1. A tesztelés a minőségellenőrzés egyik formája.
2. Mindenre kiterjedő tesztelés a triviális eseteket leszámítva nem lehetséges.
3. A tesztelésnek egyetlen univerzálisan alkalmazható megközelítése van.
4. Emberi hibák (error) mindig vezetnek meghibásodásokhoz (failure).
5. Az egyes számítógépek együttműködését a különböző hálózatok biztosítják.
6. A hálózat viselkedése nem változik a terhelés függvényében.
7. A konfiguráció segítségével jelentősen befolyásolhatjuk a rendszerek működését.
8. Ha a környezet nem olyan, mint amire a rendszert tervezték, nem garantálható a rendszer megfelelő működése.
9. A rendszer input és output adatai zártak, tehát a rendszer nem "feltételezi", hogy ezek megváltozhatnak.
10. Az emberi tévedés az egyik legkomolyabb hiba ok.
11. A folyamatos technológiai fejlődések, nincsenek hatással az ember tudására, ezáltal elég az eddig megszerzett tapasztalat.
12. A tesztelés feladata csak az, hogy a használat során fellépő problémákat csökkentse.
13. A használhatóság célja, hogy a felhasználói élmény minél jobb legyen.
14. A karbantarthatóság szempontjából, olyan megoldásokat kell keresni, hogy a rendszer átlátható legyen.
15. A tesztelést a lehető legkorábban el kell kezdeni.
16. Ha ugyan azokat a teszteket hajtjuk végre, egy idő után nem fogunk hibát kapni.
17. A hibák gyakorisága attól függ, hogy a hibás funkció milyen gyakran van használatban.
18. Ismeret hiány || időnyomás || nem megfelelő szervezés == időveszteség
19. A tesztelés teljes mértékben a teszt tárgyának verifikációjára terjed ki.
20. A tesztelés dinamikus && statikus.

1. A tesztelés céljai közé nem tartozik a követelmények, felhasználói történetek, műszaki tervek és a kód kiértékelése

H

2. A tesztelés és a hibakeresés különálló tevékenységek

I

3. A tesztelés a hibák jelenlétét mutatja, nem a hiányukat

I

4. A korai teszteléssel nem spórolsz időt és pénzt

H

5. A szoftver meghibásodások oka származhat a szoftver működési környezetéből és magából a szoftverből.

I

6. A rendszerek tervezésénél nem játszik fontos szerepet az újrafelhasználhatóság.

H

7. Ha az adatok struktúrája sérül meg, akkor az nem ismerhető fel , és erre nem lehet megfelelő hibakezelési eljárást építeni.

H

8. Sok rendszer akár több száz paraméter segítségével konfigurálható és azok értéktartománya is több tíz elemből állhat

I

9. Azoknál a rendszereknél ahol a működésnek szerves része az emberi beavatkozás, ott természetesen az emberi hibát is számításba kell venni.

I

10. A rendszer működése során eltárol bizonyos adatokat a memóriában vagy az adatbázisban

I

11. A szoftverek meghibásodása nem származhat a külső körülményeken túl olyan okokból, amely az elkészítéskor elkövetett hibákra vezethetők vissza

H

12. A minőségbiztosítás önmagában nem egy rendszer.

H

13. A tesztelés kimutathatja a hibák jelenlétét, de azt nem képes igazolni, hogy nincsenek hibák

I

14. A hibák megtalálása és javítása hasztalan, ha a kifejlesztett rendszer használhatatlan, és nem felel meg a felhasználók igényeinek, elvárásainak

I

15. Ha mindig ugyanazokat a teszteket hajtjuk végre, akkor az azonos a tesztkészlet mindig fog találni új hibákat

I

16. A tesztelést a szoftver vagy rendszerfejlesztési életciklusban nem muszáj az elején elkezdeni

H

17. A tesztelés céljai a környezettől függően változhatnak

I

18. Emberi hibákat nem csak emberi lények követhetnek el

H

19. A tesztelemzés során a funkcionalitást elemezzük

H

20. A tesztelést elszigetelten végzik

H

1, A tesztelés a minőségellenőrzés  egyik formája.

Igaz

2, A tesztelésnek ( van) egyetlen univerzálisan alkalmazható megközelítése van

Hamis

3, A tesztelés feladata az, hogy a szoftver használata során fellépő hibák előfordulását csökkentse, a szoftver megbízhatóságát növelje és a szabványoknak, előírásoknak való megfelelőségét biztosítsa. Tehát az ügyfél elégedettségét növelje!

