2. tagmondat igaz), viszont csak olyan függvényt hívhatunk vele, amit a Q-ban is van, ez viszont nem igaz a get() metódusra, így az 1. tagmondat hamis.</i>	-	+	
D	T-ből legalább kétszer annyi példány van, mint U-ból, mert egy T példány nem tartozhat két U-hoz. <i>Az diagramról pont ez olvasható le, hogy minden U-ban van 2db T (persze lehetnek önállóan létező T-k is), viszont 1 T-hez nem tartozhat 2 U, mert az jelölve lenne.</i>	+	+	+
B	T meghívhatja U bar() metódusát, mert U-nak van bar () metódusa. <i>Nope, T nem ismeri U-t. És persze U-nak van bar()-ja, mivel implementálja R-t.</i>	-	+	
A	X meghívhatja egy Q interfészes objektum foo() metódusát, mert X implementálja Q-t. <i>Igaz, semmi akadálya (X függ Q-tól, foo() metódus public). A 2. tagmondat triviálisan hamis.</i>	+	-	
C	V helyettesíthető U-val, mert mindketten megvalósítják az R interfészt. <i>Igaz, mivel U leszármazottja V-nek. És az is igaz, hogy mindketten megvalósítják az R interfészt, csak a következtetés nem igaz.</i>	+	+	-
B	S set() metódusa nem módosíthatja a value attribútumot, mert a láthatóságuk különböző. <i>Hamis, példány metódus bármikor módosíthatja az osztály bármilyen láthatóságú (akár static, akár nem) attribútumát. A 2. tagmondat triviálisan igaz.</i>	-	+	

2009.05.28 – 1. Feladat



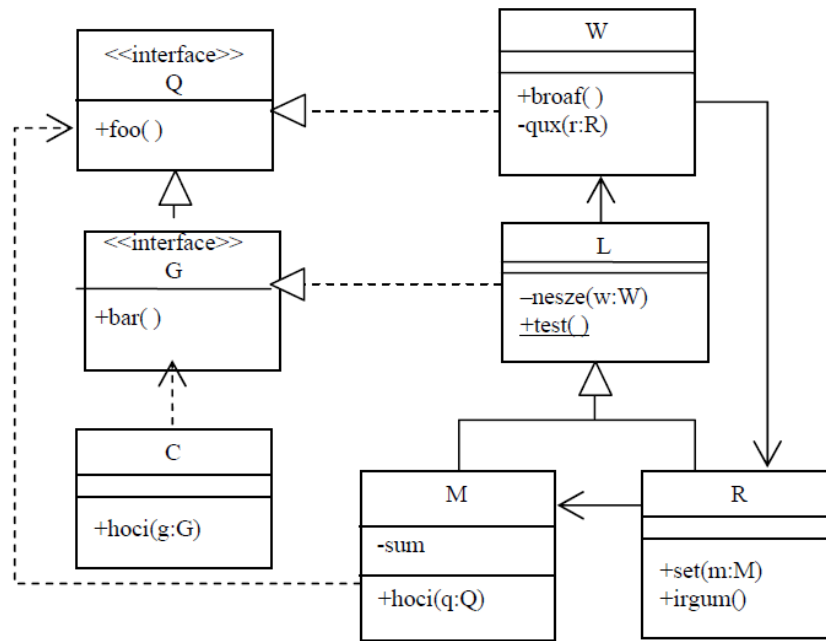
Válasz	Állítás	1.	2.	köv.
B	K helyettesíthető S-sel, mert közös ősük U. <i>S nem leszármazottja K-nak, tehát hamis. 2. triviálisan igaz.</i>	-	+	
A	K júj(w:W) metódusa kaphat paraméterül S-t, mert K az F leszármazottja. <i>S implementálja W-t, így ez igaz. 2. tagmondat nem nyert, hamis.</i>	+	-	
B	K xxx() metódusa módosíthatja bármely K objektum foo attribútumát, mert a metódus statikus. <i>Nem szeretjük, ha statikus metódus példány attribútumot piszkál, ez hamis. 2. triviálisan igaz.</i>	-	+	
B	F nem implementálja a júj() metódust, mert nem U leszármazottja. <i>Hamis, ha nem implementálná, akkor lenne abstract metódusa, és akkor az osztály is abstract lenne, tehát dőlttel lenne írva a neve, vagyis ez hamis. 2. triviálisan igaz.</i>	-	+	
C	S júj(q:Q) metódusában meghívható egy paraméterül kapott N objektum mi() metódusa, mert N megvalósítja a W interfészt. <i>N implementálja a Q interfészt, így kapható paraméterként, továbbá a Q interfész egyik (public) metódusát akarjuk elérni, ennek semmi akadálya, ez igaz. A 2. triviálisan igaz. Következtetés hamis, akkor lenne igaz, ha a Q-ra kérdeztek volna rá.</i>	+	+	-
E	S-nek nincs júj(w:W) metódusa, mert a júj(q:Q) metódusnak ugyanaz a szignatúrája. <i>Öröklí U-t, miért ne lenne? Hamis. Akkor azonos 2 metódus szignatúrája, ha azonos a nevük ÉS a paraméterlistájuk. Különbözik a paraméterlista, így ez hamis.</i>	-	-	
E	F helyettesíthető U-val, mert K mindkettejük leszármazottja. <i>U nem leszármazottja F-nek, így ez hamis. K csak U-nak a leszármazottja, F-é nem, így ez is hamis.</i>	-	-	
E	U júj(w:W) metódusából meghívhatjuk egy paraméterül kapott F nane() metódusát, mert F megvalósítja a Q interfészt. <i>Bár a W interfészt megvalósítja F, így júj(w:W) kaphat F-t, viszont csak olyan metódust hívhatunk, ami van W-nek. Ez nem igaz nane()-re, így ez hamis. F csak a W-t valósítja meg, Q-t nem, így ez is hamis.</i>	-	-	

2009.06.11 – 1. Feladat



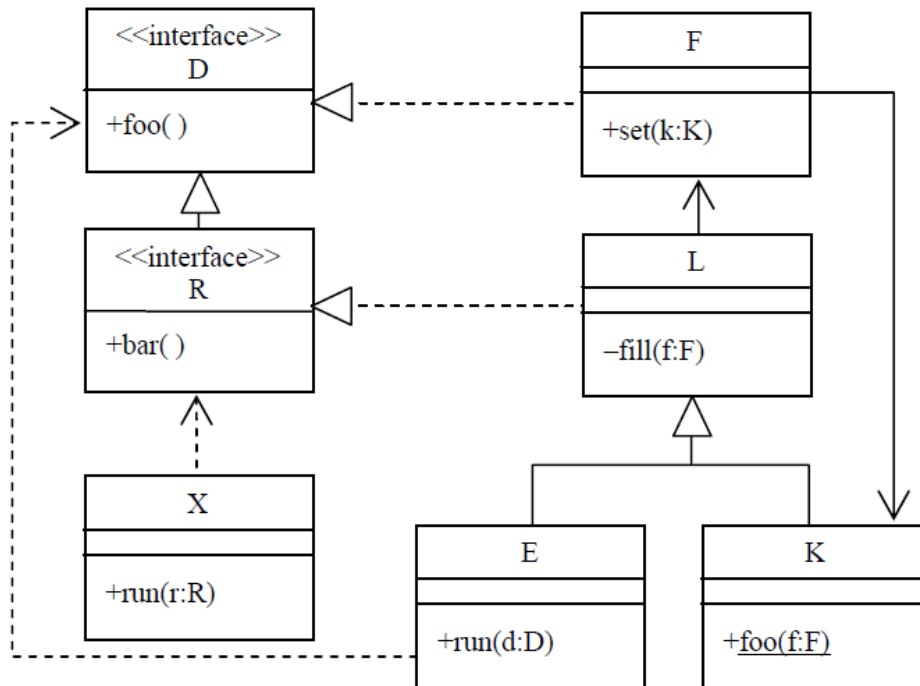
Válasz	Állítás	1.	2.	köv.
E	G bar(d:D) metódusa kaphat paraméterül B objektumot, mert G a B leszármazottja. <i>B nem implementálja D-t, így ez hamis. G a B-nek őse, triviálisan hamis.</i>	-	-	
B	H bar(g:G) metódusa kaphat paraméterül V objektumot, mert V megvalósítja a D interfészt. <i>V nem leszármazottja G-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	B qux(v:V) metódusa módosíthatja a paraméter val attribútumát, mert mind a metódus, mind az attribútum privát. <i>Az attribútum privát, így ezt B bebiztosítja, hamis. 2. triviálisan igaz.</i>	-	+	
E	H bar(g:G) metódusa nem módosíthatja az im attribútumot, mert az attribútum konstans. <i>De, módosíthatja, példány metódusból módosíthatunk static attribútumot, így ez hamis. Már megint ez a konstans baromság...hamis.</i>	-	-	
E	B objektum nem hívhatja meg egy V objektum foo() metódusát, mert V-nek nincs ilyen szignatúrájú metódusa. <i>Mivel köztük asszociáció van V felé, és a metódus public, így meghívhatja, ez hamis. V implementálja D-t, és ott bújuk meg a foo() metódus, így ez is hamis.</i>	-	-	
E	G bar(d:D) metódusa meghívhatja egy paraméterül kapott F objektum bar() metódusát, mert a két metódus azonos szignatúrájú. <i>Bár F megvalósítja a D interfészt, és így a bar(d:D) kaphat F-t, viszont csak olyan metódust hívhatunk, ami D-ben szerepel (ez bar()-ra nem igaz), így ez hamis. A 2 szignatúra nem azonos (név egyezik, de a paraméter lista nem), hamis.</i>	-	-	
B	B set(v:V) metódusa nem módosíthatja a q attribútumot, mert a láthatóságuk különböző. <i>Példány metódus bármikor módosíthatja saját osztályának bármilyen láthatóságú attribútumát (static ide vagy oda), ez hamis. 2. triviálisan igaz.</i>	-	+	
E	B-nek van foo() szignatúrájú metódusa, mert B megvalósítja a D interfészt. <i>B nem implementálja D-t, így nincs neki, mind2 hamis.</i>	-	-	

