----------Shell Execution ----------sf4

bash my_script koşturmak için bu komut kullanılabilir.

----------Numeric Variables ----------sf7

bash my_script koşturmak için bu komut kullanılabilir.

var+=1 değişkene 1 ekler

var++ artırıyor

var2=1+$var var değişkenine 1 ekleyip var 2 atıyor.

---------String Variblaes ---------- sf 8

farklı bir türde declare etmedikçe tüm değerler stringdir.

İki parantez içerisine koyarsak integer olarak işlem yapabilir.

((var2=1+$var)) gibi.

---------String Variblaes ---------- sf 9

${string:5} ilk 5 karakteri atla sonrasını al demek.

${string:(-2)} son 2 karakteri alıyor.

${string:2:10} 1. rakam, konumu, 2. rakam o komuttan sonra kaç karakter geleceği.

${#string} string karakterin boyutunu veriyor. length.

ARRAY VARIABLES SF10----------

Referans işlemi ${name[index]}

${a[3]} 4. elemana karşılık gelir.

declare -a ile tanımlama yapılır.

declare -a sports boş dizi tanımladı

sports=(basketbol futbol tenis kayak) içine verileri attı.

echo ${array[*]} tüm dizi içeriğini bastırır.

-------Export Variables ----- sf12

export edince her tarafa erişim açabiliyoruz.

----------Komut satırı argümanları sf13

Bir scriptte argümanlara geçtiğimiz zaman $1 $2 $3 ----- $9 a kadar direk erişim sağlayabiliyoruz.

$0 scriptin ismidir.

$* argümanları bir string şeklinde döndürür.

$@ benzer şekilde argümanarı döndürüyor ama bir dizi şeklinde dönderiyor.

$#

---------- sf 14

echo -n "yes/no? " soru yolluyor kulanıcıya

tek tırnak ile yazarsa olduğu gibi basar yorumlamaz.

echo " 'date +%D' "

----------Shell Variables sf 17

echo "isim gir"

read name

kullanıcının girdinisin name değişkenine alıyor.

değişken tanımlarken içeriği boşluk bırakarak atayamazsın.

mesela FRUIT= apple orange plum hatalı.

FRUIT= "apple orange plum" doğru.

cat read_name

echo Please enter your name"

read name

echoWelcome to CE dept. KTU, $name"

----------Computation on Variables sf21

echo 'expr $a + $b' toplama işlemi yaouyor

expr 5 + 7 çıktı 12 olur

sf 23----------

$1 - $9 pozisyonel parametrelere direk erişim sağlıyor.

$0 scriptin adıdır.

$# argüman sayısını veriyor.

$? koşan son komutun çıktısını vermek için kullanılır.

$$ PROCESSİN ID numarasını veriyor.(bash olarak koştuğumuz text dosyası bile bir process)

$! bu arka planda koşan en son komuta ait process ID yi veriyor.

$@ argümanları dizi şeklinde alıyordu (bash emre 10 20 30 bu parametreleri basıyor)

$@@ ise $* gibi işlem yapar.

egrep ile reguler ifadeleri kullanıyoruz.

sf25----------

ps -aux tüm processleri gösterir. Bash text adı olarak koştuğumuz processi burada görebiliriz.

(eğer sleep var ise.)

sleep 50 & echo 'hi there' sleep program arka planda koşmaya devam ediyor aslında ve durdurmuyor programı. ÖNEMLI!

sf 28----------

Eğer 9 parametreden fazla parametre script'e aktarılırsa, 9 un üzerindeki parametrelere DOĞRUDAN erişim OLAMAZ. bunun sebebi $dan sonra yalnızca 1 tane rakam kabul etmesidir.

shift komutu 1 pozisyon sağa kaymasını sağlar. bu sayede 2 rakamlı parametrelere de erişebiliriz.

while [$# -gt 0] parametre sıfırdan büyük ise while işlemine giriyor

sf 31----------

&& bir önceki komut düzgün bir şekilde çalışmış ise sonraki komutu icra eder.

|| sol taraf başarısız olur ise sonraki komutu koşturur.

sf 34----------

$? koşan son komutun çıktı durumunu öğrenmek için, yani yalnızca okunabilir değişken olan $? değişkeni içerisinde tutulur.

Eğer $? ın değeri 0 ise başarılı çalıştı demektir. farklı birşey ise başarılı çalışamadı demektir.

sf 35----------

if kontrollerinde;

integerlar ile birlikte ((condition))

stringler ile birlikte [[condition.]]

string karşılaştırılmasında tek = kullanılır [[emre = emre]] gibi.

[[-e $file]] if içerisinde bi dosya var mı yok mu onu kontrol ediyor.(SINAVDA ÇIKACAK %100)

[[-f $file]] düzenli dosya olup olmadığını kontrol ediyor. (SINAVDA ÇIKACAK %100)

[[-d $file]] bir dizin olup olmadığını kontrol ediyor.

[[-L $file]] sembolik link mi değil mi?

[[-r $file]] OKUNABİLİR Mİ?

[[-w $file]] yazılabilir Mİ?

[[-p $file]] pipe Mİ?

test komutu herhangi bir çıktı üretmez ama test komutunun çıktı durumu if yapısı şelindeki bir kontrole başarılı olup olmadığını aktarabilir.

echo $? ile kontrol sağlanıyor yani.

-z bir string boş mu değil mi? zerodan geliyor

-n de yine string uzunluğunu kontrol ediyor, sıfır mı değil mi length?

name ="Ahmad"

test -z $name 1 yani false değeri dönecek

test -n $name 0 döner

sf 47---------- integer test komutları

-f REGULER(DÜZENLİ) dosya var mı yok mu testi için kullanılır.

-s ile dosya var mı yokmu kontrol ediyor, birde boyutu 0 mı değil bi

-d dizin mi diye kontrol eder.

-r okuma izni var mı

-w yazma izni var mı

-x çalıştırılabilir izni var mı

ÖDEV: 7.SLAYT SF 14 DEN LOGİN SCRİPTE BAKABİLİRİZ.

SINAV FOR BİTMİŞ WHİLE DAHİL, UNTIL LOOPS 8. CHAPTER SF 59 SON.

ÖDEV:DFS DEEP FİRST SEARCH algoritması ödev olarak verecek ileride!