Igaz

4, A minőség biztosítás egyik komponense a hibaelemzés

Igaz

5, Karbantarthatóság szempontjából, nem feltétlen kell olyan megoldásokat  választani, ami azt biztosítja, hogy a rendszer üzemeltetése során, könnyen áttekinthető legyen.

Hamis

6, A minőség definíciója: Az a szint, amikor egy komponens, rendszer vagy folyamat megfelel a meghatározott követelményeknek és/vagy a felhasználó/ügyfél elvárásainak.

igaz

7, A karbantarthatóság nem folyásolja be a minőséget

Hamis

8, Az egyes számítógépek együttműködését a különböző hálózatok biztosítják.

Igaz

9, A rendszerek tervezésénél fontos szempont az újrafelhasználhatóság.

Igaz

10,  Hordozhatóság szempontjából fontos, hogy a rendszer úgy épüljön fel, hogy az a környezet  változásaira az előzetesen tervezettnek megfelelően tudjon reagálni.

Igaz

11,  A hatékonyság esetén azt vizsgáljuk, hogy az adott szoftver hogyan gazdálkodik az erőforrásokkal.

Igaz

12, Megbízhatóság esetén a rendszer funkcióinak nem kell folyamatos működés.

Hamis

13, A tesztelés és a hibakeresés nem különálló tevékenységek

Hamis

14, A tesztelés függ a környezettől, de magas szinten léteznek gyakori teszttevékenységek, amik nélkül kevésbé valószínű, hogy a tesztelés eléri a tesztcélokat.

Igaz

15, A tesztmegvalósítás során létrehozzuk vagy beszerezzük a tesztvégrehajtáshoz szükséges tesztvert.

Igaz

16, A tesztvégrehajtás során a teszteket a tesztvégrehajtási ütemtervben nem meghatározott sorrendben hajtjuk végre.

Hamis

17, A hatás (impact) számítása folyamatonként elvégezhető vagy tapasztalati úton becsülhető.

Igaz

18, Usability célja, hogy a felhasználói élmény (User eXperience) minél jobb legyen.

Igaz

19 , Funkcionális hibák esetén nem határozzuk meg azt a mennyiséget, típust és gyakoriságot, amely mellett még elfogadható minőségről beszélünk.

Hamis

20, A kockázat (risk) az a tényező, amely a jövőben negatív következményeket okozhat.

Igaz

1. Igaz: A vészhelyzeteket szimulációs szoftverekben próbálják ki, amelyek játékokból fejlődtek valódi kiképzést segítő eszközökké.
2. Igaz: Ha egy szoftver hibásan működik, az első játékosok vagy újságírók által írt tesztek akár egy játék kereskedelmi sikerét is eldönthetik.
3. Igaz: A szoftverek meghibásodása a szoftver működési környezetéből és magából a szoftverből is származhat.
4. Igaz: A tesztelést a lehető legkorábbi szakaszban kell kezdeni a fejlesztési folyamat során.
5. Igaz: A "féregirtó paradoxon" szerint, ha ugyanazokat a teszteket ismételjük, egy idő után nem találunk új hibákat.
6. Igaz: A tesztelés alapvető célja a hibák észlelése és azok kijavításának elősegítése.

7. Igaz: Az ISO 2014-es szabvány szerepel a dokumentumban, mint hivatkozás a szoftvertesztelési folyamatokra.
8. Igaz: Az issue tracking rendszerek szerepelnek a szövegben, mint eszközök a tesztelési folyamatok követésére.
9. Igaz: A dokumentum említi a felhasználói élményt és annak fontosságát a használhatósági tesztek során.
10. Igaz: A kockázatkezelés és annak hatásának figyelembevétele is megjelenik a szövegben.
11. Hamis: A vészhelyzeteket szimulációs szoftverekben próbálják ki, amelyek filmekből fejlődtek ki.
12. Hamis: A szoftver hibái általában nincsenek hatással egy játék kereskedelmi sikerére.
13. Hamis: A tesztelés csak a kritikus modulok hibáit találhatja meg, a kisebb modulok hibái nem számítanak.
14. Hamis: A tesztelésnek mindig ugyanúgy kell zajlania, függetlenül a rendszer típusától és céljától.
15. Hamis: Kimerítő tesztelés minden esetben lehetséges, így minden hiba megtalálható.
16. Hamis: A tesztelés során az első lépés mindig a hibakeresés.
17. Hamis: A funkcionális tesztelés kizárólag a szoftver biztonságára fókuszál.
18. Hamis: A regressziós tesztelés új funkciók hozzáadása után feleslegessé válik.
19. Hamis: Az automatizált tesztelés drágább, mint a manuális tesztelés, hosszú távon is.
20. Hamis: A szoftvertesztelés kizárólag az automatizált eszközökre támaszkodik, és a manuális tesztelés teljesen elavult.