2009.06.18 – 1. Feladat



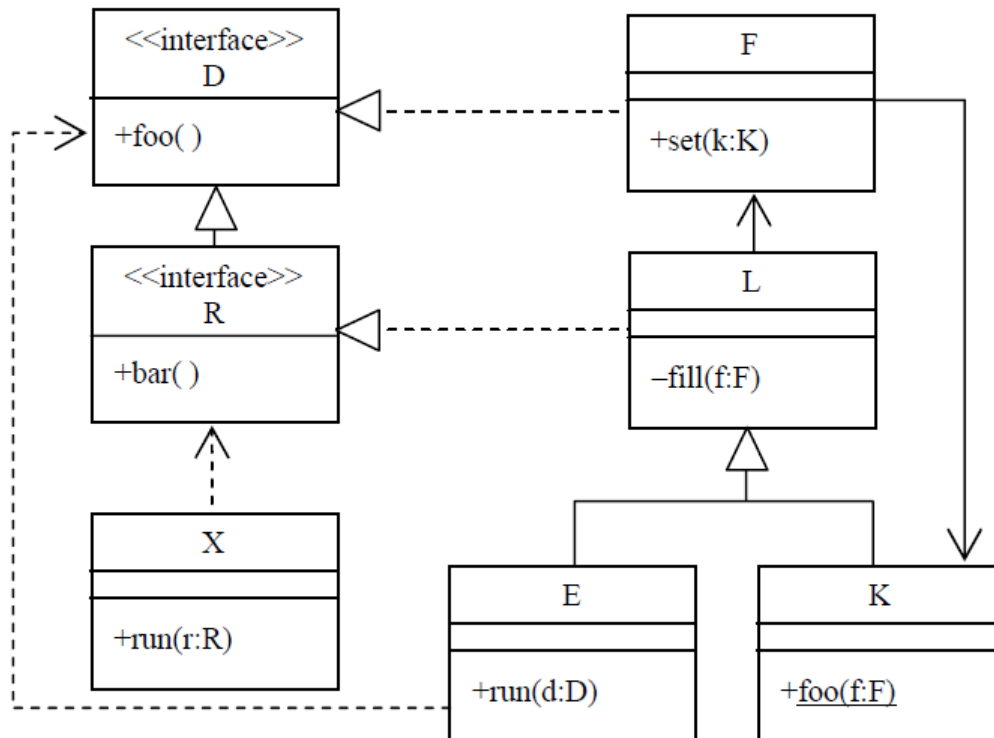
Válasz	Állítás	1.	2.	köv.
E	M hoci(q:Q) függvénye meghívhatja egy paraméterül kapott W broaf() metódusát, mert a broaf metódus statikus. <i>Bár W implementálja a Q interfészt, és így a hoci kaphat W-t is paraméterül, viszont csak a Q metódusait lehet hívni, így ez hamis. 2. tagmondat triviálisan hamis.</i>	-	-	
B	R set(m:M) metódusa kaphat paraméterül L objektumot, mert M az L leszármazottja. <i>L nem leszármazottja M-nek (hanem őse), így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	L nesze(w:W) metódusa meghívhatja a paraméterül kapott objektum qux(r:R) metódusát, mert mindkét metódus privát. <i>A qux metódus privát, így azt L nem hívhatja meg, hamis. 2. triviálisan igaz.</i>	-	+	
B	W bárhol helyettesíthető L-lel, mert mindketten megvalósítják a Q interfészt. <i>L nem leszármazottja W-nek, így ez hamis. Mindketten megvalósítják a Q interfészt, még ha L csak közvetetten is, így ez igaz.</i>	-	+	
E	R -nek nincs foo() szignatúrájú metódusa, mert nem valósítja meg a G interfészt. <i>Van, mert megvalósítja a Q interfészt (bár elég közvetetten), tehát ez hamis. Megvalósítja szerencsétlen R még a G-t is (ennek hála implementálja a Q-t), hamis.</i>	-	-	
A	C hoci(g:G) metódusa kaphat paraméterül M objektumot, mert M hoci(q:Q) metódusa is kaphat paraméterül C-t. <i>M megvalósítja közvetetten a G interfészt, így ez igaz. A C nem implementálja Q-t, így M hoci(q:Q) nem kaphat C-t, vagyis hamis.</i>	+	-	
B	L nesze(w:W) metódusa nem hívhatja meg a test() metódust, mert a test() statikus. <i>Példány metódus meghívhat static metódust, így ez hamis. 2. triviálisan igaz.</i>	-	+	
A	W qux(r:R) metódusából bármikor meghívható a paraméter irgum() metódusa, mert a két osztály nem függ egymástól. <i>Ott az asszociáció W-ből R-be, a metódus public, így semmi akadálya, igaz. W függ R-től így a 2. hamis.</i>	+	-	

2010.01.05 (A) – 1. Feladat



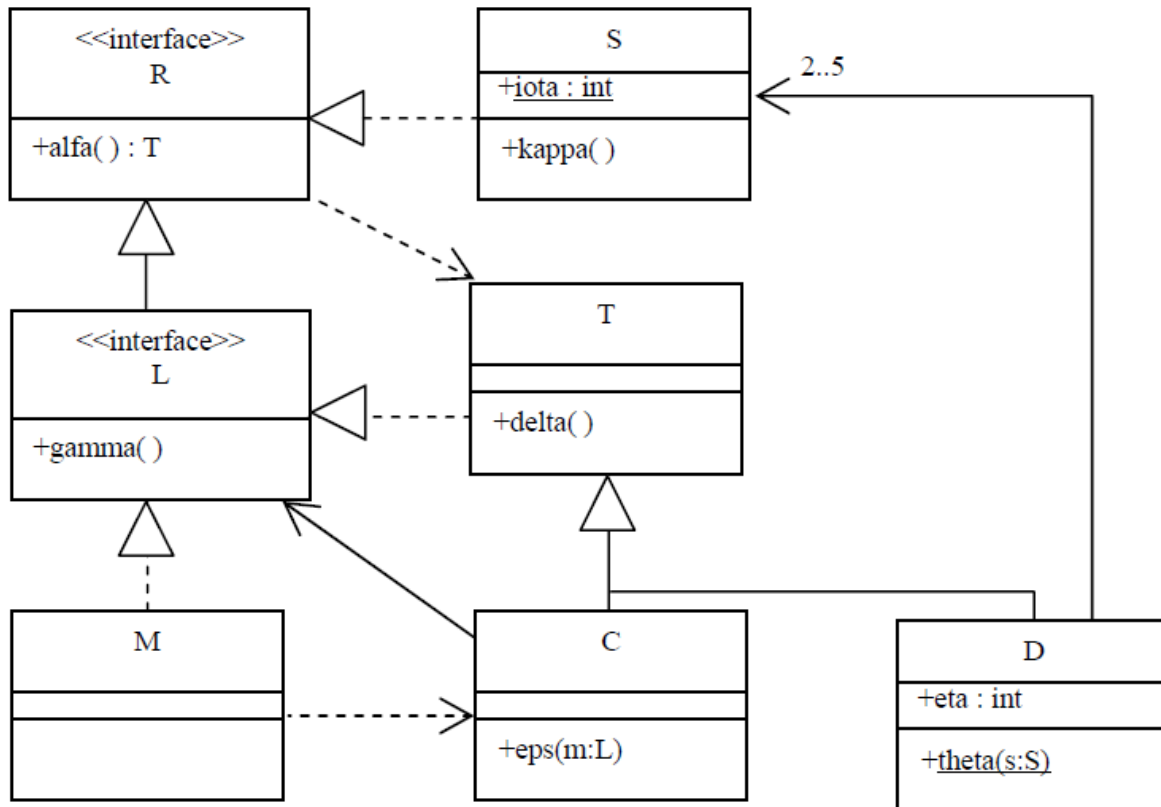
Válasz	Állítás	1.	2.	köv.
B	E bárhol helyettesíthető K-val, mert van közös ősök. <i>K nem leszármazottja E-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	L bárhol helyettesíthető F-fel, mert mindketten megvalósítják a D interfészt. <i>F nem leszármazottja F-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	L nem helyettesíthető E-vel, mert L-nek van privát metódusa. <i>E leszármazottja L-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	F set(k:K) metódusa meghívhatja egy paraméterül kapott K fill(f:F) metódusát, mert K függ F-től. <i>K-nak nincs ilyen metódusa (mert az L-ben privát), így ez hamis. 2. triviálisan igaz, hiszen foo függvénye F paramétert vár.</i>	-	+	
B	X run(r:R) metódusa kaphat paraméterül F osztályú objektumot, mert X függ R-től. <i>F nem implementálja R-t, így ez hamis. 2. triviálisan igaz.</i>	-	+	
E	K-nak nincs foo() szignatúrájú metódusa, mert K-t nem lehet példányosítani. <i>De van, implementálja a D-t, hamis. Lehet példányosítani, ez is hamis (abstract osztályt, vagy interfészt nem lehet).</i>	-	-	
B	X run(r:R) metódusa nem kaphat paraméterül K objektumot, mert K-nak van statikus metódusa. <i>K megvalósítja az R interfészt, így ez hamis. 2 triviálisan igaz.</i>	-	+	
B	K foo(f:F) metódusa nem hívhatja meg a paraméter foo() metódusát, mert az utóbbi metódus nem statikus. <i>Semmi akadálya, hogy meghívja, hamis. 2. triviálisan igaz.</i>	-	+	

