SHELL PROGRAMLAMA

Arş. Grv. Muzaffer Kaan Yüce

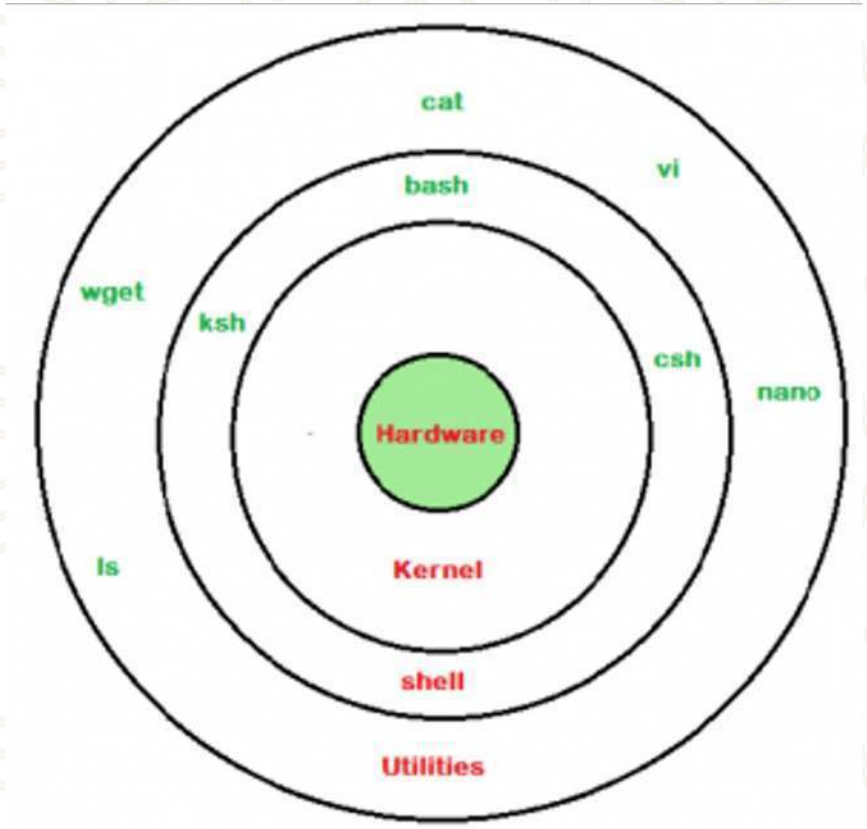
Shell nedir

Kernel : İşletim sistemi çekirdeğini oluşturan programdır.

- File Management
- Process Management
- I/O & Memory Management
- Device Management

Shell : İşletim sistemi servislerine erişmek için kullanılan programdır.

- BASH(Bourne Again **SH**ell)
- CSH(C **SH**ell)
- KSH(Korn **SH**ell)



Bazı işlemler

Klasör işlemleri:
mkdir <Klasör Adı>: klasör oluşturma
cd <Path/Klasör Adı> : klasör konumuna gitme
ls : mevcut klasördeki dosyaları listeler
rmdir <Klasör Adı> : boş klasörü silme

Dosya işlemleri
touch <dosya_adı.uzantısı>: text dosyası oluşturur
rm <dosya adı>: dosyayı siler
mv <dosya adı> <yeni konumu>: dosya/klasörü taşır
cat: dosyaların içeriğini ekrana yazdırır
nano: dosyaların içeriğini düzenlemeyi sağlar

Kullanıcı Yönetimi

Kullanıcı oluşturma

- Yeni kullanıcı oluşturma
  sudo useradd -m username -p

- Yeni kullanıcı oluşturma ancak,
  sudo useradd -M USERNAME
  sudo usermod -L USERNAME

- Kullanıcı silme
  userdel -r username → remove user
  userdel -r -f username → remove user with its all files

