

## Quiz navigation

1	2	3	4	5	6	7	8
✓	✓	✓	✓	✓	✓	✓	✓
9	10	11	12	13	14	15	16
✓	✓	✓	✓	✓	✓	✓	✓
17	18	19	20	21	22	23	24
●	✓	✓	✓	✓	✓	✓	✓
25	26	27	28	29	30	31	32
✓	✓	✓	✓	✓	✓	✓	✓
33	34	35	36	37			
✓	✓	✓	✓	✓			

Show one page at a time

[Finish review](#)

**Started on** Thursday, 14 January 2021, 10:04 AM

**State** Finished

**Completed on** Thursday, 14 January 2021, 10:47 AM

**Time taken** 43 mins 19 secs

**Grade** 35.33 out of 50.00 (71%)

**Feedback** Jó (4)

### Question 1

Correct

Mark 1.00 out of 1.00

[Flag question](#)

Igaz vagy hamis a következő állítás?

**"REST szolgáltatók esetén a HTTP body-ban vándorló üzenetek formátuma minden JSON vagy XML."**

Select one:

True

False ✓

The correct answer is 'False'.

### Question 2

Incorrect

Mark 0.00 out of 1.00

[Flag question](#)

Igaz vagy hamis a következő állítás?

**A Proactor minta lényege, hogy aszinkron módon reagálunk a kívülről érkező eseményekre.**

Select one:

True ✗

False

The correct answer is 'False'.

### Question 3

Correct

Mark 1.00 out of 1.00

[Flag question](#)

Az oktató emailben kérdést intéz a tárgy hallgatóihoz, és megkéri a hallgatókat, hogy válasz emailben tüntessék fel a Neptun-kódjukat. Az alábbiak közül melyik konkurens-elosztott minta szerepére tölti be a Neptun-kód?

Scheduler

Auto reset event

Manual reset event

Guarded suspension

Proactor

Cancellation token

Asynchronous completion token

Reactor

Válasza helyes.

The correct answer is:

Asynchronous completion token

### Question 4

Incorrect

Mark 0.00 out of 1.00

[Flag question](#)

Egy telefonos alkalmazás egy háttérzalon kérést intéz a szerver felé, majd az eredményt a megjeleníti a felhasználói felületen. Az alábbiak közül melyik konkurens-elosztott minta valósítja ezt meg az alkalmazásban?

Reactor

Auto reset event

Proactor

Scheduler

Manual reset event

Válasza helytelen.

The correct answer is:

Proactor

### Question 5

Correct

Mark 1.00 out of 1.00

[Flag question](#)

Telefonunkon egy társkereső alkalmazás több háttérzalon keresi a számunkra megfelelő jelöltek profiliat. A felhasználói interfészben egyszerre csak egy találat tud megjelenni. Amikor a jelolt profiliát jobbra vagy balra elhúzzuk, a következő megtalált jelölt profilja jelenik meg. Az alábbiak közül melyik konkurens-elosztott minta segít a háttérzálak és a felhasználói interfész közötti szinkronizálásban?

Auto reset event

Manual reset event

Double-checked locking

Readers-writer lock

Global context

Válasza helyes.

The correct answer is:

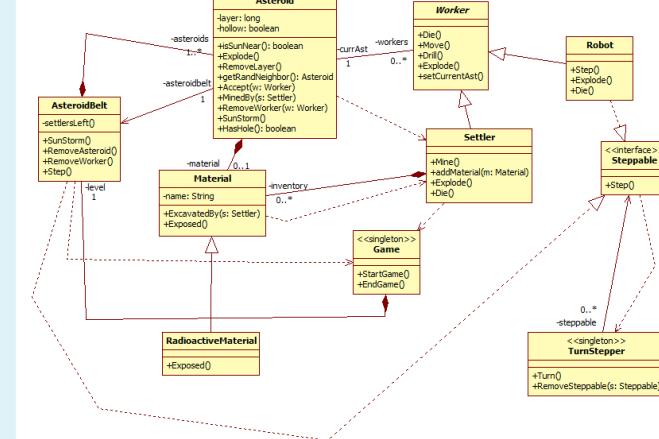
Auto reset event

### Question 6

Correct

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):