2010.01.05 (B) – 1. Feladat



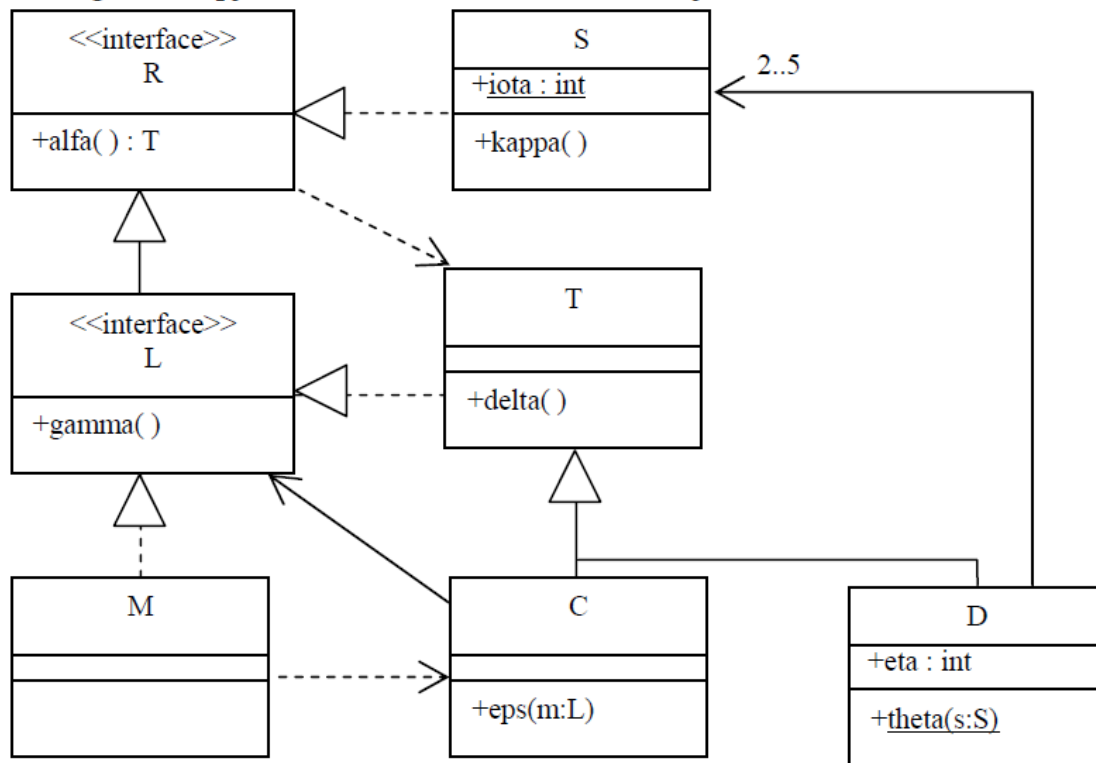
Válasz	Állítás	1.	2.	köv.
B	X run(r:R) metódusa kaphat paraméterül F osztályú objektumot, mert X függ R-től. <i>F nem implementálja R-t, így ez hamis. 2. triviálisan igaz.</i>	-	+	
E	K-nak nincs foo() szignatúrájú metódusa, mert K-t nem lehet példányosítani. <i>K implementálja D-t, így ez hamis. Lehet példányosítani K-t, ez is hamis (abstract osztályt, vagy interfészt nem lehet).</i>	-	-	
E	L bárhol helyettesíthető F-fel, mert mindketten megvalósítják az R interfészt. <i>F nem leszármazottja L-nek, ez hamis. Csak L valósítja meg az R interfészt, F nem, így ez is hamis.</i>	-	-	
B	L nem helyettesíthető E-vel, mert L-nek van privát metódusa. <i>E leszármazottja L-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	X run(r:R) metódusa nem kaphat paraméterül K objektumot, mert K-nak van statikus metódusa. <i>K implementálja R interfészt, így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	K foo(f:F) metódusa nem hívhatja meg a paraméter foo() metódusát, mert az utóbbi metódus nem statikus. <i>Semmi akadálya, hogy meghívja, hamis. 2. triviálisan igaz.</i>	-	+	
B	E bárhol helyettesíthető K-val, mert van közös ősük. <i>K nem leszármazottja E-nek, így ez hamis. 2. triviálisan igaz.</i>	-	+	
C	F set(k:K) metódusa nem hívhatja meg egy paraméterül kapott K fill(f:F) metódusát, mert K függ F-től. <i>K-nak nincs fill(f:F) metódusa (L-nél privát, nem örökli), így ez igaz. foo(f:F) miatt K fütt F-től, ez is igaz. A 2-nek persze kb. köze sincs egymáshoz, vagyis a következtetés hamis.</i>	+	+	-

2010.01.12 (A) – 1. Feladat



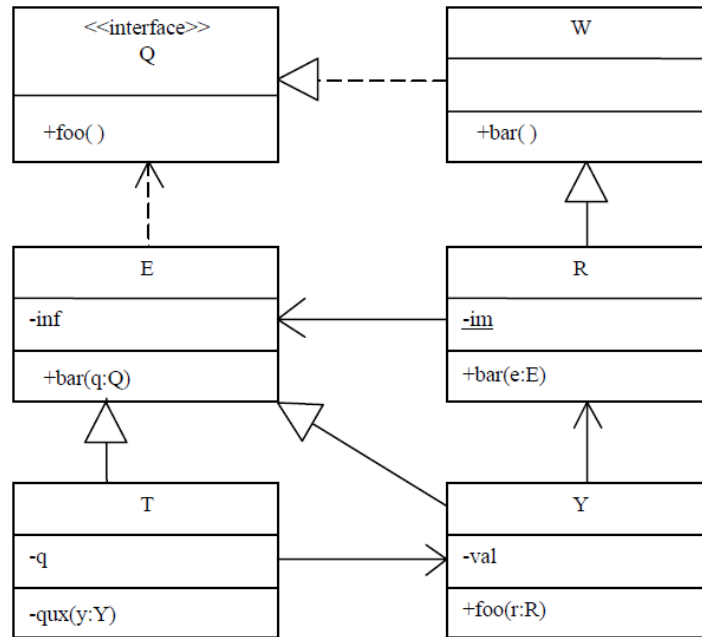
Válasz	Állítás	1.	2.	köv.
B	M bárhol helyettesíthető C-vel, mert mindketten megvalósítják az R interfészt. <i>C nem leszármazottja M-nek, tehát hamis. 2. triviálisan igaz.</i>	-	+	
E	C eps(m:L) metódusa nem hívhatja meg a paraméter gamma() metódusát, mert az utóbbi metódus protected láthatóságú. <i>Triviálisan hamis mind2.</i>	-	-	
C	D theta(s:S) metódusa módosíthatja a paraméter iota attribútumát, mert a theta(s:S) statikus. <i>A itoa attr. publikus...why not, igaz. 2. triviálisan igaz, de a következtetés hamis, minden bizonnyal pont azért, mert az publikus.</i>	+	+	-
E	T osztálynak nincs alfa():T szignatúrájú metódusa, mert T nem valósítja meg az R interfészt. <i>T implementálja R-t, így ez hamis. Nem a faszt nem, hamis.</i>	-	-	
A	M alfa():T metódusa visszaadhat C objektumot, mert C függ M-től. <i>C T-nek a leszármazottja, így ez igaz. C nem függ az M-től (jelölve sincs), hamis.</i>	+	-	
B	D theta(s:S) metódusa kaphat paraméterül C objektumot, mert S és C is megvalósítja az R interfészt. <i>C nem leszármazottja S-nek, így ez hamis. A 2. triviálisan igaz.</i>	-	+	
E	S nem valósítja meg az alfa():T szignatúrájú metódust, mert S nem függ T-től. <i>Hamis, ha nem tenné, akkor abstract lenne. Függ, mivel R-t örökli, aki függ T-től, így ez is hamis.</i>	-	-	
B	D theta(s:S) metódusa legfeljebb 5-ször hívható meg, mert D objektum legfeljebb 5 S-sel állhat asszociációban. <i>Ez egy ritka nagy baromság, hamis. 2. triviálisan igaz.</i>	-	+	

2010.01.12 (B) – 1. Feladat



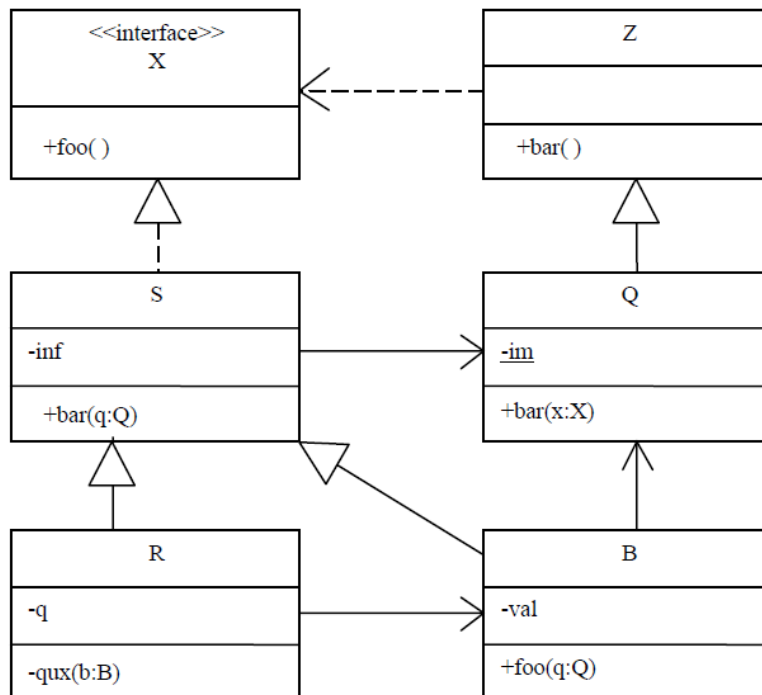
Válasz	Állítás	1.	2.	köv.
A	M alfa():T metódusa visszaadhat C objektumot, mert C függ M-től. <i>C a T-nek leszármazottja, így ez igaz. 2. triviálisan hamis.</i>	+	-	
C	D theta(s:S) metódusa nem kaphat paraméterül C objektumot, mert S és C is megvalósítja az R interfészt. <i>C nem leszármazottja S-nek, így ez igaz. 2. triviálisan igaz. A kettőnek viszont köze sincs egymáshoz.</i>	+	+	-
E	C eps(m:L) metódusa nem hívhatja meg a paraméter gamma() metódusát, mert az utóbbi metódus protected láthatóságú. <i>Triviálisan hamis mind2.</i>	-	-	
B	D theta(s:S) metódusa nem módosíthatja a paraméter iota attribútumát, mert a theta(s:S) statikus. <i>Semmi akadálya, Just do it, hamis. 2. triviálisan igaz.</i>	-	+	
E	S nem valósítja meg az alfa():T szignatúrájú metódust, mert S nem függ T-től. <i>Hamis, ha nem tenné, akkor abstract lenne. Függ, mivel R-t örökli, aki függ T-től, így ez is hamis.</i>	-	-	
B	D theta(s:S) metódusa legfeljebb 5-ször hívható meg, mert D objektum legfeljebb 5 S-sel állhat asszociációban. <i>Ez egy ritka nagy baromság, hamis. 2. triviálisan igaz.</i>	-	+	
B	M bárhol helyettesíthető C-vel, mert mindketten megvalósítják az R interfészt. <i>C nem leszármazottja M-nek, tehát hamis. 2. triviálisan igaz.</i>	-	+	
D	T osztálynak van alfa():T szignatúrájú metódusa, mert T megvalósítja az R interfészt. <i>Hát igen, pont ezért.</i>	+	+	+

