SAKARYA ÜNİVERSİTESİ - BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
İŞLETİM SİSTEMLERİ PROJE ÖDEVİ

KABUK - SHELL

Teslim Tarihi: 30 Kasım 2021 Salı, Saat: 29:59

Programlama dili: Projede sadece C dili kullanılabilir.
Diğer kısıtlar: Proje kapsamında `system()` sistem çağrısı kullanılamaz.

Amaç

Bu projenin amacı, bir kabuk arayüzünün geliştirilmesi yoluyla proses kontrolünün mekaniğine aşina olmanızı sağlamaktır. Proje çalışması, ebeveyn ve yavru prosesler arasındaki ilişkiyi, yeni bir proses oluşturmak için gereken adımları, kabuk değişkenlerini ve kullanıcı girdisini ayrıştırma işlemlerini içerir.

Bu proje için daha önce ilan edilen şekilde grup halinde çalışabilirsiniz. Geliştireceğiniz kabuk (Shell) Linux işletim sistemi üzerinde çalıştırmalısınız. Teslim edilen tüm projeler Linux üzerinde test edilecektir.

Değerlendirme

Tüm proje notu sınıf notunuzun %10'u değerindedir. Kopya ödevler 0 (sıfır) ile notlandırılacaktır. Ödevler GiHub'a grup temsilcisi tarafından yüklenecektir. Yüklenen proje dosyasının adı grup adı olacaktır. Ders yürütücülerine ve ders yardımcılarına GitHub üzerinde proje dosyalarına erişim yetkisi (sadece okuma) verilecektir.

Problem ifadesi

Aşağıda özellikleri verilen, bir dizi yerleşik işlevin (buit-in functions) yürütülmesini ve diğer proseslerin çalıştırılmasını sağlayan temel bir kabuk arabirimi tasarlanması ve kodlanması. Kabuk robust olmalıdır, örneğin, donanımsal bir arıza dışında hiçbir koşulda çökmemelidir.

Bölüm 1: Genel Kabuk Yapısı

Kabuk, sürekli olarak kullanıcı girdisi isteyen, belki de kullanıcı adına bir şeyler yapan, sonra kendini sıfırlayan ve tekrar kullanıcı girdisi isteyen bir programdır. Örnek:

```
while(1)
{
          */ Get user input */               */
Exit? */
          */ Do something with input */
}
```

Bölüm 2: Prompt

Prompt, kabuğun kullanıcıdan gelen girdiyi kabul etmeye hazır olduğunu belirtmesi için kullanılan semboldür. Çoğu zaman, kabuğu çalıştıran kullanıcının adı ve geçerli dizin gibi yararlı bilgileri de gösterir. Şimdilik, basit bir prompt kullanmanız yeterlidir, örneğin "" gibi.

• Prompt aşağıdaki gibi görünmelidir:

```
sau >
```

• Kullanıcı girişinin görsel olarak prompt'a karışmaması için '>' işaretinden sonra bir boşluk olmalıdır.

• İlave not – farklı renk(ler) yapın!

• İlave not – prompt, geçerli çalışma dizinini de göstermeli:

```
/home/bin/: sau >
```

Bölüm 3: Komut satırının çözümlenmesi (parse)

Kabuk komutları yürütmeye başlamadan önce, komut adını ve parametrelerini ayıklaması gerekir. Bu komutları ve parametreleri bir dizide saklamak güzel olabilir, böylece her birini sırayla ayrıştırabilirsiniz. Kabuğumuzda, ilk parametre her zaman yürütmek istediğimiz programın adı, kalan tüm kelimeler (belki de ilk kelime dahil) o programın parametreleri olacaktır.

Aşağıdaki varsayımları dikkate alın:
• Komut önünde asla boşluk yok.
• Komut satırındaki kelimeler bir boşluk ile ayrılmıştır.
• Komut sonunda boşluk yok.
• Komut en fazla 80 karakter olabilir.
• Bir komutun boşlukla ayrılmış en fazla 10 kelimeden oluşabilir.