1. A tesztelést nem elszigetelten végzik. IGAZ
2. Azoknál a rendszereknél, ahol a működésnek szerves része az emberi beavatkozás, ott természetesen az emberi hibát is számításba kell venni. IGAZ
3. A rendszereknek számos különböző környezetben kell működnie. IGAZ
4. Ha az adatok struktúrája sérül meg, akkor az felismerhető IGAZ
5. Az emberi tévedés NEM az egyik legkomolyabb hiba ok. HAMIS
6. A szoftverfejlesztés általában költséges tevékenység. IGAZ
7. A szoftverfejlesztés / tesztelés komoly szellemi tevékenység. IGAZ
8. A minőségbiztosítás önmagában NEM egy rendszer. HAMIS
9. A hibák gyakorisága attól függ, hogy a hibás funkció milyen gyakran van
10. használatban. IGAZ
11. A tesztelés nem nyújt költséghatékony módszert a hibák észlelésére. HAMIS
12. A tesztelést szerződéses vagy jogi követelménynek való megfelelés érdekében is megkövetelhetik. IGAZ
13. A mai korszerű autókban 30-nál kevesebb különálló számítógép van. HAMIS
14. Amikor egy rendszer elkészül, akkor meghatározásra kell kerüljön az a környezet, amelyben a megfelelő működés biztosított. IGAZ
15. A szoftverek meghibásodása származhat a külső körülményeken túl olyan okokból, amely az elkészítéskor elkövetett hibákra vezethetők vissza. IGAZ

16. A tesztelés feladata tömören az ügyfél elégedettségének növelése. IGAZ
17. Minél jobban ismerjük azt a körülményt, ahol a rendszert használni fogják, annál jobb lesz a használhatóság. IGAZ
18. Hordozhatóság szempontjából fontos, hogy a rendszer úgy épüljön fel, hogy változások esetén is tökéletesen működjön. IGAZ
19. A tesztelést minél hamarabb el kell kezdeni, előre meghatározott problémákat kiszűrve. IGAZ.
20. A hibák keresése, javítása haszontalan, ha a rendszer nem felel meg az ügyfél elvárásainak. IGAZ
21. A kimerítő teszt helyett kockázatelemzést és prioritásokat kell alkalmazni. IGAZ

1. gyakori félreértés a teszteléssel kapcsolatban, hogy teljes mértékben a teszt tárgyának verifikációjára fókuszál. I
2. A tesztelés hozzájárulása a sikerhez korlátozódhat a tesztcsoport tevékenységeire. H
3. A tesztelés és minőségbiztosítás mint fogalom ugyanazt jelenti. H
4. Hibákat csak az ember követhet el.H
5. A tesztmenedzsment felelősséget vállal a tesztfolyamatért, a tesztcsoportért és a teszttevékenységek vezetéséért.I
6. Hardver hiba lehet például egy olyan érzékelő meghibásodása, amely adatokkal lát el összetettebb
7. szoftvereket.I
8. A tesztelés során a kritikus rendszereket úgy kell vizsgálni, hogy az ilyen esetekben való működés is egyértelműen igazolható legyen.I
9. Nem célszerű olyan önellenőrző mechanizmusokat bevezetni, amelyek vizsgálják, hogy az aktuális konfiguráció érvényes-e.H
10. ha valamilyen ok miatt az adott struktúrában letárolt adatok megsérülnek (HDD vagy memória hiba) és a
11. rendszer erre nincs felkészítve, akkor ez a rendszer nem megfelelő működéséhez vezet. I
12. A szoftverek meghibásodása származhat olyan okokból, amely az
13. elkészítéskor elkövetett hibákra vezethetők vissza.I
14. A tesztelés feladata az, hogy a szoftver használata során fellépő hibák előfordulását véglegesen megszüntesse.H
15. Könnyű pontos leírást adni egy jó minőségű termékre.H
16. A minőségi szint egy olyan mértékrendszer, amely felett a rendszer elofagadható (jó) minőségű, és amely a feltételeket, amely alatt nem fogadható el (nem jó) minőségű.I
17. Funkcionális hibák esetén meghetározzuk azt a mennyiséget, típust és gyakoriságot, amely mellett még elfogadható minőségről beszélünk.I
18. A hibák gyakorisága nem a hibás funkció gyakoriságától függ.H
19. A tesztelést minél előbb kell elkezdeni.I
20. A kimerítő tesztelés csak felületi tesztelésre használják. H