2010.01.26 – 1. Feladat



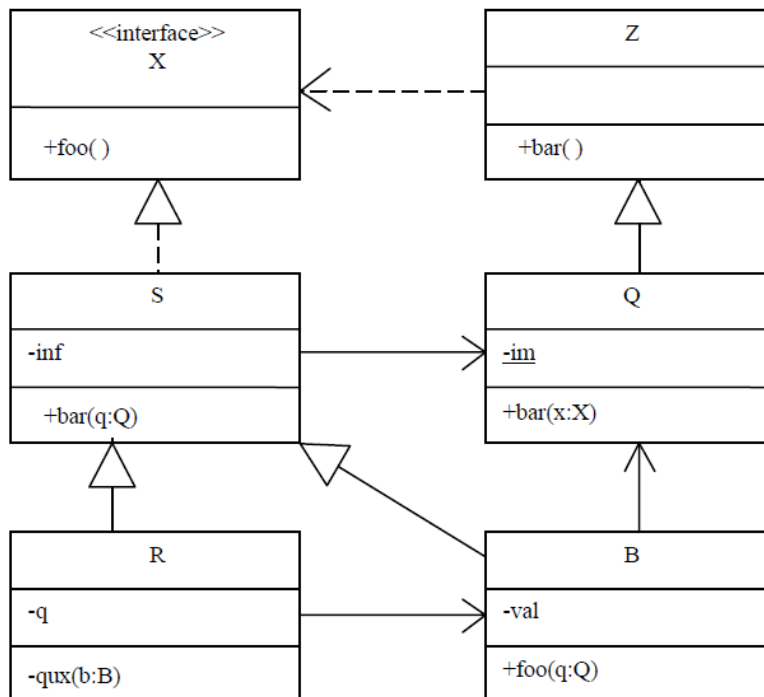
Válasz	Állítás	1.	2.	köv.
C	Y bar(q:Q) metódusa kaphat paraméterül R objektumot, mert Y függ R-től. <i>R megvalósítja a Q interfészt, így ez igaz. 2. triviálisan igaz. A következtetés meg bullshit.</i>	+	+	-
B	T qux(y:Y) metódusa módosíthatja a paraméter val attribútumát, mert a metódus privát. <i>Az attr. bizony privát, úgyhogy ez felejtős T-nek, hamis. 2. triviálisan igaz.</i>	-	+	
E	E bárhol helyettesíthető R-rel, mert azonos az interfészük. <i>R nem leszármazottja E-nek, így ez hamis. R-nek van bar(e:E) metódusa, E-nek nincs, már rögtön rá is vágthatjuk, hogy hamis.</i>	-	-	
C	Y foo(r:R) metódusa nem módosíthatja a paraméter im attribútumát, mert az attribútum statikus. <i>Igaz (ld. 2. kérdés). 2. triviálisan igaz. Don't give a fuck, hogy statikus, csak ne lenne privát.</i>	+	+	-
E	E bar(q:Q) metódusa kaphat E objektumot paraméterül, mert az E megvalósítja a Q interfészt. <i>E nem valósítja meg a Q interfészt (csak függ tőle), így ez hamis. Amint azt már mondtam, nem valósítja meg, így ez is hamis.</i>	-	-	
E	E bar(q:Q) metódusa nem hívhatja meg egy paraméterül kapott W foo() metódusát, mert W-nek nincs ilyen szignatúrájú metódusa. <i>W implementálja Q-t, így kapható paraméterként, és „Q-ban levő” foo() metódust meghívhatjuk, így ez hamis. W implementálja Q-t, így van neki, ez is hamis.</i>	-	-	
B	R bar(e:E) metódusa nem kaphat paraméterül Y objektumot, mert az Y-R asszociációban csak Y hívhatja R-t. <i>Y az E-nek leszármazottja, így ez hamis. 2. (triviálisan) igaz.</i>	-	+	
B	R nem valósítja meg a Q interfészt, mert van olyan szignatúrájú metódusa, ami nem szerepel a Q metódusai között. <i>Ne hazudj, megvalósítja, hamis. Van, mégpedig a bar(e:E), mivel a paraméterlista nem egyezik a bar() paraméterlistájával, igaz.</i>	-	+	

2011.01.04 (A) – 1. Feladat



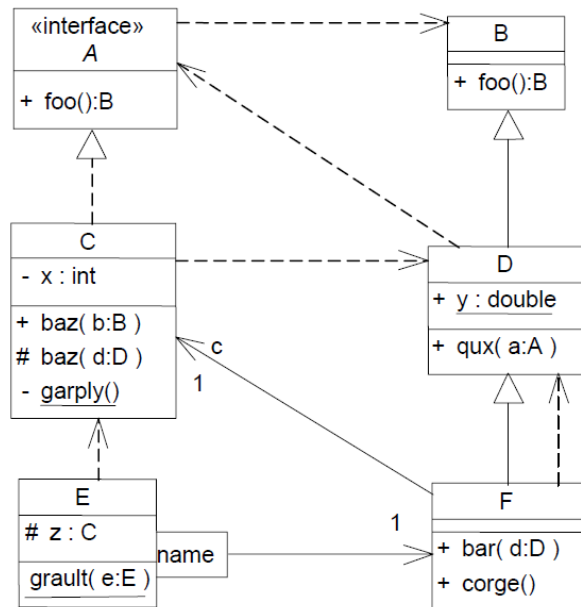
Válasz	Állítás	1.	2.	köv.
E	Q helyettesíthető S-sel, mert S a Q leszármazottja. <i>S nem leszármazottja Q-nak, így hamis. Ez sem nyert, hamis.</i>	-	-	
C	S helyettesíthető B-vel, mert B megvalósítja az X interfészt. <i>B leszármazottja S-nek, így ez igaz. 2. triviálisan igaz, viszont nem emiatt lesz igaz az 1., hanem mert B leszármazottja S-nek.</i>	+	+	-
A	R átadható paraméterül Q bar(x:X) metódusának, mert Q és S interfésze megegyezik. <i>R megvalósítja az X interfészt, így ez igaz. 2. triviálisan hamis (egyiknek bar(x:X)-je, másiknak bar(q:Q)-ja van).</i>	+	-	
B	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát. <i>Triviálisan hamis. (ha ő nem látná, ki más láthatná? :O). Hamis. 2. triviálisan igaz.</i>	-	+	
E	B interfésze tartalmazza a bar(x:X) metódust, mert a metódus statikus. <i>Hamis, B nem leszármazottja Q-nak. 2. triviálisan hamis.</i>	-	-	
E	Q meghívhatja S bar(q:Q) metódusát, mert mindketten megvalósítják az X interfészt. <i>Q-nak fogalma sincs S létezéséről (semmilyen függés, még „tranzitív” sem), így ez hamis. Az X nem valósítja meg, csak függ tőle, hamis ez is.</i>	-	-	
A	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum konstans. <i>Mivel Q nem override-olja a bar() metódust, és az attr. privát, így ez igaz. De rég is volt ez a konstansos baromság...és még mindig hamis.</i>	+	-	
A	Q nem implementálja a foo() metódust, ezért nem függ az X interfésztől. <i>Igaz, mivel nem implementálja az X interfészt. Ettől függetlenül függ (ld. bar(x:X) függvénye, ill. mivel Z is függ X-től, így ez kihat Q-ra is), hamis.</i>	+	-	

2011.01.04 (B) – 1. Feladat



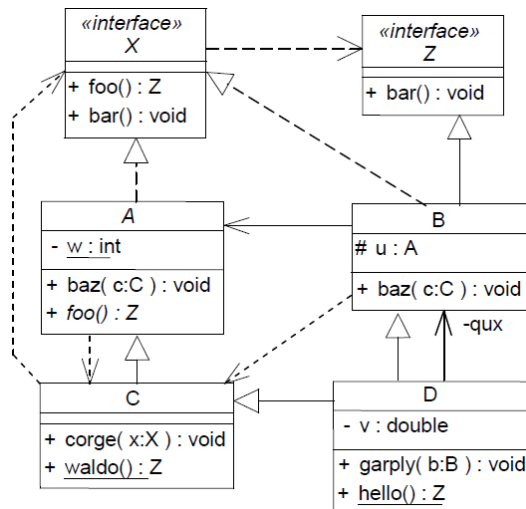
Válasz	Állítás	1.	2.	köv.
B	R helyettesíthető B-vel, mert R függ B-től. <i>B nem leszármazottja R-nek, tehát hamis. 2. triviálisan igaz, ott az asszociáció.</i>	-	+	
E	Q meghívhatja S bar(q:Q) metódusát, mert mindketten megvalósítják az X interfészt. <i>Q-nak fogalma sincs S létezéséről (semmilyen függés, még „tranzitív” sem), így ez hamis. A Q nem valósítja meg, csak függ tőle, hamis ez is.</i>	-	-	
E	Q helyettesíthető S-sel, mert S a Q leszármazottja. <i>S nem leszármazottja Q-nak, így hamis. Ez sem nyert, hamis.</i>	-	-	
E	B interfésze tartalmazza a bar(x:X) metódust, mert a metódus statikus. <i>Hamis, B nem leszármazottja Q-nak. 2. triviálisan hamis.</i>	-	-	
B	B foo(q:Q) metódusa nem látja saját val attribútumának értékét, mert az attribútum privát. <i>Triviálisan hamis. (ha ő nem látná, ki más láthatná? :O). Hamis. 2. triviálisan igaz.</i>	-	+	
C	Q bar() metódusa nem módosíthatja az im attribútumot, ezért az attribútum statikus. <i>Mivel Q nem override-olja a bar() metódust, és az attr. privát, így ez igaz. 2. triviálisan igaz. A következtetés ezek alapján hamis.</i>	+	+	-
A	Q nem implementálja a foo() metódust, ezért nem függ az X interfésztől. <i>Igaz, mivel nem implementálja az X interfészt. Ettől függetlenül függ (ld. bar(x:X) függvénye, ill. mivel Z is függ X-től, így ez kihat Q-ra is), hamis.</i>	+	-	
A	B átadható paraméterül Q bar(x:X) metódusának, mert Q és S interfésze megegyezik. <i>B megvalósítja az X interfészt, így ez igaz. 2. triviálisan hamis (egyiknek bar(x:X)-je, másoknak bar(q:Q)-ja van).</i>	+	-	