Kullanıcı Yönetimi

Grup oluşturma

- Group creation

  sudo groupadd testGroup

- Add a user to a group

  sudo usermod -a -G testGroup testUsers

- Check existing group

  cat editorial **/**etc**//**group **/ grep** username

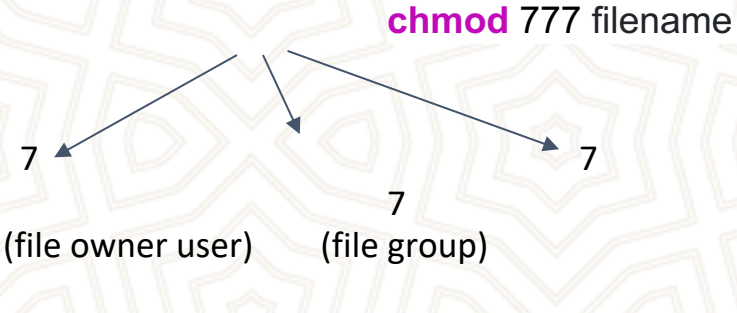
- Remove Group

  groupdel groupname

Dosya Yönetimi

- Dosya sahipliği
  chown :username filename
  chown :groupname filename

- İzinler : 4 - read , 3 - write , 1 - execute



- File Types:
  - d(directory)
  - c(character device)
  - l(symlink)
  - p(named pipe)
  - s(socket)
  - b(block device)
  - D(door, not common on Linux systems, but has been ported)

```
parallels@parallels-Parallels-Virtual-Platform:~/Desktop$ ls -l
total 12
lrwxrwxrwx 1 root      root        10 Oct  6 19:35 'Parallels Shared Folders' -> /media/psf
drwxrwxrwx 2 parallels parallels 4096 Oct  9 14:16  shellprog
drwxrwxrwx 2 parallels parallels 4096 Oct  8 09:04 'shellprog  1'
-rwxrwxrwx 1 parallels parallels   19 Oct  9 14:14  test.sh
parallels@parallels-Parallels-Virtual-Platform:~/Desktop$
```

Shell Programlama - 1. “Hello World”

```
1. clear
2. echo "Hello, world"
```

Shell Programlama - 2. Info Echo

```
1. #
2. # info dosyasi #
3. clear
4. # Oturum acmis olup o an terminali kullanan kullaniciyi ekrana yazdirir
5. echo "Merhaba $USER"
6. # date degiskeni anlik tarih bilgisi icindir. echo ve date komutlari arasindaki noktali
   virgul ( ; ) birden fazla komutu ard arda calistirabilmemize olanak tanir. Calistirilan
   komutlar birbirinden bagimsizdir.
7. echo "Bugun \c" ; date
8. # who degiskeni oturum acmis olan tum kullanicilari listeler pipe ( | ) ile birden fazla
   komutu ard arda isleme koyabiliriz ve soldan saga islenen komutlar, bir onceki komutun
   ciktisini parametre olarak alirlar
9. echo "Oturum acmis kullanici sayisi : \c" ; who | wc -l
10.# cal degiskeni ay bazinda takvimi gosterir
11.echo "Takvim" cal
```

Shell Programlama - 3. Echo Tipleri

```
1. #
2. # echo_secenekleri dosyasi #
3. clear
4. # ekrana basildiginda bir uyari sesi cikarir
5. echo -e "deneme yazi \a"
6. # eger yazi arasinda ise kendinden onceki bir karakteri siler
7. echo -e "deneme yazi \b"
8. # ekran ciktisinin sonunda yer alan yeni satiri siler
9. echo -e "deneme yazi \c"
10.# ekran ciktisinin sonuna bir yeni satir ekler
11.echo -e "deneme yazi \n" # satirbasi acar
12.echo -e "deneme yazi \r"
13.# bir tab tusu kadar bosluk birakir
14.echo -e "deneme yazi \t"
15.# \ karakterinin yazilabilmesi
16.echo -e "deneme yazi \\"
```

Shell Değişkenleri

İki tür Shell değişkeni vardır:

- **System variables** → Büyük harflerden oluşur, halihazırda sistemde bulunur ve sistem bileşenlerini gösterir.
- **User variables** → Kücük harflerden oluşması beklenen kullanıcı değişkenleridir. Kullanıcı tarafından tanımlanır.

SHELL	/bin/bash	Shell adı
COLUMNS	204	Terminal ekranının sütun sayısı
LINES	24	Terminal ekranının satır sayısı
HOME	/home/parallels	Home klasörünün konumu
USER	parallels	Kullanıcı adı
OSTYPE	linux-gnu	İşletim sistemi tipi
PATH	/usr/local/sbin:/usr/local/bin	Path klasörünün konumu
PWD	/home/parallels/Desktop/shellprog	Bulunduğumuz klasörün konumu