Mark 2.00 out of 2.00  
Flag question

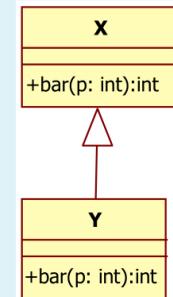


Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- Közös interfész csak akkor valósultunk meg, ha a viselkedés is közös!
- Az öröklődés célja minden a viselkedés újrahasznosítása!
- Az opcionális elemeket tartalmazásként implementáljuk, ne öröklődéssel!
- Egyik felkinált heurisztika sem sérül
- Egy osztály ne függjen a saját leszármazottaitól!
- Absztrakt osztályok az öröklési hierarchia gyökerében legyenek!

Question 7  
Correct  
Mark 1.00 out of 1.00  
Flag question

Teljesül-e a Liskov-elv, ha X bar() függvénye 0..100 közötti értékeket ad vissza, míg Y bar() függvénye 1..100 közötti értékeket ad vissza?  
(Vegyük úgy, hogy a feltétlenben nem szereplő elemek teljesítik a Liskov-elvet)

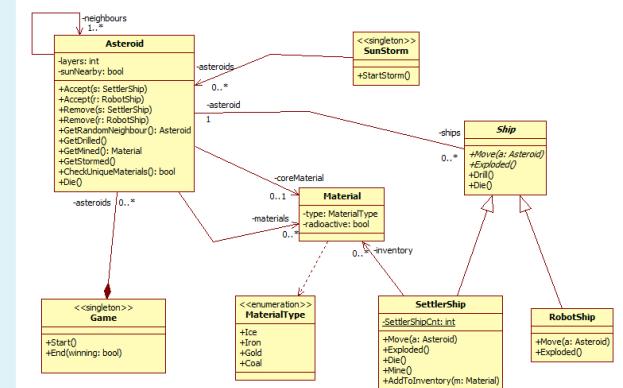


- Select one:
- Teljesül
  - Sérül
  - Nem jelölök egyiket sem

The correct answer is: Teljesül

Question 8  
Partially correct  
Mark 1.00 out of 2.00  
Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):



Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- Ne keverjük össze a statikus és a dinamikus kényeszereket!
- Egy osztály ne függjen az őt használó osztályoktól!
- Modellezzünk a maradék abstraktív részleteket!

- Egyik felkinált heurisztika sem sérül
- Soha ne kódoljuk a típust (pl. enum vagy int értékekbe), használunk helyette polimorfizmust!
- Modellezésnél maradjunk a rendszer határain belül!

✓

Válasza részben helyes.

You have correctly selected 1.

The correct answers are: Soha ne kódoljuk a típust (pl. enum vagy int értékekbe), használunk helyette polimorfizmust!

Egy osztály ne függjen az ot használó osztályoktól!

Question 9  
Correct  
Mark 1.00 out of 1.00  
[Flag question](#)

Igaz vagy hamis a következő állítás?

**"Publish-subscribe típusú üzenetkezelés során minden feliratkozó minden üzenetet megkap."**

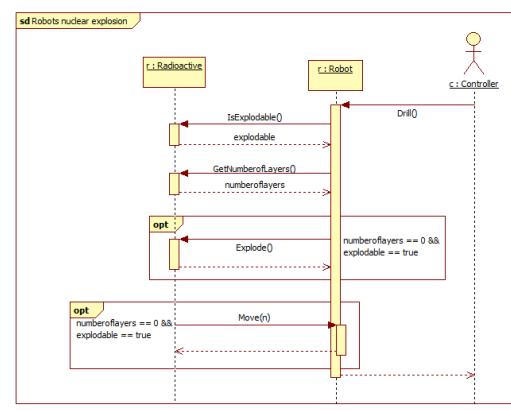
Select one:

- True ✓
- False

The correct answer is 'True'.

Question 10  
Correct  
Mark 1.00 out of 1.00  
[Flag question](#)

Tekintsük az alábbi szekvencia diagramot (követelmények mutatása/elrejtése):



Igaz vagy hamis a következő állítás a szekvencia diagram alapján?

**"Sérül a TDA elv."**

Select one:

- True ✓
- False

The correct answer is 'True'.