2011.01.18 – 1. Feladat



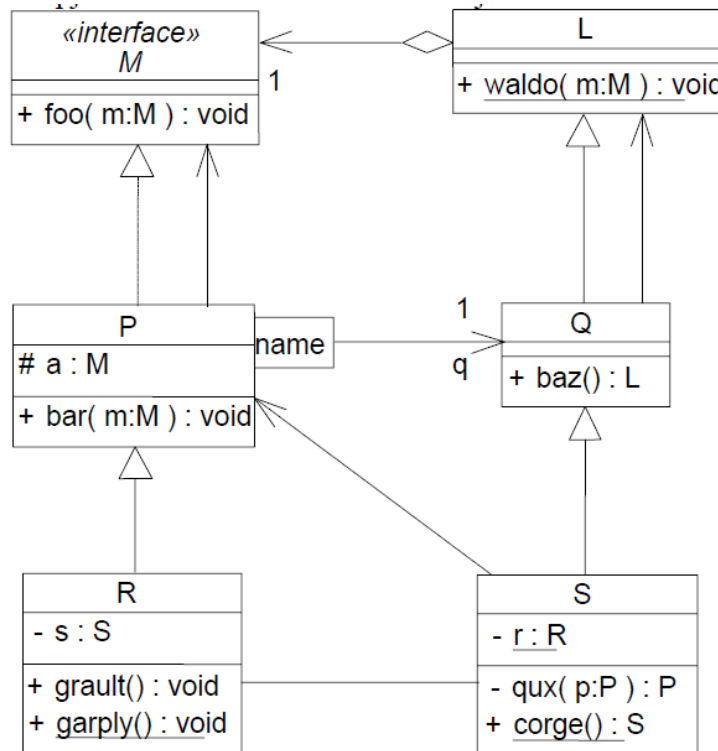
Válasz	Állítás	1.	2.	köv.
B	D implementálja az A interfészt, mert A és B interfésze megegyezik. <i>1. triviálisan hamis. Mindkettőnek csak foo():B metódusa van, így ez igaz (netes források alapján a visszatérési érték nem a szignatúra része, így ha az egyik más osztállyal térne vissza _elvileg_ akkor is azonos lenne az interfész).</i>	-	+	
B	F corge() metódusa nem módosíthatja D y attribútumát, mert D y attribútuma statikus. <i>Publikus, így simán megteheti, hamis. 2. triviálisan igaz.</i>	-	+	
B	E grault(e:E) metódusa nem hívhatja meg e z attribútumának baz(b:B) metódusát, mert a grault() statikus. <i>A paraméterül kapott e-nk köszönhetően nincs probléma abból, hogy a grault statikus, és mivel C baz(b:B) metódusa public, így meghívható, tehát hamis (baz(d:D)-t már nem hívhatná meg, mert az protected). 2. triviálisan igaz.</i>	-	+	
E	Egy F objektum pontosan egy E objektumot ismer, mert egy E objektum pontosan egy F objektumot ismer. <i>F nem ismeri E-t, így ez hamis. 2. a qualifier miatt hamis (ld. szofttech jgyzet).</i>	-	-	
D	C foo() metódusa példányosíthat B típusú objektumot, ezért C függ B-től. <i>Az állítás megmagyarázza magát helyettem is.</i>	+	+	+
E	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. <i>F „asszociálva van” c-vel C-vel, a metódus public, így semmi akadálya, hamis. D leszármazottja B-nek, így ez is hamis.</i>	-	-	
B	C baz(d:D) metódusa nem hívhatja meg C garply() metódusát, mert C baz(d:D) metódusa nem statikus. <i>Példány metódus gond nélkül hívhat meg static metódust, így ez hamis. 2. triviálisan igaz.</i>	-	+	
A	D qux(a:A) metódusa nem hívhatja meg egy paraméterül kapott C típusú objektum baz(b:B) metódusát, mert A nem helyettesíthető C-vel. <i>Bár a metódus kaphat C-t (C imp. A-t), viszont csak C metódusai hívhatók meg, így a C metódusai nem, tehát ez igaz. C implementálja A-t, így ez hamis.</i>	+	-	

2011.05.24 – 1. Feladat



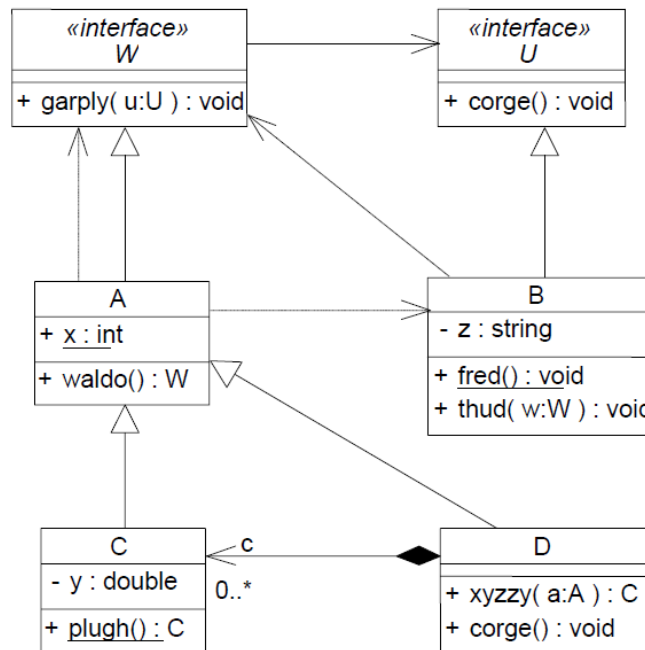
Válasz	Állítás	1.	2.	köv.
E	D garply metódusa nem módosíthatja a b paraméter u attribútumát, mert protected attribútumhoz csak privát és protected metódusok férhetnek hozzá. <i>Mivel D örökli B-t, és az attribútum protected (lehetne még publikus is, de private már nem), ezért ez hamis. Bármilyen láthatóságú attribútumhoz bármilyen láthatóságú metódus hozzáfér osztályon belül, ez triviálisan hamis.</i>	-	-	
A	C corge metódusa kaphat paraméterül D típusú objektumot, ezért a metódus meghívhatja a kapott objektum garply metódusát. <i>D megvalósítja az X interfészt (C-A-n keresztül), így ez igaz, viszont ekkor is csak X metódusai hívhatók meg, így ez hamis.</i>	+	-	
B	D garply metódusa kaphat paraméterül A típusú objektumot, mert A és B interfésze megegyezik. <i>A nem leszármazottja B-nek így ez hamis. 2. igaz (baz(c:C), bar(), foo() metódusai vannak mind2-nek).</i>	-	+	
B	C waldo metódusa virtuális, ezért a B osztály baz függvénye egy paraméterül kapott D típusú objektumon meghívhatja a waldo metódust. <i>Statikus, nem virtuális, hamis. D örökli C-t, így átadható a metódusnak paraméterként, és mivel a waldo C osztályú függvény, így meghívható (pl. garply-ra már hamis lenne), igaz.</i>	-	+	
B	A baz metódusa nem módosíthatja A w attribútumát, mert A baz metódusa nem statikus. <i>Egy osztály metódusa bármikor módosíthatja annak bármilyen láthatóságú attribútumát, akkor is, ha az static, hamis. 2. triviálisan igaz.</i>	-	+	
A	C-nek van bar metódusa, ezért C implementálja a Z interfészt. <i>Implementálja közvettem X-t, így ez igaz. 2. triviálisan hamis.</i>	+	-	
C	D hello metódusa nem módosíthatja D v attribútumát, mert D v attribútuma privát. <i>Static metódus ne akarja osztályának példány attribútumát módosítani, igaz. Az, hogy private, nem kétséges, a baj, hogy nem static.</i>	+	+	-
B	B baz metódusa nem hívhatja meg B u attribútumának foo metódusát, mert az A osztály foo metódusa absztrakt. <i>B asszociál A-val, és a metódus public, így az sem akadály, hogy a metódus abstract, hamis-igaz.</i>	-	+	

