LINUX RENDSZERFEJLESZTÉS ÉS ÜZEMELTETÉS

7. forduló

ONE IDENTITY by Quest

A kategória támogatója: One Identity - Quest Hungary

Ismertető a feladathoz

Felhasznált idő: 10:00/10:00 **Elért pontszám:** 0/8

1. feladat 0/2 pont

A Fedora 36-on fejlesztett és remekül futó shell scriptem Ubuntu 22.04-en "./x.sh: 4: [: test: unexpected operator" hibát produkál. Sikerült a scriptet egészen kicsi méretre redukálni, ami még mutatja ugyanezt a jelenséget. **Mi a gond, és hogyan lehetne kijavítani, hogy Ubuntun is fusson?**

```
#!/bin/sh

a="test"
if  [ $a == "test" ] ; then
        echo "TEST"
fi
```

Válasz

- ○ Az Ubuntu nem engedi feltételes kifejezésben idézőjel nélkül változók használatát, a $a köré idézőjeleket téve már működni fog a script
- ● A feltételes kifejezés [használatával obsolete-nek számít, az Ubuntu a 22.04-es verziótól kezdődően már nem is fogadja el, dupla szögletes zárójelet ([[ ) kell használni. A Fedora ebben a tekintetben sokkal megengedőbb, de várhatóan ott is meg fog szűnni ennek a szintakszisnak az elfogadása a 37-es vagy 38-as verzióban.
  Ez a válasz helytelen, de megjelölted.
- ● Ubuntun a /bin/sh alapesetben egy symlink a /bin/dash-re, ami nem ismeri ezt a feltételes szintaxist. Fedorán a /bin/sh a bashre mutat, ezért működik. Megoldás a shebang átírása #!/bin/bash -re, így maga a script változatlan maradhat.
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

Mivel az „a” változóban whitespace nélküli érték van, nem gond az idézőjel hiánya, és a [parancs is remekül működik Ubuntun is.

2. feladat 0/6 pont

C nyelvű (gcc 9-cel fordított) programunkban a **main** függvényt tartalmazó file-ban egy globális "FILE* logfile" változón keresztül érjük el a logolásra használt file-unkat. Szeretnénk a log file-t a programból való kilépéskor lezárni, de sajnos a rengeteg forrásfile-ból álló, másoktól megörökölt kódban sok helyen **exit()** hivásokkal lép ki a program.

Hogyan tudnánk megoldani, hogy mindenképpen automatikusan lezárjuk a log file-t, bárhonnan is lépünk ki?

Válaszok

- [x] A __cleanup__ gcc attribútumot megadjuk a "logfile" változónak:

```
void close_logfile(FILE* fp)
{
  fclose(fp);
}
FILE *logfile __attribute__ ((__cleanup__(close_logfile)));
```

Ez a válasz helyes, de nem jelölted meg.

- [x] Az atexit()-et rairanyitjuk a close_logfile-ra a main függvény elején:

```
void close_logfile()
{
  fclose(logfile);
}
atexit(close_logfile);
```

Ez a válasz helyes, de nem jelölted meg.

- [x] A close_logfile függvényhez megadjuk a destructor gcc attribútumot:

```
__attribute__((destructor))
void close_logfile()
{
  fclose(logfile);
}
```

Ez a válasz helyes, de nem jelölted meg.

- [ ] A program kilépésekor érkező SIGQUIT signalra rákötünk egy signal handlert ami lezárja a file-tt:

```
void sighandler(int signum)
{
      fclose(logfile);
}
signal(SIGQUIT,sighandler);
```

a main függvényben:

signal(SIGQUIT,sighandler);

Magyarázat

A __cleanup__ használata azért nem jó, mert globális változóra nem hívódik meg (függvényen belüli lokális változónál ez is működő megoldás lehetne). A SIGQUIT signal pedig nem hívódik meg automatikusan.

Megjelenés

Világos