Question 11  
Correct  
Mark 1.00 out of 1.00  
[Flag question](#)

Igaz vagy hamis a következő állítás?

**"API készítése során a kísérleti funkciókat is érdemes publikálni, hogy a könyvtárunk felhasználói mielőbb hasznát vehessék azoknak."**

Select one:

- True
- False ✓

The correct answer is 'False'.

Question 12  
Correct  
Mark 1.00 out of 1.00  
[Flag question](#)

Tekintsük az alábbi kód részletét:

```

public class Person {
    /**
     * <summary>
     * </summary>
     * protected Person()
     * {
     * }
     *
     * <summary>
     * </summary>
     * private int age;
     */
    /**
     * <summary>
     * </summary>
     * Returns the age of the person.
     * <returns>the age of the person</returns>
     */
    public int Age
    {
        get { return age; }
    }
}
  
```

Igaz-e a következő állítás?

**A Clean Code elvek szerint helyesen használtuk a kommenteket.**

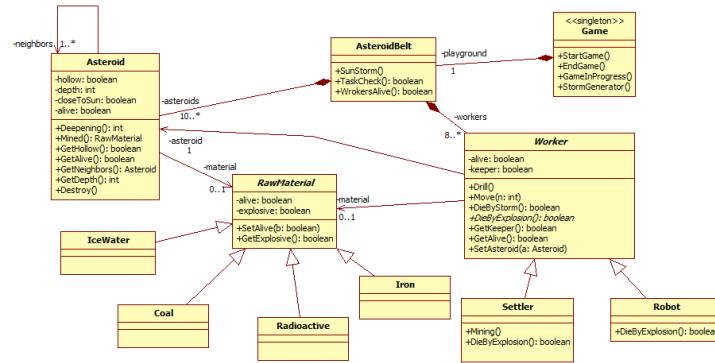
Select one:

- True
- False ✓

The correct answer is 'False'.

Question 13  
Partially correct  
Mark 1.33 out of 2.00  
 Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):



Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- A tartalmazó objektum használja a tartalmazott objektumokat!
- Az öröklési hierarchia gyökerében interfések vagy absztrakt osztályok legyenek!
- Ne készítsünk függvényeket a típusok illetve a képességek megkülönböztetésére, használunk helyettük polimorfizmust!
- Absztrakt osztályok az öröklési hierarchia gyökerében legyenek!
- Egyik felkinált heurisztika sem sérül
- Ne keverjük össze a leszármazottakat az objektumokkal!



Válasza részben helyes.

You have selected too many options.

The correct answer is: Ne készítsünk függvényeket a típusok illetve a képességek megkülönböztetésére, használunk helyettük polimorfizmust!

Question 14  
Correct  
Mark 1.00 out of 1.00  
 Flag question

Igaz vagy hamis a következő állítás?

**A C# nyelvben minden objektum egyben Monitor object is.**

Select one:

- True
- False

The correct answer is 'False'.

Question 15  
Incorrect  
Mark 0.00 out of 1.00  
 Flag question

Igaz vagy hamis a következő állítás?

**"API készítése során a példák írása megelőzi az API implementálását."**

Select one:

- True
- False

The correct answer is 'True'.

Question 16  
Partially correct  
Mark 3.00 out of 5.00  
 Flag question

Párositsa az alábbi OO elveket és tervezési heurisztikákat a nekik leginkább megfelelő refaktorálási mintákkal és code smellekkell!

Ha az ős működését üres implementációval írjuk felül, akkor hibás az öröklési hierarchia!

DRY

Demeter-törvény

SDP

Minimalizáljuk az együttműködő osztályok között használt metódusok számát!

Refused bequest	
Shotgun surgery	
Add parameter	
Substitute algorithm	
Move method	

Válasza részben helyes.

You have correctly selected 3.