2011.06.07 – 1. Feladat



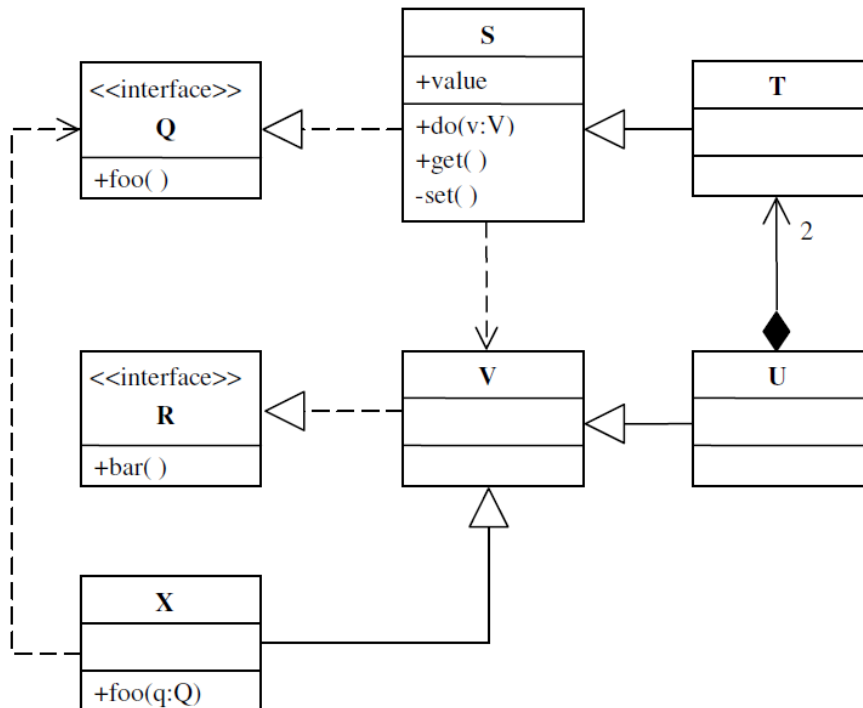
Válasz	Állítás	1.	2.	köv.
B	R garply metódusa meghívhatja az s attribútum corge metódusát, mert a garply és a corge metódus is statikus. <i>Az s attribútum nem statikus, így azzal a static garply nem dolgozhat, hamis. 2. triviálisan igaz.</i>	-	+	
B	P nem hívhat Q osztályon waldo függvényt, mert L waldo függvénye nem virtuális. <i>P asszociál Q-ra, és a waldo public, így ez hamis. 2. triviálisan igaz.</i>	-	+	
A	Q függ M-től, mert Q ismeri P-t. <i>Igen, mivel L-t örökli, aki függ M-től (asszociál, de az egyben függés is). Q nem ismeri P-t, csak P ismeri Q-t, hamis.</i>	+	-	
A	Az M típus közvetlenül nem példányosítható, ezért R grault metódusa nem hívhatja meg P a attribútumának foo metódusát. <i>Interfész nem példányosítható, tehát ez igaz. R örökli P-t, így annak protected attr.-t pont még látja, és azon keresztül eléri M foo(m:M) metódust, hamis.</i>	+	-	
B	Egy P típusú objektum pontosan egy Q típusú objektumot ismer, ezért P függ Q-tól. <i>Hamis a qualifier miatt. P asszociál Q-val, ami egyben függés is, tehát ez igaz.</i>	-	+	
A	S az L leszármazottja, ezért Q baz függvénye példányosíthat S típusú objektumot. <i>Igaz, közvetetten leszármazottja. Q nem függ S-től, nem is ismeri, így ez hamis.</i>	+	-	
D	S qux függvénye nem módosíthatja az r attribútum s attribútumát, mert R s attribútuma privát. <i>Pontosan. Teszem azt ha protected lenne, akkor is igaz lenne.</i>	+	+	+
D	S ismeri M-et, mert az S osztály L őse függ M-től. <i>Igen, pont azért. A függés is öröklődik.</i>	+	+	+

2011.06.14 – 1. Feladat



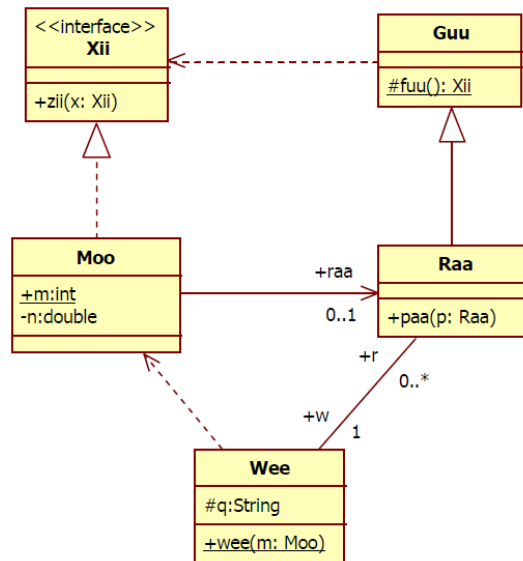
Válasz	Állítás	1.	2.	köv.
A	U interfésze részhalmaza D interfészének, ezért D megvalósítja az U interfészt. <i>U-ban minden benne van, ami D-ben is, így ez igaz. 2. triviálisan hamis.</i>	+	-	
B	D xyzy függvénye visszaadhatja eredményként a paraméterként kapott a objektumot, mert C az A leszármazottja. <i>Az A nem leszármazottja C-nek (hanem őse), így ez hamis. 2. triviálisan igaz.</i>	-	+	
B	A waldo függvénye nem példányosíthat B típusú objektumot, mert B nem implementálja a W interfészt. <i>A asszociál B-vel (tehát függ is tőle), így példányosíthat B-t, ez hamis. 2. triviálisan igaz.</i>	-	+	
E	C plugh függvénye nem módosíthatja A x attribútumát, mert A x attribútuma protected. <i>Hamis, még akkor is módosíthatná, ha protected láthatóságú lenne (ha nem lenne static, akkor már igaz lenne). Public, nem protected, hamis ez is.</i>	-	-	
E	Egy C objektum sok D objektumot tartalmaz, ezért C ismeri D -t. <i>Pont fordítva lenne igaz, D-ben van sok C, D ismeri C-t, de így hamis mind2.</i>	-	-	
C	B fred függvénye nem módosíthatja a z attribútum értékét, mert B z attribútuma nem protected. <i>Static metódus ne akarjon példány attr.-t módosítani, ez igaz. Private, nem protected, igaz ez is, de a baj, hogy nem static attr. a z.</i>	+	+	-
E	B thud függvénye meghívhatja egy paraméterül kapott C típusú objektum plugh függvényét, mert C plugh függvénye virtuális. <i>C implementálja W-t, így thud kaphat C-t is, viszont csak W interfész metódusai hívhatók meg (garply), így ez hamis. Nem virtális, static, hamis ez is.</i>	-	-	
E	C nem függ U -tól, mert C A ősztyála sem függ U -tól. <i>W asszociál U-val (így függ is tőle), ezt öröklí C is, így ez hamis. De-de, függ ő is.</i>	-	-	

2013.06.18 – 5. feladat



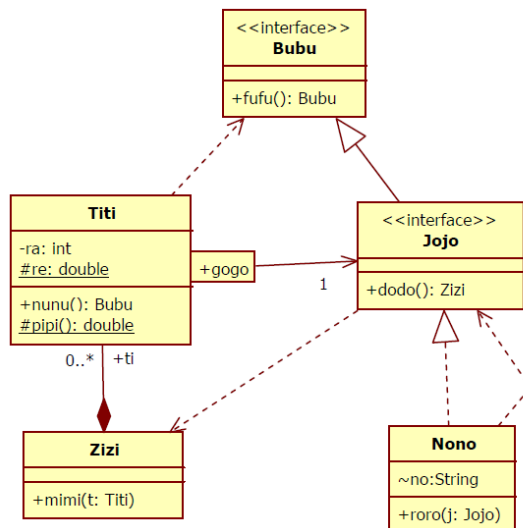
Válasz	Állítás	1.	2.	köv.
B	T létrehozhat U osztályú objektumot, mert S létrehozhat V-t és T az S-nek, U a V-nek leszármazottja. <i>Ebben nem vagyok biztos, de valószínűleg azért nem hozhat létre, mert ha tudna, akkor (legalább) függenie kellene tőle, ami az ábrán viszont nem szerepel. 2. része triviálisan igaz.</i>	-	+	
C	X foo(q:Q) metódusa kaphat paraméterül T-t, mert T-nek is van foo() metódusa. <i>Implicit megvalósítja T a Q-t, így ez igaz, és pont ezért is lesz neki foo() metódusa, de a következtetés hamis (ld. a mondat eleje).</i>	+	+	-
B	X foo(q:Q) metódusa meghívhatja a paraméterül kapott S get() metódusát, mert S megvalósítja a Q interfészt. <i>A paraméteren csak olyan függvények hívhatóak, amik az „eredeti” osztályban is szerepelnek (itt csak a foo()), így ez hamis.</i>	-	+	
B	V törlésekor törölni kell két T-t is, mert egy U-nak két T komponense van és U a V leszármazottja. <i>Kell a fenéket (miért kellene?...), a második része viszont igaz.</i>	-	+	
A	T nem függ U-tól, mert T nem függ V-től sem. <i>Jha, ábrán semmi nyoma ennek. De, S-en keresztül implicit függ.</i>	+	-	
A	X meghívhatja egy Q interfészü objektum foo() metódusát, mert X implementálja Q-t. <i>Pontosan, foo()-t azt szabad hívni. Nincs útvonal...blabla.</i>	+	-	
B	X bar() metódusából meghívhatjuk egy Q interfészü objektum foo() metódusát, mert X foo(q:Q) metódusából is hívhatjuk egy Q interfészü objektum foo() metódusát. <i>Nem, hiszen azt R-ből veszi, akinek fogalma sincs Q-ról. 2. triviálisan igaz.</i>	-	+	
B	S set() metódusa nem módosíthatja a value attribútumot, mert a láthatóságuk különböző. <i>Simán módosíthatja (itt általában csak a „static metódus ne nyúljon példány metódushoz”-ra kell odafigyelni).</i>	-	+	

