Beágyazott Operációs Rendszerek ZH 2016. április. 28.

1. 8 pont
    a. Mi a különbség a preemptív és a nem preemptív ütemező között?
        - A preemptív ütemező megszakíthatja a taszkok futását
    b. Miért nem használjuk általában a malloc() függvényt beágyazott rendszerekben?
        - Mert, nem kiszámítható a futásideje
    c. Mit nevezünk kritikus szakasznak?
        - Azt a kódrészletet, amelynek valamilyen oknál fogva atomi módon kell lefutnia
    d. Milyen módszerrel lehetséges a közös erőforrás problémát megoldani beágyazott operációs rendszerekben?
        - Megszakítások tiltása
        - Ütemező tiltása
        - Védett globális változók használata
        - Szemaforok alkalmazása
2. Igaz-Hamis µCOS-II 16 pont
    a. Alapvetően C++ nyelven készült H
    b. Konfigurálása egy külön segédprogrammal történik H
    c. Futtatható x86 architektúrájú asztali PC-n I
    d. Preemptív ütemezési algoritmust használ I
    e. Több taszknak is lehet azonos prioritása H
    f. Nincs felkészítve a prioritás jelenségének kezelésére H
    g. A taszkok prioritása futásidőben módosítható I
    h. Forráskódja nem tartalmaz assembly részeket H
    i. Tartalmaz memória menedzsment szoláltatást I
    j. A taszkok „Várakozik” állapotban vannak, amikor a processzor felszabadulására várnak H
    k. Az ütemezőt nem lehet letiltani H
    l. Az összes megszakítást az operációs rendszer kezeli le H
    m. Támogatja a pipe jellegű üzenetküldést taszkok között H
    n. A konfigurálás során egyebek mellett az operációs rendszer RAM és ROM felhasználását tudjuk befolyásolni I
    o. Ütemezési algoritmusa tartalmaz ciklust H
    p. A szabad taszkleírók egy láncolt listában tárolódnak I
3. Linux jogosultságok 16 pont
    a. Igen, mert a törlés könyvtár jog és a megnyitott könyvtárban van írás joga a felhasználónak, valamint a könyvtárba el is tud jutni a felhasználó.
    b. Nem, mert nincs írás joga a felhasználónak
    c. Igen, mert hardlink
    d. Igen, mert van olvasás joga viszont ismernie kell a mappa nevét, mert fölötte lévő könyvtárba nem lát be.
4. Linux mappák 8 pont
    a. /bin Alapvető programok
    b. /dev Eszközállományok
    c. /etc Konfigurációs állományok

    d. /proc Kernel interfész
    e. /sys Kernel interfész programoknak, vagyis hozzáférés az eszközhöz
    f. /sbin Rendszergazdai programok
    g. /tmp Ideiglenes, átmeneti fájlok
    h. /usr/lib Fejlesztői könyvtárak
5. Shell funkciók 8 pont
    a. Kimenet állományba irányítása felülírással
        - parancs > kimenet
    b. Hibakimenet állományba irányítása hozzáfűzéssel
        - parancs 2>> kimenet
    c. Csővezeték
        - parancs1 | parancs2
    d. Parancssorozat, amelynél ha az első parancs sikertelen, akkor fusson le a második
        - parancs1 || parancs2
    e. Parancshelyettesítés
        - du `cat p.txt`
        - du $(cat p.txt)
    f. Változó értékeknek megadása
        - var=alma
    g. Állomány írhatóságának tesztelése
        - test –w allomany
    h. Numerikus egyenlőség vizsgálata
        - test num1 –eq num2
6. 10 pont
    a. Kernel
        - Hardver kezelése
    b. Glibc
        - Alapvető C library, vagyis C függvények gyűjteménye
    c. Busybox
        - Alapvető rendszerprogramok
    d. Grub
        - Bootmanager
    e. Dropbear
        - SSH-hoz szükséges program
7. Yocto 10 pont (bocsi de ezt nem sikerült gyorsan leírni)
    a. bblayers.conf szerkesztése
    b. local.conf szerkesztése
    c. ../oe-init-build-env
    d. bitbake nano
    e. bitbake core-image-base
    f. bitbake recept szerkesztése
8. Makefile 8 pont (bocsi de ezt nem sikerült gyorsan leírni)
    a. explicit szabály
    b. helyettesítő szabály
    c. feltételes értékadás

d. változó hivatkozás
e. egyszerű kiértékelés
f. általános minta szabály
g. hibás példa
h. automatikus változó

9. Igaz-hamis 8 pont
    a. A close() függvény visszatérhet hibával I
    b. Beállítható, hogy a read() függvény ne blokkolódjon I
    c. A poll() függvény mindig blokkolódik I
    d. Az lseek() függvénnyel pozícionálhatunk a fájl elejére H
    e. A chown() függvény használatakor törlődik a sticky bit H
    f. A rename() függvény csak a partíciókon belül működik I
    g. A pthread_cond_wait() függvény, akkor tér vissza, mikor a megadott feltétel teljesül H
    h. Az mlockall() függvény képes kiiktatni a memória lapcsere algoritmust I
10. TCP függvények 8 pont
    a. Szerver
        - socket()
        - bind()
        - listen()
        - accept()
    b. Kliens
        - socket()
        - connect()
    c. Kommunikáció
        - receive() (ssize_t recv())
        - send() (ssize_t send())