/etc/shells : sistemde bulunan kabukları listeler.

**help** : built-in komutları listeler.

#!/bin/bash : script'in hangi shell tarafından çalıştırılacağını OS'e söyler.

.sh : script dosyasının .sh uzantılı olmasının bir önemi yoktur.

chmod u+x : script'i çalıştırılabilir moda sokar.

${..} : tanımlanan variable'ın bu şekilde kullanılması gerekir.

export : ile tanımlanan variable, child-process'den de görülür.

-default : VARIABLE=${USER\_VAR: -default} eğer user\_var tanımlı değilse default kullanılır.

$1, $2 : console'da tanımlanan parametreler, script'e bu şekilde geçer. ./calc.sh 12 65 (Burada $1=12, $2=65 'dir)

**source** : çalıştırılan bir başka child-betik (sub-script), bu şekilde çalıştırılırsa, onun için ayrı bir process başlatılmaz, böylece onun export variable'larına erişilebilir.

**eval** : bir sub-script programın stdout çıktısını sanki bir komutmuş gibi okuyup, işler:

**read** : bir sub-script 'in çıktısını okuyup bir değere atar. read SYSLOG\_IP <<< $(./sub\_script.sh)

**set** : kabuğun opsiyonlarını ayarlamak için kullanılır.

-e : hata modu aktif, +e: hata modu pasif, yani hata olsa bile script çalışmaya devam eder.

-u : environment checking aktif, environment bulunmazsa, script çıkış verir.

-x : debug modu aktiftir

-f : bash'ın özel karakter kullanmasını engeller.

pipeline (I): Bash, pipe ile yapılan bütün işlemler için ayrı bir process çalıştırır.

Process Substitution: Bir komutun çıktısı, başka bir komuta girdi olarak verilir. Böylece bütün komutlar ayrı process'de çalıştırılmış olur.

VAR1= '...' : VAR1= 'ls -la | grep ferat'

VAR2= $(...) : VAR2= $(ls -la | grep ferat) Programın ekran çıktısını alıp bir değere atıyor.

read VAR3 <<< $(./sub\_script.sh)

Hata: scriptte komutlardan biri hata verse bile işlemeye devam eder. set -e kullanılırsa durur.

/bin/false : program existing with error code "1".

**until**: komut success (0) olana kadar belli aralıklarla çalıştırılır, ne zaman success döner, o zaman devam eder.

& : Bir komutu arka planda çalıştırır.

$? : bir önceki komutun sonucunun success(0), failure (1) sonucunu döner.

$! : bir önceki komutun process id'sini döner.

**wait** : çalıştırılan sub-script'lerin bitmesi beklenir. child-process'ler parent'a signal gönderir ve wait geçilir.

kill 0 <pid> : sıfır sinyali yoktur, sadece bir process'in çalışıp çalışmadığına bakılır.

**(....)** : Bir grup komut sub-shell'de ayrı bir process'de koşturulur

**{....}** : aynı shell'de koşturulur, yeni bir process açılmaz.

/bin/false || { ... } komut hata verirse, aynı shell'de başka bir komut koşturulur.