2014.01.14 – 1. Feladat



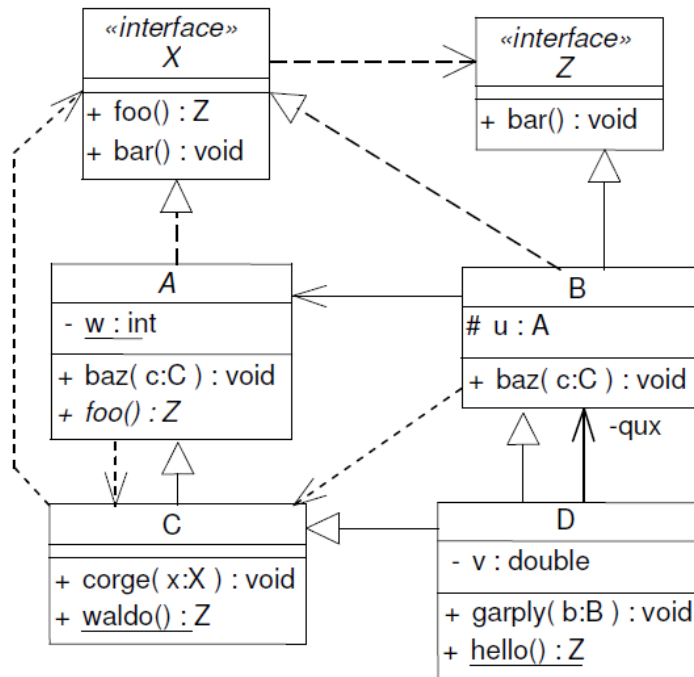
Válasz	Állítás	1.	2.	köv.
B	Raa nem függ Xii-től, mert Guu nem implementálja a Xii interfészt. <i>Raa öröklí a Xii-től való függést Guu-tól, hamis. 2. triviálisan igaz.</i>	-	+	
E	Raa paa(p:Raa) függvénye kaphat paraméterül Guu objektumot, mert függvényparaméterként Raa helyettesíthető Guu-val. <i>Guu nem leszármazottja Raa-nak, így nem helyettesíthető Raa Guu-val, mind2 hamis.</i>	-	-	
C	Raa paa(p:Raa) függvénye nem módosíthatja Moo m attribútumát, mert Moo m attribútuma statikus. <i>Raa még csak nem is ismeri Moo-t, így ez igaz. 2. triviálisan igaz, de a kettőnek köze sincs egymáshoz.</i>	+	+	-
B	Moo zii(x:Xii) függvénye nem szorozhatja össze az m és n attribútumok értékeit, mert az n attribútum privát. <i>Moo implementálja a Xii interfészt (így annak zii metódusát, amiben simán szerepelhet a 2 attr. összeszorása), így ez hamis. 2. triviálisan igaz.</i>	-	+	
E	Raa paa(p:Raa) függvénye nem hívhatja meg a fuu():Xii függvényt, mert az sértené a Pauli-elvet. <i>A protected metódust is öröklí az osztály, így meghívható, hamis. A 2. vagy nem figyeltem, vagy csak egy becsapós kérdés...</i>	-	-	
A	Wee wee(m:Moo) függvénye nem módosíthatja a q attribútumot, mert q package láthatóságú. <i>Statikus metódus akarna példány attr.-t módosítani, ezt nem szabad, igaz. Protected, nem package láthatóságú, hamis.</i>	+	-	
C	Wee wee(m:Moo) függvénye meghívhatja az m paraméter zii(x:Xii) függvényét, mert az nem sérti a Demeter-törvényt. <i>Moo-nak van zii-je, és pulikus, így ez igaz (private vagy protected-re már hamis lenne). A m.zii(...) még belefér a LoD-be („túlpontozást” nem szeretjük, a „láncolást”, vagyis a a.b.c.d.foo(), a.bar().bar1().bar2())...és társait, részletesebb ld. jegyzet/google), így ez is igaz. A kettőnek viszont semmi köze egymáshoz.</i>	+	+	-
D	Moo és Xii interfésze megegyezik, mert Moo nem definiál újabb függvényt. <i>Pontosan, mind2-nek csak zii(x:xii) metódusa van.</i>	+	+	+

2014.01.21 – 1. Feladat



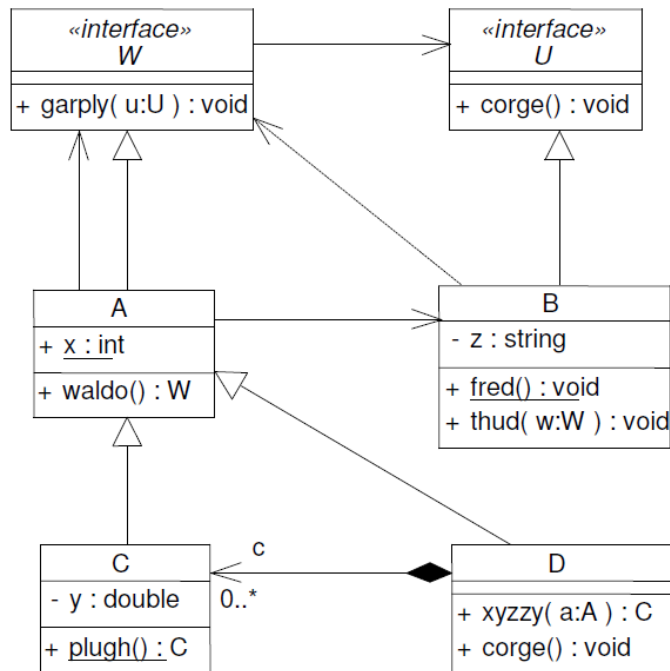
Válasz	Állítás	1.	2.	köv.
C	Zizi mimi(t:Titi) függvénye nem hívhatja meg a t paraméter nunu() függvénye által visszaadott Jojo interfészű objektum dodo() függvényét, mert az sértené a Demeter-törvényt. <i>Tehát: t-n keresztül elérjük a Titi.nunu()-t, ami adhat vissza Jojo-t, mivel az öröklí a Bubu interfészt, viszont csak az „eredeti”, Bubu interfész függvényeit hívhatjuk meg, így a dodo-t nem, így ez igaz. Demeter-törvénye alapján csak a közvetlenül elért metódusokat/attribútumokat piszkáljuk (kicsit leegyszerűsítetten megfogalmazva), ez dodo-ra nem lenne igaz, így ez is igaz. Viszont nem emiatt nem hívható meg (ld. fentebb az indokot).</i>	+	+	-
A	Titi pipi() függvénye nem szorozhatja össze a ra és re attribútum értékét, mert privát változóhoz csak privát függvény férhet hozzá. <i>Statikus metódus nem nyúlka-piszka példány attribútumhoz, így ez igaz. 2. triviálisan hamis, osztályon belül a láthatóságok nem befolyásolják az elérhetőséget (bármilyen láthatóságú metódus bármilyen láthatóságú attribútumot használhat, DE! osztályon belül!).</i>	+	-	
B	Zizi a ti.nunu() hívás eredményén hívhat roro(j:Jojo) metódust, mert Nono implementálja a Bubu interfészt. <i>Hasonló az 1.-höz: nunu Bubu-t ad vissza, és bár visszaadhat ezt megvalósító osztályt/öröklő interfészt, csak Bubu-ban levő metódusok hívhatók meg, hamis. Nono közvetlen implementálja Bubu interfészt, igaz.</i>	-	+	
B	Egy Titi objektum csak egy Jojo interfészű objektumot ismer, mert közöttük lévő asszociáció Jojo oldalán 1-es számosság szerepel. <i>A qualifier miatt hamis. 2. triviálisan igaz.</i>	-	+	
E	Nono roro(j:Jojo) függvénye kaphat paraméterül Bubu interfészű objektumot, mert Jojo implementálja a Bubu interfészt. <i>Bubu nem Jojo leszármazottja, így ez hamis. Nem implementálja, hanem öröklí, maga is egy interfész, így nem is implementálhatná.</i>	-	-	
E	Nono dodo() függvénye nem hozhat létre Zizi objektumot, mert Nono nem ismeri Zizi -t. <i>De, mivel Nono öröklí Jojo függését Zizi-től, így ez hamis. Nono ismeri Z-t, hiszen – ahogy azt már mondtam – függ Zizi-től.</i>	-	-	
B	Zizi függ Bubu -tól, mert Titi függ Bubu -tól. <i>Hamis, Zizi-nek fogalma sincs Bubu interfészről. 2. triviálisan igaz.</i>	-	+	
B	Nono dodo() függvénye nem módosíthatja a no attribútum értékét, mert Javában a String típus immutábilis. <i>Hamis, ez esetben bármilyen láthatóságú attribútumot módosíthatna dodo. A String osztálynak nincs a String objektumot módosító metódusa, így ez igaz.</i>	-	+	

2014.05.27 – 1. Feladat



Válasz	Állítás	1.	2.	köv.
B	B baz metódusa nem hívhatja meg B u attribútumának foo metódusát, mert az A osztály foo metódusa absztrakt. <i>A foo() metódus publikus, így hozzáférünk, vagyis ez hamis, utóbbi viszont igaz (dólt betűkkel van írva).</i>	-	+	
A	C corge metódusa kaphat paraméterül D típusú objektumot, ezért a metódus meghívhatja a kapott objektum garply metódusát. <i>Mivel D implicit leszármazik X-ből, így az első része igaz, viszont ettől függetlenül csak olyan metódusokat hívhat, amik X-ben szerepelnek, vagyis utóbbi hamis.</i>	+	-	
B	C waldo metódusa virtuális, ezért a B osztály baz függvénye egy paraméterül kapott D típusú objektumon meghívhatja a waldo metódust. <i>Hell no. A baz kaphat paraméterül D-t (D leszármazik C-ből), nincs semmi akadálya, hogy egy nem static függvény static metódust hívjon... szóval jha, ez igaz.</i>	-	+	
B	A baz metódusa nem módosíthatja A w attribútumát, mert A baz metódusa nem statikus. <i>Hát ez max fordítva lenne igaz (static metódus nem cseszegethet nem static attribútumot), fordítva viszont semmi akadálya, 2. meg triviálisan igaz.</i>	-	+	
A	C -nek van bar metódusa, ezért C implementálja a Z interfészt. <i>Öröklí X-ből, szóval igaz. Nem igaz („nem jutunk el öröklési nyilakon keresztül C-ből Z-be”).</i>	+	-	
B	D garply metódusa kaphat paraméterül A típusú objektumot, mert A és B interfésze megegyezik. <i>Mivel A nem származik le B-ből (amit garply vár), így ez hamis. 2 pedig igaz: leírva mind2 osztály metódusait (paramétereket is figyelembe véve) nem lesz különbség \o/.</i>	-	+	
C	D hello metódusa nem módosíthatja D v attribútumát, mert D v attribútuma privát. <i>Amit fentebb is írta: static metódus ne nyúljon nem statikus attribútumhoz (+1: és ne is használjon nem statikus metódust!). Privát, mivel kiszúrja a szemünk a '-' jel, viszont a következtetés hamis a mondat elején leírtak miatt.</i>	+	+	-
E	D garply metódusa nem módosíthatja a b paraméter u attribútumát, mert protected attribútumhoz csak privát és protected metódusok férhetnek hozzá. <i>Módosíthatja, mivel protected (privátot nem tudna!). Bullshit, aki leszármazik az osztályból, az hozzáfér.</i>	-	-	

