Szoftverfejlesztési módszerek és paradigmák kérdéssor

Fejlesztő eszközök hatékony használata

1. A fejlesztői eszköz kiválasztásánál milyen szempontokat érdemes figyelembe venni és miért?
2. Jellemezd egy hatékony tervezési, fejlesztési és javítási mikociklust (mind a hármat)! Mire érdemes odafigyelni?

Módszertanok

3. Adj kritikát a vízesés és az evolúciós szoftverfejlesztési modellekre!
4. Adj jellemzést a különböző szerződés modellekről!
5. Írd le, hogy a különböző szerződés modellek mire motiválják a résztvevőket!
6. Írd le, hogy a különböző szerződés modellek kockázataira milyen kockázat-mérséklési technikákat ismersz!
7. A különböző szerződés modellek milyen elvárásokat támasztanak a módszertannal szemben? Mit gondolsz az alkalmazásban megjelenő rugalmasságról?

Tesztelés

8. A tesztelésbe milyen szerepkörű résztvevők vannak bevonva? Mi a tipikus feladatuk a teszteléssel kapcsolatban? Jellemezd, hogy a projekt különböző fázisaiban mik a módosítások költségei!
9. Mi az ellenőrzés és mi a validáció? Mit tartalmaz a teszterv? Röviden jellemezd az elemeket! Mit értünk tesztelési stratégián?
10. Mi a unit tesztelés alapkoncepciója? Mik egy unit teszt tipikus lépései?
11. Ismertesd a jó unit teszt ismérveit, röviden jellemezd őket!
12. Röviden jellemezd az *NUnit* keretrendszert, ismertesd a főbb attribútumokat, használatukat, ellenőrző módszereket!
13. Jellemezd a *moq* keretrendszert! Add meg a főbb osztályok, mutasd be a legfontosabb kódolási mintákat (nem kell feltétlenül kód, elég a koncepció)!
14. Mi jelentenek az *IoC* és *DI* rövidítések? Mik a fogalmak jellemzői? Mit jelent ebben a kontextusban a resolution, resolution root, injection?
15. A DI keretrendszerek tipikusan milyen életciklus modelleket támogatnak? Mutasd meg a *dryIoc* keretrendszer néhány kapcsolódó funkcióját!
16. Írd le, hogy egy adatbázis alapú alkalmazáshoz milyen módon érdemes unit teszteket készíteni és miért?
17. Jellemezd a *Repository* mintát! Mi az előnye, hátránya más megoldáshoz képest? Milyen viszonyban van az ORM technológiákkal?
18. Jellemezd a property és konstruktor alapú DI megoldásokat!
19. Jellemezd a különböző *Factory* mintákat és *Service Locator* mintát *DI* szempontból! Hasonlítsd össze a két megoldást!
20. Írd le, hogy az alkalmazást hogyan kell strukturálni ha unit tesztelni szeretnénk? Milyen módszerek segítenek ebben? Mik a különböző megoldások előnyei, hátrányai?

Specifikációs módszerek és üzleti elemzés

21. Foglald össze az *Structured Systems Analysis And Design Method* (SSADM) módszer lényegét, fázisait, előnyeit és hátrányait!
22. Foglald össze a követelményelemzés céljait, módszereit, azok előnyeit és hátrányait!
23. Mutasd be a követelményelemzés során végzett aktivitásokat és azok jellemzőit!
24. Mutasd be a követelményelemzés során előtérbe kerülő különböző követelménytípusok jellemzőit!
25. Mutasd be a követelményelemzés során előtérbe kerülő kihívásokat!
26. Foglald össze az üzleti elemzés céljait!
27. Foglald össze az üzleti elemzés területeit, technikáit és jellemzőit!

Szoftver tervezési módszerek

28. Mutasd be a szoftvertervezés során alkalmazott alapelveket és koncepciókat!
29. Mi a szoftvermodellezés szerepe a szoftvertervezés során? Milyen módszereket, technikákat ismersz?
30. Miért használunk a modellezés során eltérő absztrakciós szintű modelleket? Mutass rá példákat!
31. Mutasd be a *Domain Driven Design*-t (DDD)! Térj ki a szakterület és a kód kapcsolatára!
32. Mutasd be a modellvezérelt fejlesztést, elemezed a jellemzőit!

Módszertanok, Klasszikus módszertanok

33. Mutasd be a szoftverfejlesztési folyamat lépéseit, röviden elemezd őket!
34. Hasonlítsd össze a vízesés, a spirál és az iteratív fejlesztési módszereket!
35. Mutasd be az iteratív és inkrementális fejlesztési módszerek jellemzőit!
36. Mutasd be a *Capability Maturity Model Integration* (CMMI) különböző területeinek céljait és jellemzőit!
37. Mutasd be a *Capability Maturity Model Integration* (CMMI) különböző szintjeit és azok jellemzőit!

Agilis fejlesztési módszerek

38. Foglald össze az agilis módszerek értékeit és elveit!
39. Foglald össze az agilis tervezés jellemzőit és szintjeit!
40. Mutasd be az agilis módszerek során alkalmazott kiadás tervezést és annak jellemzőit! Térj ki a felhasználói sztorik szerpére!
41. Foglald össze az agilis iterációk jellemzőit és az iteráció tervezés lépéseit!
42. Mutasd be a napi *stand-up* esemény és a “Kész, kész” (done, done) jellemzőit!
43. Foglald össze az *eXtreme Programming* (XP) jellemzőit és eszközeit!
44. Foglald össze az *eXtreme Programming* (XP) értékeit és elveit!
45. Mutasd be a *Scrum* fejlesztési folyamatot és jellemzőit!
46. Mutasd be a *Scrum* ceremóniák jellemzőit!
47. Mutasd be a *Scrum* szerepköröket és a sprint tervezés jellemzőit!

Forráskód kezelő eszközök

48. Mik az előnyei a verzió kezelő eszközök használatának? Milyen verzionálási modelleket ismersz, mik ezek előnyei, hátrányai?
49. Ismertesd a központosított és elosztott verziókezelés előnyeit, hátrányait!
50. Milyen elágaztatási stratégiákat ismersz? Jellemezd őket röviden! Jellemezd Git legfontosabb fogalmait!
51. Jellemezd a git legfontosabb objektumait, adj példát egy módosítási művelet során végbemenő változásokra!
52. Jellemezd a git rebasing és a merge műveleteit!

Felhasználó központú tervezés

53. Mit jelent a felhasználói élmény? Elemezd a kapcsolódó fogalmakat!
54. Mit jelent a perszóna? Mutasd be a perszónák és a use case-ek kapcsolatát!
55. Mit jelentenek a következő fogalmak: sketch, wireframe, vizuális terv, prototípus?
56. Mit jelent a használhatósági tesztelés? Milyen típusai vannak? Jellemezd őket!

Projektmenedzsment

57. Jellemezd a szervezeti stratégiát és viszonyát a projekthez!
58. Jellemezd a lineáris-funkcionális szervezeti formát!
59. Jellemezd a projektorientált szervezeti formát!
60. Jellemezd a mátrix szervezeti formát!
61. Írj a projekt tipológiáról!
62. Mi a projekt karakterisztika, kik egy projekt résztvevői?
63. Sorold fel a PMBOK folyamatcsoportjait és tudásterületeit!