The correct answer is:

Ha az ős működését üres implementációval írjuk felül, akkor hibás az öröklési hierarchia! → Refused bequest,

DRY → Shotgun surgery,

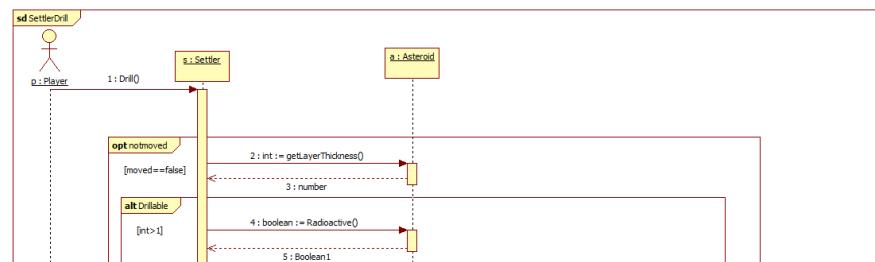
Demeter-törvény → Hide delegate,

SDP → Separate domain from presentation,

Minimalizáljuk az együttműködő osztályok között használt metódusok számát! → Move method

Question 17  
Partially correct  
Mark 1.00 out of 2.00  
 Flag question

Tekintsük az alábbi szekvencia diagrammot (követelmények mutatása/elrejtése):





Az alábbiak közül mely OO tervezési elvek és heurisztikák sérülnek a szekvencia diagram alapján?

- Kerüljük az isten-osztályokat!
- TDA
- Modellezük a valódi világ működését!
- Ne készítünk függvényeket a típusok illetve a képességek megkülönböztetésére, használunk helyettük polimorfizmust!
- A tartalmazó objektum használja a tartalmazott objektumokat!
- Demeter-törvény
- Egyik felkinált elv vagy heurisztika sem sérül
- Az összetartozó adatot és viselkedést tartsuk egy helyen!
- A felfelességeket egyenletesen osszuk szét!
- SRP
- Minimalizáljuk az együttműködő osztályok között használt metódusok számát!
- 

Válasza részben helyes.

You have correctly selected 4.

The correct answers are:

Kerüljük az isten-osztályokat!

A felelősségeket egyenletesen osszuk szét!,

Modellezük a valódi világ működését!,

Ne készítünk függvényeket a típusok illetve a képességek megkülönböztetésére, használunk helyettük polimorfizmust!,

Minimalizáljuk az együttműködő osztályok között használt metódusok számát!,

TDAs,

SRP,

Az összetartozó adatot és viselkedést tartsuk egy helyen!

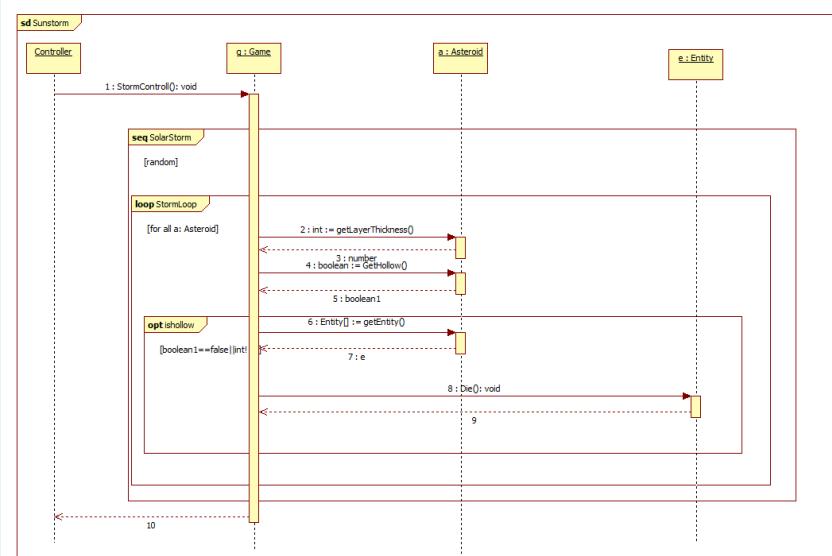
#### Question 18

Correct

Mark 1.00 out of 1.00

Flag question

Tekintsük az alábbi szekvencia diagramot (követelmények mutatása/elrejtése):



Igaz vagy hamis a következő állítás a szekvencia diagram alapján?

"A felelősségek rosszul vannak kiosztva."

Select one:

True ✓

False

The correct answer is 'True'.