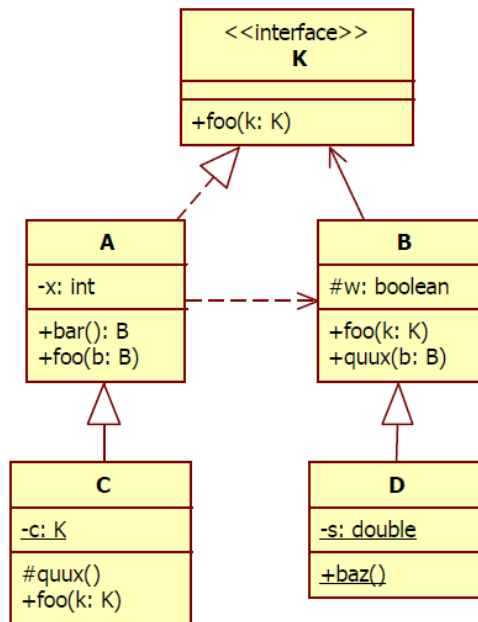
2014.06.03 – 1. Feladat



Válasz	Állítás	1.	2.	köv.
B	D törlésekor legalább egy C objektumot is törölni kell, mert D tartalmaz C-t. <i>Mivel 0..* van, így nem biztos, hogy kell törölni C-t. A másik fele triviálisan igaz.</i>	-	+	
A	U interfésze részhalmaza D interfészenek, ezért D megvalósítja az U interfészt. <i>Igen, mivel U-nak is van corge() : void metódusa, mint D-nek. Bullshit, nincs útvonal D-ből U-ba háromszögfejű-nyilak mentén.</i>	+	-	
E	C nem függ U-tól, mert C ősoosztálya (A) sem függ U-tól. <i>De, implicite függ, és de, pont ezért függ (+1: „C függ U-tól, mert C ősoosztálya (A) függ U-tól” → D lenne...szerintem).</i>	-	-	
B	A waldo függvénye nem példányosíthat B típusú objektumot, mert B nem implementálja a W interfészt. <i>Az A függ B-től, így emiatt példányosíthat B típusú objektumot (hiszen ismeri), a második meg triviálisan igaz.</i>	-	+	
E	C plugh függvénye nem módosíthatja A x attribútumát, mert A x attribútuma protected. <i>Zöld lámpát kap (mindkettő statikus, illetve láthatóság sem zavar be, az x még protected is lehetne), így ez hamis, a második meg csak akkor igaz, ha fordított nap van.</i>	-	-	
B	D xyzy függvénye visszaadhatja eredményként a paraméterként kapott a objektumot, mert C az A leszármazottja. <i>Nem, fordítva lenne igaz (ha nem C-t adunk vissza, akkor olyat kellene, ami származik C-ből). 2. triviálisan igaz.</i>	-	+	
C	B fred függvénye nem módosíthatja a z attribútum értékét, mert B z attribútuma nem protected. <i>Statikus metódus nyúlna nem statikus attribútumhoz? Hát ezt nem szabad, ez tény. Az persze igaz, hogy z az nem protected, de azért nem nyúlhat hozzá fred, mert nem statikus az attribútum.</i>	+	+	-
E	B thud függvénye meghívhatja egy paraméterül kapott C típusú objektum plugh függvényét, mert C plugh függvénye virtuális. <i>C-t kaphat paraméterül W helyett, hiszen C implementálja a W interfészt, de Csak a W interfészeben található függvényeket szabad meghívni, így ez hamis. Lofact, statikus, nem virtuális.</i>	-	-	

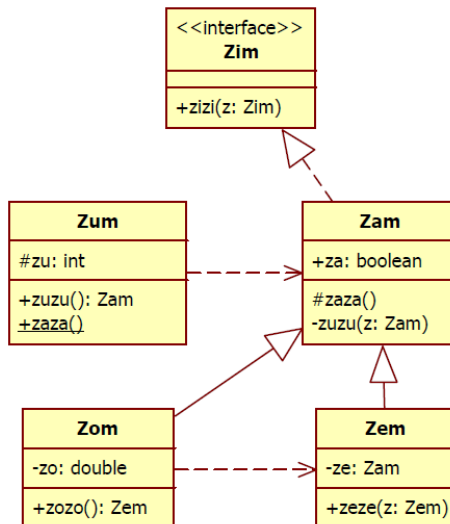
2015.01.06 – 4. Feladat

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



Válasz	Állítás	1.	2.	köv.
A	C egyetlen függvénye sem módosíthatja egy paraméterül kapott B típusú objektum w attribútumát, mert C nem függ B-től. <i>w protected, így ahhoz csak a B osztály, vagy a B leszármazottjai férnek hozzá. Közvetlenül függ (A függ, C meg öröklí A-t).</i>	+	-	
A	D baz függvénye nem módosíthatja B w attribútumát, mert w statikus. <i>Sajna a baz() static, míg a w attr. nem @. w nem statikus, csak protected.</i>	+	-	
D	Van olyan foo függvény, amely nem kaphat paraméterül B típusú objektumot, mert B nem implementálja a K interfészt. <i>A foo(k:K) nem kaphat, hiszen B nem implementálja a K interfészt. És ezzel meg is magyarázva a 2. tagmondat és a következmény (hiszen pont ezért nem kaphat).</i>	+	+	+
C	A bar függvénye nem példányosíthat D típusú objektumot, mert D nem függ A-tól. <i>Ha példányosíthatna, akkor függenie kellene tőle, ez viszont nem áll fent, így igaz. Triviálisan igaz, viszont az ellenkezője miatt igaz.</i>	+	+	-
B	C quux függvénye nem módosíthatja a c attribútum értékét, mert quux nem privát. <i>Példány metódus módosíthat static attribútumot (csak fordítva nem igaz). 2 triviálisan igaz, hiszen protected, nem private.</i>	-	+	
B	D foo függvénye nem hívhatja meg a paraméterül kapott C objektum foo(k:K) függvényét, mert D nem ismeri C-t. <i>A foo függvény K-t vár (vagy ezt implementáló osztályt), így kaphat paraméterül C-t, és mivel K-ban van foo(k:K) metódus, így ez hamis. Se közvetlenül, se közvetetten nem függ, és nincs köztük semmilyen kapcsolat se, így ez igaz.</i>	-	+	
D	Minden D-ben deklarált függvény módosíthatja az s attribútumot, mert s statikus. <i>Static saticot, ez persze, hogy igaz, 2. triviálisan igaz, és a következtetés is igaz (ha s nem lenne statikus, vagyis példány attr. lenne, akkor baz() nem férne hozzá, hiszen ő meg static).</i>	+	+	+
C	C és D interfésze különbözik, mert D nem implementálja a K interfészt. <i>Igaz, hiszen egyiknek quux(), míg a másiknak quux(b:B) metódusa van. 2. triviálisan igaz, viszont a következtetés bullshit, hamis.</i>	+	+	-

2015.01.13 – 4. Ffeladat



Válasz	Állítás	1.	2.	köv.
D	Van olyan zaza függvény, amely nem módosíthatja a za attribútumot, mert za nem statikus. <i>za az public, így max egy static nem tudná, Zum-ban pedig meg is bújuk egy. 2. triviálisan igaz, és a következtetés is adja magát (hiszen pont ez hibádzik, hogy za static legyen).</i>	+	+	+
C	Zum zuzu függvénye nem példányosíthat Zem objektumot, mert Zem nem függ Zum -tól. <i>Zum nem függ Zem-től (nem ismeri), így ez igaz. 2. igaz, hiszen se explicit, se implicit nem függ tőle, de a következtetés hamis, hiszen fordítva (Zum Zem-től) lenne igaz, vagyis D.</i>	+	+	-
A	Zom zozo függvénye példányosíthat Zom objektumot, mert Zom a Zem leszármazottja. <i>Egy osztály saját magát mindig példányosíthatja, így ez igaz. 2. meg triviálisan hamis (Zom csak függ Zem-től).</i>	+	-	
A	Zum zaza függvénye nem módosíthatja a zu attribútum értékét, mert zu privát. <i>Static metódus nem nyúlhat nem static attribútumhoz, így ez igaz. Nope, zu az protected.</i>	+	-	
E	Zom zizi függvénye nem hívhatja meg Zem zizi függvényét, mert egyiknek sincs zizi függvénye. <i>Itt jobb híján csak arra kell figyelni, hogy Zom az függ Zem-től, így meghívhatja (illetve persze mert public a metódus). Mind2 Zim-től kapja a zizi metódust.</i>	-	-	
C	Zem zeze függvénye nem kaphat paraméterül Zom objektumot, mert Zom függ Zem -től. <i>Zem helyére csak Zem, vagy Zem leszármazott kerülhetne, Zom meg nem az, így ez igaz, 2. triviálisan igaz, következtetés hamis, lásd a mondat eleje.</i>	+	+	-
A	Zam zuzu függvénye meghívhatja a paraméterül kapott Zom típusú objektum zaza függvényét, mert zaza absztrakt. <i>zuzu kaphat Zom-t (Zom egy Zam leszármazott), és mivel Zom Zam-tól kapja (az „eredeti” paramétertől) a zaza függvényt, így meghívhatja (pl. a zozo-t már nem hívhatná meg!). Hamis, abstract akkor lenne, ha dőltenne írva.</i>	+	-	
D	Zam zaza függvénye nem hívhat meg minden zuzu függvényt, mert Zam nem ismeri Zum -ot. <i>A mondat meg is magyarázta magát, *flies away*.</i>	+	+	+

asd

asd