21. A hibák megtalálása és javítása hasztalan, ha a kifejlesztett rendszer használhatatlan, és nem felel meg a felhasználók igényeinek, elvárásainak. I
22. A "féregirtó paradoxon" megjelenése ellen a teszteseteket rendszeresen felül kell vizsgálni.I
23. A tesztelés kimutatja a hibákat és ezáltal igazolja hogy egyáltalán nincsenek hibák. H

1. A tesztelés a minőségellenőrzés egyik formája. -I
2. A tesztelés a hibák jelenlétét mutatja meg - I
3. A tesztelésnek egyetlen alkalmazható megközelítése van. -H
4. Emberi hibák mindig meghibásodásokhoz vezetnek -H
5. Egyes számítógépek együttműködését a különböző hálózatok biztosítják. -I
6. A hálózat viselkedése változik a terhelés függvényében. - I
7. A konfiguráció segítségével befolyásolhatjuk a rendszerek működését. -I
8. Ha a környezet nem olyan, mint amire a rendszert tervezték, nem garantálható a rendszer megfelelő működése. -I
9. A rendszer input és output adatai zártak -I
10. Az emberi tévedés az egyik legkisebb hiba ok. -H
11. A folyamatos technológiai fejlődések, nincsenek hatással az ember tudására-H
12. A tesztelés feladata csak az, hogy a használat során fellépő problémákat csökkentse. -H
13. A használhatóság célja, hogy a felhasználói élmény minél rosszabb legyen. - H
14. A karbantarthatóság szempontjából, olyan megoldásokat kell keresni, hogy a rendszer átlátható legyen. - I
15. A tesztelést a lehető legkésőbb kell elkezdeni. - H
16. Ha ugyan azokat a teszteket hajtjuk végre, egy idő után nem fogunk hibát kapni.- I
17. A hibák gyakorisága attól függ, hogy a hibás funkció milyen gyakran van használatban. - I
18. Az smeret hiány és a nem megfelelő szervezés == időveszteség - I
19. A tesztelés teljes mértékben a teszt tárgyának verifikációjára terjed ki. -H
20. A kockázat az a tényező amely a jövőben negatív következményeket okozhat -I

1. hordozhatóság szempontjából fontos a rendszer felépítése úgy épüljn fel hogy az a környezet(hardver, szoftver) változásaira az előzetesen tervezettnek megfelelően tudjon reagálni Igaz
2. megbízhatósag esetén a rendszer funkcióinak működéséről beszélünk Igaz
3. Igaz: Az energiatermelés biztosítása érdekében az energiapiac teljes átalakítását szorgalmazzák.
4. Hamis: Az energiatermelést jelenleg kizárólag megújuló forrásokból oldják meg.
5. Igaz: A gázárrobbanás tovább növeli az inflációs nyomást Európában.
6. A programot azoknak a fiataloknak tervezték, akik úgy érzik, hogy nem elég önállóak, amikor probléma merül fel a tanulás vagy a hétköznapi élet során. Igaz
7. Az egyik legfontosabb célkitűzés, hogy ezek a fiatalok felismerjék, milyen erősségeik vannak, és azokat hogyan hasznosíthatják a mindennapi életben. Igaz

8. mentorálás során a tanulók egyéni igényeihez igazított módszereket alkalmaznak. Igaz
9. program keretében rendszeres egyéni és csoportos foglalkozásokon vesznek részt a diákok. Igaz
10. fiataloknak lehetőségük van arra, hogy kipróbálják magukat különféle szituációkban. Igaz
11. program alapját a tapasztalati tanulás módszere képezi, amely a gyakorlat közbeni tanulást helyezi előtérbe. Igaz
12. Az eredményességet a rendszeres visszajelzések és reflexiók segítségével mérik. Igaz
13. A tesztelés mennyiségének során figyelembe kell venni hogy milyen célra készült a rendszer Igaz
14. hatékonyság esetén azt vizsgáljuk, hogy az adott szoftver hogyan gazdálkodik az erőforrásokkal Igaz
15. a hatás impactr számítása folyamatonként elvégezhető vagy tapasztalati úton becsülhető Igaz
16. A tesztelés feladata az hogy a szoftver használata során fellépő hibák előfordulása csökkentse Igaz
17. korai tesztelést a szoftver vagy rendszerfejlesztési életciklusban a lehető legkorábban el kell kezdeni és előre meghatározott célokra kell összpontosítani Igaz
18. A tesztelést különböző körülmények között esetén különbözőképpen hajtjár végre Igaz
19. hibamentes rendszer téveszméje a hibák megtalálása és javítása hasztalan Hamis