Shell Programlama - 4. Değişken tanımlama

```
1. # degisken_tanimla dosyasi #
2. clear
3. # Degisken isimleri alt cizgi (_) veya harf ile baslar, esittir isaretinden once ve sonra
   bosluk konulmaz
4. # Degisken isimleri buyuk ve kucuk harflere duyarlidir
5. # Yazim aninda degeri belli olmayan degiskenler icin NULL degeri, degisken isminden sonra
   esittir konularak verilebilir
6. degisken0=
7. degisken1=10
8. degisken2="deneme"
9. degisken3=Deneme
10.echo $degisken1
11.echo $degisken2
12.echo $degisken3
13.# yukaridaki degiskenlerin degerleri kullanici tarafindan verildi. Degiskenlere ayni
   zamanda sistem degiskenleri de eklenebilir
14.degisken4=$pwd
15.echo $degisken4
16.
```

Shell Programlama - 5. Arithmetics

```
1. #
2. # aritmetik_islemler dosyasi #
3. clear
4. expr 1 + 2 # toplama
5. expr 3 \* 4 # carpma
6. expr 2 - 1 # cikarma
7. expr 10 % 3 # kalan alma (bazi terminallerde \% seklinde yazmak gerekebilir)
8. echo `expr 3 + 4` # Burada back quote kullanilir. Bu sembol ~ tusunun altinda yer alir
```

Shell Programlama - 6. Veri girişi

```
1. #
2. # kullanici_veri_girisi dosyasi #
3. clear
4. echo "Lutfen adinizi giriniz"
5. read kullanici_adi
6. echo "Merhaba $kullanici_adi !"
7.
```

Shell Programlama - 7. Eşleştirmeler

```
1. #
2. # eslestirme dosyasi #
3. clear
4. ls * # tum dosyalar listelenir
5. ls a* # a ile baslayan tum dosyalar listelenir
6. ls *.py # uzantisi .py olan tum dosyalar listelenir
7. ls deneme_*.py # deneme_ ile baslayip uzantisi .py olan tum dosyalar listelenir
8. ls ? # tek karakterli adi olan tum dosyalar listelenir
9. ls deneme? # deneme ile baslayip ardindan yalnizca bir karakter gelen tum dosyalar
   listelenir ls [de]* # d veya e ile baslayan tum dosyalar listelenir
10.
```

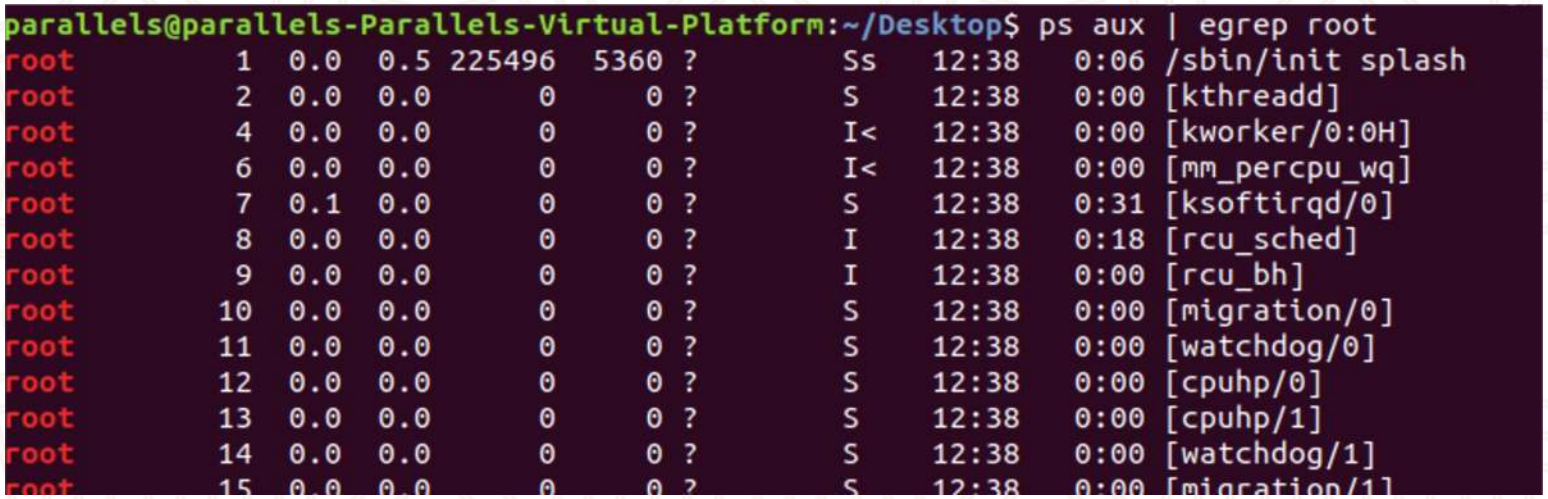
Shell Programlama - 8. Dosya işlemleri

```
1. # cesitli_dosya_islemleri dosyasi #
2. mkdir klasor_ismi cd klasor_ismi touch deneme
3. ls deneme
4. mv deneme deneme.js
5. ls deneme # hata mesaji verecek
6. ls deneme.js
7. echo "dosyaya yaz beni" > deneme.js cat deneme.js
8. touch deneme2
9. echo "1" > deneme2
10.echo "3" > deneme2 # > isareti dosyaya bastan yazar
11.echo "1" >> deneme2 # >> isareti, dosyaya yeni bir satir ile yazmaya devam eder echo "7"
   >> deneme2
12.echo "5" >> deneme2
13.sort deneme2 # satirlari siralar (A-Z , 0-9, vs.) ancak dosyayi guncellemez sort deneme2 >
   deneme3 # siralanmis dosyayi yeni bir dosyaya yazar rm deneme2 # dosyayi siler
14.
```