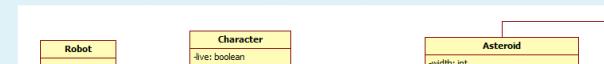
#### Question 19

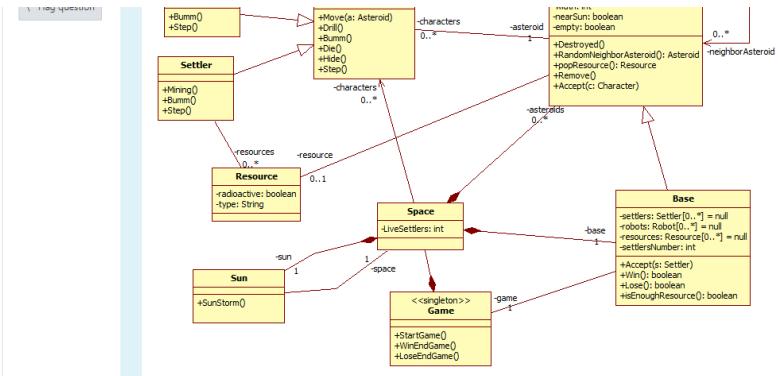
Correct

Mark 2.00 out of 2.00

Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):





Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- Kerüljük a csak adattárolásra használt osztályokat!
- Egyik felkínált heurisztika sem sérül
- Soha ne kódoljuk a típust (pl. enum vagy int értékekbe), használunk helyette polimorfizmust!
- A közös viselkedés és közös adat minél magasabban legyen az öröklési hierarchiában!
- Ha többszörös öröklésre van szükségünk, gondoljuk át még egyszer a terveket!
- Protected láthatóságot csak metódusoknál használunk, az attribútumok minden privátok legyenek!

Válasza helyes.

The correct answers are: Kerüljük a csak adattárolásra használt osztályokat!, Soha ne kódoljuk a típust (pl. enum vagy int értékekbe), használunk helyette polimorfizmust!

#### Question 20

Correct

Mark 1.00 out of 1.00

Flag question

Telefonkon az egyik alkalmazás egy háttérálon tölti le a számára szükséges adatokat. A művelet tul sokáig tart, elununk magunkat, és a felhasználói felületen egy gomb megnyomásával leíllítjük azt. Az alábbiak közül melyik konkurens-elosztott minta segít ebben?

- Cancellation token
- Guarded suspension
- Double-checked locking
- Manual reset event
- Global context

Válasza helyes.

The correct answer is:  
Cancellation token

#### Question 21

Correct

Mark 1.00 out of 1.00

Flag question

Igaz vagy hamis a következő állítás?

"SOAP webszolgáltatások esetén az interfészleíró XML szintaktikát követ."

Select one:

- True ✓
- False

The correct answer is 'True'.

#### Question 22

Partially correct

Mark 0.67 out of 2.00

Flag question

Válassza ki az alábbiak közül az immutable objektumok használatának hátrányait!

- Olcsóbb egy létező objektum módosítása, mint egy új létrehozása
- Nem szálbiztosak
- Visszatéréskor védelmi másolatot kell készíteni
- Implementálásuk imperativ nyelvekben általában sok programkódot igényel
- Kényelmetlen szintaxis
- Változhat az identitásuk

Válasza részben helyes.

You have correctly selected 2.

The correct answers are:

Kényelmetlen szintaxis,

Olcsóbb egy létező objektum módosítása, mint egy új létrehozása,

Implementálásuk imperativ nyelvekben általában sok programkódot igényel

#### Question 23

Incorrect

Mark 0.00 out of 1.00

Flag question

Igaz vagy hamis a következő állítás?

**A Strategized locking minta segít elkerülni a rekurzív hívásokból adódó deadlock helyzeteket.**

Select one:

- True ✗
- False

The correct answer is 'False'.