1. A szoftver meghibásodások oka származhat a szoftver működési környezetéből. IGAZ
2. A környezetből származó hibák hatással vannak a szoftver működésére. IGAZ
3. A szoftverek nem minden esetben működnek valamilyen hardver környezetben. HAMIS
4. A hardver hiba nem az egész eszköz működésképtelenségéhez vezet. HAMIS
5. Hardver hiba lehet egy érzékelő meghibásodása. IGAZ
6. Az egyes számítógépek együttműködését a különböző hálózatok biztosítják. IGAZ
7. IP alapú hálózatnak hívjuk a LAN-t, és az internetet. IGAZ
8. A mai korszerű autókban 30-nál kevesebb számítógép van. HAMIS
9. Az adatbusz biztosítja a hálózatok közötti együttműködést. IGAZ
10. A hálózatok önmagukban kevés komponensből állnak. HAMIS
11. A hálózat viselkedése jelentősen változik a terhelés függvényében. IGAZ
12. A szoftver rendszer leállását "bedugulás" is okozhatja. IGAZ
13. Kapacitásfelesleggel ma már sok lokális hálózat nem rendelkezik. HAMIS.
14. A tesztelés feladatába nem tartozik az ügyfél elégedettsége. HAMIS
15. Külön területté nőtte ki magát a "használhatóság". IGAZ
16. Egy adott funkcionalitást minden programozó hasonló képpen készít el. HAMIS
17. A hordozhatóság nem egy fontos tervezési szempont. HAMIS
18. A tesztelés képes igazolni hogy nincsenek hibák. HAMIS
19. A" féregirtó paradoxon" megjelenése ellen a teszteseteket rendszeresen felül kell vizsgálni. IGAZ
20. A tesztelést a szoftver életciklusában a lehető legkésőbb kell elkezdeni. HAMIS

1. A szoftvertesztelés célja kizárólag a hibák felderítése.
2. A szoftvertesztelés része lehet a felhasználói igényeknek való megfelelés ellenőrzése.
3. A hibakeresés célja a hibák diagnosztizálása és javítása.
4. A minőségbiztosítás folyamatorientált, és megelőző jellegű megközelítés.
5. A tesztelés célja, hogy közvetlenül eltávolítsa a hibákat a szoftverből.
6. A kiváltó okok elemzése a hibák megelőzésének egyik eszköze.
7. A meghibásodásokat kizárólag hibák és emberi eredetű hibák okozzák.
8. A minőségellenőrzés célja, hogy javítsa a termék minőségét hibák kijavításával.
9. A tesztelés önmagában biztosítja a sikeres projekt megvalósítását.
10. A tesztelés célja az, hogy minden hibát felfedjen és biztosítsa, hogy a rendszer hibamentes legyen.
11. A hibák gyakran koncentrálódnak néhány rendszerkomponensben.
12. A tesztek mindig hatékonyak maradnak, ha ismételjük őket.
13. Minden tesztkörnyezetben ugyanazokat a tesztelési megközelítéseket kell alkalmazni.
14. A független tesztelők eltérő hibákat ismerhetnek fel, mint a fejlesztők.
15. A fejlesztők soha nem találhatnak hibákat a saját kódjukban.
16. A hálózati hibák ritkán okoznak problémát a szoftver működésében, mivel a hálózatok mindig stabilak.
17. A rendszer konfigurációjának helytelen beállítása meghibásodást okozhat a szoftver működésében.
18. A tesztelés során a hálózati eszközök mindig megfelelően vannak terhelve, és nem okoznak fennakadást.
19. A hálózat terhelésének növekedése bedugulást okozhat, ami a szoftver lassulásához vagy leállásához vezethet.
20. A szoftver megfelelő működéséhez elengedhetetlen, hogy a tervezett környezet megegyezzen a valós környezettel.