Shell Programlama - 9. Pipe ile komutları birbirine bağlamak

“|” (Pipe) karakteri ile birden fazla komut birbirine bağlı olarak çalıştırılabilir.
Bu komut ile bir komutun çıktısı diğer bir komuta parametre olarak verilebilir.

1. *#*
2. *# pipe dosyasi #*
3. **ps aux** *# ps aux ile anlik olarak kullanicinin calisan islemlerini satir satir gorebiliyoruz*
4. **egrep 1 deneme2** *# egrep ile istedigimiz bir klasorde veya dosyada, dosyalar icinde bulunan belli metinleri filtreleyebiliyoruz. Buradaki islem, deneme2 ‘nin icinde 1’i filtrelemektir.*
5. **ps aux | egrep root** *# Iki komutu birlestirdigimizde su sonuc cikar : icerisinde root kelimesi gecen satirlari filtrele. Ilgili aramayi da ps aux komutu ciktisinden edin.*



Shell Programlama - 10. TR komutu ile karakter dönüştürme

```
1. # tr_komutu dosyasi #
2. # asagidaki komuttan sonra tr, girilen kucuk harfli karakterleri buyuk harfe cevirecektir
3. echo "col1;col2;col3;col4" | tr a-z A-Z
4. # Birebir karakter bulma ve degistirme icin asagidaki komut kullanilabilir
5. echo "col1;col2;col3;col4" > birdosya.txt
6. cat birdosya.txt | tr ';' ',' # tr, icerigini aldigi dosyadaki noktali virgulleri virgul
   ile degistirecek
7. echo "col1;col2;col3;col4" > birdosya.txt
8. cat birdosya.txt | tr -d ';' # tr, icerigini aldigi dosyadaki tum noktali virgulleri
   silecek
```

Shell Programlama - 11. Sütun Değer Kıyaslaması (Awk)

```
1. #
2. # awk_komutu dosyasi #
3. # asagidaki komuttan sonra tr, girilen kucuk harfli karakterleri buyuk harfe cevirecektir
4. echo "ahmet;4;1" > birdosya.txt echo "ahmet;3;2" >> birdosya.txt echo "ahmet;2;3" >>
   birdosya.txt echo "ahmet;1;4" >> birdosya.txt echo "ahmet;0;5" >> birdosya.txt
5. cat birdosya.txt | awk -F ';' 'int($2)>3' # ikinci sutunu 3ten buyuk olan satirlar
6. cat birdosya.txt | awk -F ';' 'int($3)<=4' # ucuncu sutunu 4ten kucuk veya esit olan
   satirlar
7.
```

Shell Programlama - 12. Bazı Operatörler

```
1. #
2. # hesap_makinesi dosyasi #
3. bc
4. 3>1 # 1
5. 3<1 # 0
6. 1==1 # 1
7. 1+1 # 2
8. 3-1 # 2
9. 5%5 # (Kalan) 0
10.
```