Test çalışmaları yaparken, kabuk döngünüzden girilen farklı komutlar için, komut ve parametrelerini oluşturan diziyi başarıyla yazdırabildiğinizden emin olun. Komutlarınızdan veya parametrelerinizden herhangi birinde bozulma/çöp görürseniz, komut ve parametre dizinizi kullanmadan önce ve/veya kullandıktan hemen sonra temizlemek için C kitaplığı çağrısı `memset()` veya `bzero()`'yu kullanmayı deneyebilirsiniz.

C kütüphanesi çağrısı `fgets()`, ekrandan kullanıcı girdisini toplayabilir ve bunu bir diziye (C karakter dizisi) kaydedebilir. Daha fazla bilgi için `fgets` için `man` sayfalarına bakın.

Bölüm 4: Komut icrası

Kabuk, hangi komutların yürütüleceğini anladıktan sonra, basit komutların yürütülmesi test edilebilir. Başka bir prosesin yürütülmesi yeni bir proses oluşturmayı gerektirdiğinden, yeni bir proses oluşturmak için `fork()` sistem çağrısını kullanmanız gerekecektir. Yeni prosesi oluşturduktan sonra, buna yeni bir kod yüklemek için `execvp()` sistem çağrısını kullanabilirsiniz. Son olarak, ebeveyn proses (kabuk), `waitpid()` sistem çağrısı ile yavru prosesin görevini tamamlamasını beklemelidir.

Ancak, bir hata oluşursa execvp() sistem çağrısı hata dönebilir. Bu durumda, kabuk hata mesajı yazdırmalı ve yeni giriş istemelidir. Örnek:

```
sau > lalala -a
Hata: Komut icra edilemiyor.
sau >
```

Bölüm 5: Built-in komutlar

Tüm komutlar aslında program değildir ve kabuğunuz en az iki "buit-in" komut bulundurmalıdır. Başka bir deyişle, bu iki komuttan herhangi biri için `fork(), exec()` ve `waitpid()` kullanılmayacaktır. Bunun yerine, kabuğunuz aşağıdaki işlevleri uygulayan bir fonksiyon çağırmalıdır:

- **exit** – çalışan kabuk prosesi sonlandırılır ve alt satıra 'exit' yazdırılır.

```
Sau > exit
exit
        (kabuk sonlanır)
```

- **cd [PATH]** – Mevcut çalışma dizinini değiştirir. `chdir()` sistem çağrısını kullanmanız ve `PWD` çevresel değişkenini `setenv()` ile güncellemeniz gerekecektir.

```
sau > pwd
/user/sau/os/project_1
sau > cd ..
sau > pwd
/user/sau/os
sau > cd project_1
sau > pwd
/usr/sau/os/project_1
```

- **showpid** – Kabuk tarafından oluşturulmuş en az 5 ***yavru*** proses PID'sini ekrana yazdırır. Örnek:

```
sau > showpid
4987
4992
5001
5002
5004
```

README dosyası

Lütfen aşağıdakileri içeren bir README metin dosyası oluşturun:

• Grubunuzdaki tüm üyelerin adları
• Gönderiminizdeki tüm dosyaların/dizinlerin bir listesi ve her birinin kısa bir açıklaması
• Programlarınızı derlemek için talimatlar
• Programlarınızı/komut dosyalarınızı çalıştırma talimatları
• Geliştirme sırasında karşılaştığınız zorluklar
• Programı yazmanıza yardımcı olması için kullandığınız kaynaklar

Değerlendirme Rubriği (100 puan üzerinden)

İSTER	**PUAN**
1. Komut satırı doğru bir şekilde çözümleniyor (parsing)	15
2. Hiç zombi proses oluşmuyor (`waitpid` doğru kullanılmış)	5
3. `fork` doğru kullanılmış ve çalışıyor	15
4. `execvp` kullanılmış (harici program çalışıyor)	15
5. Built-in "`cd`" çalışıyor	15
6. Built-in "`showpid`" çalışıyor	15
7. README tamam	10
8. Program okunaklı ve yorumlar uygun şekilde yapılmış	10
TOPLAM	100

Eğer program derlenmezse proje notunuz grup olarak 0 (sıfır) olacaktır.