Question 24  
Correct  
Mark 1.00 out of 1.00  
 Flag question

Tekintsük az alábbi szekvencia diagrammot (követelmények mutatása/elrejtése):

The sequence diagram illustrates the interaction between several objects:

- Participants:** s : Sun, ab : AsteroidBelt, a : Asteroid, r : Robot, ai : AI.
- Initial Message:** The sequence begins with a synchronous message `SunStormHits()` sent from the `s : Sun` object to the `loop` object.
- loop Object:** The `loop` object contains a condition `[for all a:Asteroid]`. It receives a synchronous message `HitBySunStorm()` from the `a : Asteroid` object.
- alt Object:** The `loop` object branches into an `alt` block. The first alternative `[layer == 0 && material[ai] == null]` leads to a synchronous message `BreakDown()` sent to the `r : Robot` object.
- else Alternative:** The second alternative `[else]` leads to a synchronous message `RemoveRobot(r)` sent to the `loop` object.
- Return Path:** The `loop` object returns to the `s : Sun` object via a dashed arrow.

**Question 25**  
Correct  
Mark 1.00 out of 1.00  
[Flag question](#)

Tekintsük az alábbi kód részletet:

```
try
{
    logger.Log(message);
}
catch (Exception ex)
{
    // Intentionally empty
}
```

Igaz-e a következő állítás?

**A Clean Code elvek szerint helyesen használtuk a kommenteket.**

Select one:

True ✓

False

The correct answer is 'True'.

**Question 26**  
Correct  
Mark 2.00 out of 2.00  
 Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):

```

classDiagram
    class Driller {
        +CollideWith(t: Thing)
        +Move()
        +Drill()
        +Die()
        +Hit()
        +HitBy(ss: Sunstorm)
        +HitBySun()
    }
    class Thing {
        +CollideWith(t: Thing)
        -things 0..*
    }
    class Asteroid {
        -asteroids 0..*
        -neighbours 0..1
        1
        +rocks int
        +Accept(t: Thing)
        +Remove(t: Thing)
        +HandleSettlersAlive(): boolean
        +DecreaseRocks()
    }
    class AsteroidBelt {
        +CreateSunstorm()
        +RemoveAsteroid(a: Asteroid)
    }
    class Settler {
        +Mine()
        +HitBySun()
    }
    class Robot {
        +HitBySun()
    }
    class Sunstorm {
        +CollideWith(t: Thing)
        +Storm()
        +KillSettler()
    }
    class RadioactiveAsteroid {
        +Explode()
    }
    class Game {
        -settlers: int
        +Win()
        +Lose()
        +DecreaseSettlers()
    }
    class RawMaterial {
        -rawMaterials 0..1
        0..1
        +close
        +far
    }
    class Sun {
        +close
        +far
    }
    class WaterIce {
        +HitBy(s: Settler)
    }
    class Iron {
        +HitBy(s: Settler)
    }
    class Carbon {
        +HitBy(s: Settler)
    }

    Driller <|-- Thing
    Asteroid <|-- Thing
    AsteroidBelt <|-- Thing
    Settler <|-- Thing
    Robot <|-- Thing
    Sunstorm <|-- Thing
    RadioactiveAsteroid <|-- Thing
    Game <|-- Thing

    Asteroid "1" *-- "0..1" RawMaterial
    Asteroid "1" *-- "0..1" Sun
    Asteroid "1" *-- "0..1" AsteroidBelt
    Asteroid "1" *-- "0..1" Settler
    Asteroid "1" *-- "0..1" Robot
    Asteroid "1" *-- "0..1" Sunstorm
    Asteroid "1" *-- "0..1" RadioactiveAsteroid

    Asteroid "1" *-- "0..1" WaterIce
    Asteroid "1" *-- "0..1" Iron
    Asteroid "1" *-- "0..1" Carbon

    RawMaterial "0..1" *-- "0..1" WaterIce
    RawMaterial "0..1" *-- "0..1" Iron
    RawMaterial "0..1" *-- "0..1" Carbon
  
```

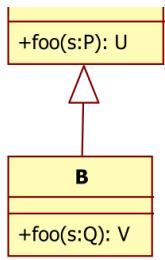
Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- A közös viselkedéssel nem rendelkező közös adat tartalmazás relációban legyen!
- A közös viselkedéssel rendelkező közös adat az osztályban legyen!
- Az öröklési hierarchia gyökerében interfések vagy absztrakt osztályok legyenek!
- A viselkedést modellezük, ne a szerepeket!
- A közös viselkedés és közös adat minél magasabban legyen az öröklési hierarchiában!
- Egyik felkínált heurisztika sem sérti

Válasza helyes

The correct answers are: A viselkedést modellezük, ne a szerepeket!, Az öröklési hierarchia gyökerében interfések vagy absztrakt osztályok legyenek!