Shell Programlama - 13. Sorgular

```
1.  # sorgular dosyasi #
2.  # $0 = shell script dosyamizin adi ,
3.  # $1 = scripti calistirirken komut satirina girdigimiz birinci
    parametre ,
4.  # $2 = scripti calistirirken komut satirina girdigimiz ikinci
    parametre ..
5.  touch birdosya.txt # Dosya olusturduk
6.  echo "Ornek bir metni dosyaya koyduk" > birdosya.txt # dosyaya veri
    ekledik
7.  if cat $1 # eger cat $1 komutu sonuc dondurur ise (exit status 0)
8.  then # bu durumda
9.  echo "$1 isimli dosya mevcut" # bu komutu calistir
10. else # eger degil ise
11. echo "$1 isimli dosya mevcut DEGIL" # bu komutu calistir
12. fi # sorguyu tamamla
13. # [ expr ] komutu iki degeri kiyaslamak icin kullanilir. Sonuc dogru
    ise 0, # degil ise sifirdan farkli bir deger dondurulur
14. if [ $# -gt 3 ] # eger girilen toplam parametre sayisi 3ten buyuk
    ise
15. then # bu durumda
16. echo "$# adet parametre girdiniz" # bu komutu calistir
17. else # eger degil ise
18. echo "$# adet parametre yeterli degildir" # bu komutu calistir
19. fi # sorguyu tamamla
20. # test komutu iki degeri kiyaslamak icin kullanilir. Sonuc dogru ise
    0,
21. # degil ise sifirdan farkli bir deger dondurulur
```

...

...

```
22. if test $3 = $4 # eger $3 parametresi ile s4 ayni degerde
    string ise
23. then # bu durumda
24. echo "$3 ile $4 ayni" # bu komutu calistir
25. else # eger degil ise
26. echo "$3 ile $4 farkli" # bu komutu calistir
27. fi # sorguyu tamamla
28. # birden fazla alt alta sorgu yapmak icin elif ( else if )
    komutu kullanilir
29. if [ -w $1 ] # eger $1 bir dosya ve okunabilir ise
30. then # bu durumda
31. echo "$1 yazilabilir bir dosyadir" # bu komutu calistir
32. elif [ -r $1 ] # ilk kosul saglanmadiysa buna bakalim
33. then # bu durumda
34. echo "$1 yazilamaz ama okunabilir bir dosyadir" # bu komutu
    calistir
35. else # eger degil ise
36. echo "$1 ne yazilabilir ne de okunabilir bir dosyadir" # bu
    komutu calistir
37. fi # sorguyu tamamla
```

Shell Programlama - 14. Döngüler -1

```
1.  # donguler dosyasi #
2.  # for icin temel kullanim. $i degiskeni, for icindeki i sayacini
    ifade eder
3.  for i in 1 2 3 4 5
4.  do
5.  echo "sayac $i"
6.  done
7.  # yukaridaki ile ayni sonucu cikarir
8.  for i in {1..5}
9.  do
10. echo "sayac $i"
11. done
12. # yukaridaki ile ayni sonucu cikarir
13. for ((i=1; i<=5; i++))
14. do
15. echo "sayac $i"
16. done
17. # 1'den baslar 10'a kadar 2'ser 2'ser sayar
18. for i in {1..10..2}
19. do
20. echo "sayac $i"
21. done
22. # $(..) kalibi ile bir shell komutunun ciktisini for icin
    kullanabiliriz
23. for i in $(ls)
24. do
25. echo "dosya $i"
26. done
```

...

...

```
27. # liste cikaracak her komutu for icin kullanabiliriz. Asagidaki ~/
    (home) dizini
28. # altindaki dosyalari ve klasorleri listeler. Dolayisi ile her bir
    dongude i, dosya ismi olacaktir
29. for i in ~/*
30. do
31. if [ -f $i ]
32. then
33. echo "dosya : $i"
34. elif [ -d $i ]
35. then
36. echo "klasor : $i"
37. else
38. echo "bilemedim bu ne : $i"
39. fi
40. done
```

Shell Programlama - 15. Döngüler -2

```
1.  #
2.  # donguler2 dosyasi #
3.  i=10 # sayacimiza ilk degeri verdik
4.  while [ $i -gt 0 ]
5.  do
6.  echo "sayac $i"
7.  i=`expr $i - 1` # sayacimizi bir azalttik
8.  done
9.
10. counter=6
11. until [ $counter -lt 3 ]; do
12.     let counter--=1
13.     echo $counter
14. done
```

Shell Programlama - 16. Case komutu

```
1. #
2. # case dosyasi #
3. read ne_ariyorum
4. case $ne_ariyorum in
5. "arac") echo "arac icin www.arac.com";; # degiskenimiz arac ise
6. "ev") echo "ev icin www.ev.com";; # degiskenimiz ev ise
7. *) echo "biz sadece arac ve ev icin yonlendirebiliyoruz";; # diger durumlar
8. esac
9.
```