Question 27	Teljesül-e a Liskov-elv, ha P megegyezik Q-val, A foo() függvénye null értékeket is elfogad és B foo() függvénye nem fogad el null értékeket. (Végyük úgy, hogy a feltételben nem szereplő elemek teljesítik a Liskov-elvet!)
Correct	
Mark 1.00 out of 1.00	



Select one:

- Teljesül
- Sérül
- Nem jelölöm egyiket sem



The correct answer is: Sérül

Question 28

Correct

Mark 1.00 out of 1.00

Flag question

Igaz vagy hamis a következő állítás?

"API készítése során az elnevezéseknek a betűszavak rendben vannak, de a rövidítések érdemes kerülni."

Select one:

- True ✓
- False

The correct answer is 'True'.

Question 29

Correct

Mark 1.00 out of 1.00

Flag question

Igaz vagy hamis a következő állítás?

A Leader-Followers minta segítségével rugalmasabban oszthatók ki a feladatok, mint a Scheduler segítségével.

Select one:

- True
- False ✓

The correct answer is 'False'.

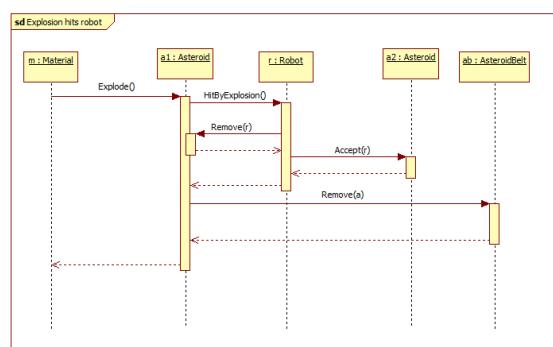
Question 30

Correct

Mark 1.00 out of 1.00

Flag question

Tekintsük az alábbi szekvencia diagrammot (követelmények mutatása/elrejtése):



Igaz vagy hamis a következő állítás a szekvencia diagram alapján?

"Van olyan hívás, amely hibás, mert a hívó nem ismeri a hívott objektumot."

Select one:

- True ✓
- False

The correct answer is 'True'.

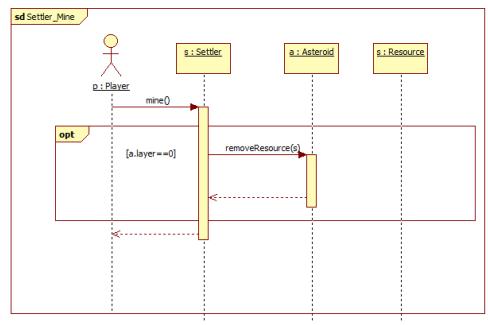
Question 31

Incorrect

Mark 0.00 out of 1.00

Flag question

Tekintsük az alábbi szekvencia diagrammot (követelmények mutatása/elrejtése):



Igaz vagy hamis a következő állítás a szekvencia diagram alapján?

"Sérül a TDA elv."

Select one:

- True  
 False 

The correct answer is 'True'.

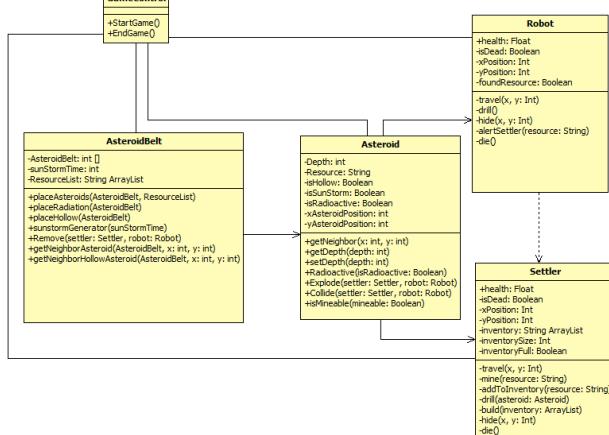
Question 32

Partially correct

Mark 0.33 out of 2.00

 Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):



Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- A tartalmazott objektum ne használja a tartalmazó objektumot!
- Az attribútumok minden legyenek privátok!
- Egyik felkinált heurisztika sem sérül
- A közös viselkedéssel rendelkező közös adat az óosztályban legyen!
- Ne használjuk másik osztály nempublikus tagjait!
- Közös interfész csak akkor valósítsunk meg, ha a viselkedés is közös!

Question 33

Incorrect

Mark 0.00 out of

1.00

 Flag question

Válasza részben helyes.

You have correctly selected 2.

The correct answers are:

A közös viselkedéssel rendelkező közös adat az óosztályban legyen,

Ne használjuk másik osztály nempublikus tagjait!

Az attribútumok minden legyenek privátok!

The correct answer is 'True'.

Question 34

Correct

Mark 1.00 out of

1.00

 Flag question

Igaz vagy hamis a következő állítás?

**A Guarded suspension mintában nem szabad az "if" kulcsszót használni a "while" kulcsszó helyett.**

Select one:

- True  
 False 

The correct answer is 'True'.

Készítettünk egy webalkalmazást, amely a látogató számára kijelzi, hogy mi a látogató IP címe. Az alkalmazást egy olyan webszerverre telepítettük, amely egyszerre több szalonban képes kienseket kiszolgálni. Az alábbiak közül melyik konkurens-elosztott minta segíthet abban, hogy az alkalmazásunkban webszervertől elkerülhetők a kérést indító kiens IP címét?

- Thread local context
- Manual reset event
- Double-checked locking
- Auto reset event
- Readers-writer lock

Válasza helyes.

The correct answer is:

Thread local context

Question 35

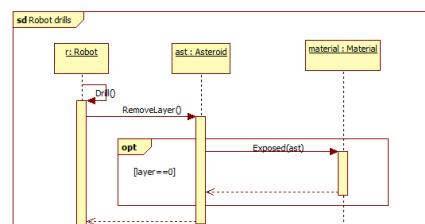
Incorrect

Mark 0.00 out of

1.00

 Flag question

Tekintsük az alábbi szekvencia diagrammot (követelmények mutatása/elrejtése):



Igaz vagy hamis a következő állítás a szekvencia diagram alapján?

"Sérül a TDA elv."

Select one:

True

False

The correct answer is 'False'.

### Question 36

Correct

Mark 1.00 out of 1.00

Flag question

Tekintsük az alábbi kód részletét:

```
try
{
    logger.Log(message);
}
catch (Exception ex)
{
    // Oh, come on!!!
}
```

Igaz-e a következő állítás?

A Clean Code elvek szerint helyesen használtuk a kommenteket.

Select one:

True

False ✓

The correct answer is 'False'.

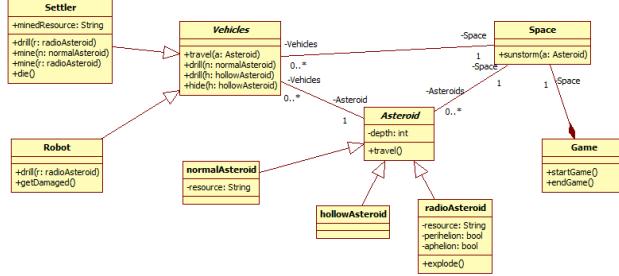
### Question 37

Correct

Mark 2.00 out of 2.00

Flag question

Tekintsük az alábbi osztálydiagramot (követelmények mutatása/elrejtése):



Az alábbiak közül mely OO tervezési heurisztikákat sérti az osztálydiagram?

- Egyik felkinált heurisztika sem sérül
- Az attribútumok minden legyenek privátok!
- Ne keverjük össze a statikus és a dinamikus kényszerkezetet!
- Egy osztály ne függjen a saját leszármazottaitól!
- Implementáljuk a sztenderd metódusokat!
- Modellezésnél maradjunk a rendszer határain belül!

Válasza helyes.

The correct answers are:

Ne keverjük össze a statikus és a dinamikus kényszerkezetet,

Implementáljuk a sztenderd metódusokat!,

Az attribútumok minden legyenek privátok!

Finish review