SHELL programazioa

Sistema Eragileen Oinarriak

Ingurune aldagaiak

Aurredefinitutako aldagaiak

$* parametro guztiak
$? azken komandoaren emaitza
$i i-garren parametroa
$# parametro kopurua
$$ prozesuaren identifikadorea
$0 script-aren izena ($zero)

Posizioaren araberako aldagaiak

$0, **$1**, **$2**, ..., **$9**

$0 script-aren izena, $1 lehen parametroa, $2 bigarrena, ...

shift (10 parametro baino gehiago badaude)

Aldagai lokalak

aldagaia=balioa (balioa testu katea da. ***EZ da zenbakia!***)
unset aldagaia (aldagaia ezabatzen du)

Adibideak

mezua=“Kaixo zer moduz” → “Kaixo zer moduz” katea
kont=100 → “100” katea
mezua=Kaixo_zer_moduz → “Kaixo_zer_moduz” katea

Komandoak

- ***echo*** argumentu bezala pasatako kateak irteera estandarrean idazten ditu

  Adibideak:

  ```
  echo kaixo          # “kaixo” idazten du
  echo $x             # x aldagaiaren balioa idazten du
  ```

  Sekuentzia bereziak (-e adierazlearekin):

\c	Ez du lerro jauzirik idazten
\n	Lerro jauzia idazten du
\f	Orri jauzia idazten du
\t	Tabulazioa idazten du
\r	*Return* idazten du (lerroaren hasierara bueltatuz)

- ***read*** balioak teklatutik irakurtzea ahalbidetzen du, argumentu bezala pasatako aldagaiari esleituz

  Adibidea:

  ```
  echo –e Sar ezazu prezioa: ’\c’
  read prezioa
  ```

Komandoak

- ***expr*** Oinarrizko eragiketa aritmetikoak egiteko erabiltzen da, emaitza irteera estandarrean idazten duelarik

  Honako eragiketak onartzen ditu:

+ Batuketa	adibidea: expr 3 + 2
- Kenketa	adibidea: expr 5 - 2
***** Biderketa	adibidea: expr 6 * 4
/ Zatiketa osoa	adibidea: expr 18 / 7
% Zatiketaren hondarra	adibidea: expr 18 % 7

‘*’ karakterea berezia denez (metakarakterea edo komodina), bere aurretik ‘\’ karakterea idatzi behar da karaktere berezi bezala ulertua izan ez dadin

Eragiketen lehentasuna: %, *, /, +, -

Adibideak:

```
expr 3 + 4              # irteera estandarrean emaitza idatziko da (7)
x=`expr 3 + 4`          # azentuak beharrezkoak dira eragiketa burutzeko
expr `expr $x + $y` / `expr $a + $b`
```

Komandoak

- ***test*** Baldintza-espresioak: Argumentuak aldagaiak, konstanteak, testu kateak edota fitxategiak / katalogoak izan daitezke.

Sintaxia: test arg1 -opzioa arg2 # test ordez ‘[…]‘ erabil daiteke

Opzioak:

Zenbakiak	
-eq	berdina
-ne	desberdina
-gt	handiagoa
-ge	handiagoa / berdina
-lt	txikiagoa
-le	txikiagoa / berdina

Testu kateak	
=	berdina
!=	desberdina
-z	kate nulua
-n	kate ez nulua

Fitxategi / Katalogoak	
-a	existitzen da
-s	exist. eta ez hutsik
-f	fitxategi arrunta
-d	katalogoa
-r	irakur. baimena
-w	idazk. baimena
-x	exekut. baimena

Baldintza-espresioak konbina daitezke operadore logikoen bitartez:

-a AND (“eta” logikoa)

-o OR (“edo” logikoa)

Baldintza agindua: *if*

- **if** Baldintzazko egitura

Sintaxia:

a) **if** baldintza
then
aginduak
fi

b) **if** baldintza
then
aginduak
else
aginduak
fi

baldintza: programa / komandoa da
TRUE emaitza 0 denean
FALSE emaitza 0 ez denean

c) **if** baldintza1
then
aginduak
elif baldintza2
then
aginduak
fi

Baldintza agindua: ***if***

Adibideak:

```
if test $n -gt 10
then
        echo “$n zenbakia 10 baino handiagoa da”
fi
```

```
if [ $# -lt 1 ]                 # errorea
then
        echo “errorea: bi argumentu gutxienez behar dira”
        exit 1
else                            # zuzena
        ….
fi
```

Baldintza agindua: ***case***

- **case** Aukera anitzeko sekuentzia

  Sintaxia:

  a) **case** espresioa **in**

  patroia1**)** aginduak ;;

  patroia2**)** aginduak ;;

  patroia3**)** aginduak ;;

  .....

  ***)** aginduak

  esac

Baldintza agindua: ***case***

Adibidea:

```
read ald1
case $ald1 in
        A) echo “A aukeratu duzu” ;;
        B) echo “B aukeratu duzu” ;;
        C) echo “C aukeratu duzu” ;;
        ......
        *) echo “Baliogabeko aukera”
esac
```

Iterazio agindua: ***while***

- **while** Baldintza betetzen den bitartean errepikatuko den agindu multzoa

Sintaxia:

while baldintza
do
aginduak
done

baldintza: programa da
TRUE emaitza 0 denean
FALSE emaitza 0 ez denean

Iterazio agindua: ***until***

- **until** Baldintza bete arte errepikatuko den agindu multzoa

  Sintaxia:

```
until baldintza
do
        aginduak
done
```

baldintza: programa da
TRUE emaitza 0 denean
FALSE emaitza 0 ez denean

Iterazio agindua: ***for***

- **for** Pasatako zerrendako balio bakoitzarentzat errepikatuko den agindu multzoa

  Sintaxia:

  for aldagaia **in** balio_zerrenda
  do
  aginduak
  done

Iterazio agindua: ***for***

Adibideak:

```
for i in $*
do
        echo “===== $i argumentua =====”
done
```

```
for i in Ane Koldo Miren
do
        echo Kaixo $i
done
```

```
for zenb in 1 2 3 4 5 6 7 8 9 10
do
        echo “Katea:” $zenb
done
```

Adibideak

1.- *script*-aren izena idaztea

```
echo $0
```

2.- Konektaturik dagoen erabiltzaile kopurua ematen duen komandoa

```
echo “erabiltzaile kopurua:” `who | wc -l`
```

3.- Erabiltzaile bat konektaturik dagoen ala ez egiaztatzen duen *script*-a. Erabiltzailea argumentu bezala pasatzen zaio

```
if  who | grep “$1” >/dev/null
then
      echo “$1 konektaturik dago”
fi
```

Adibideak

4.- Komandoaren oihartzuna

```
echo $0
for i in $*
do
        echo $i
done
```

5.- “.h” bukaera duten fitxategien lehen 10 lerroak listatu, aurretik buru bat jarriz

```
for i in *.h
do
        echo “===== $i fitxategia =====“
        head $i
done
```

Adibideak

6.- Aurreko ariketa bezala, baina bukaera(k) parametro gisa lortuz eta fitxategi arruntak direla egiaztatuz. Gutxienez parametro bat pasatzen zaiola ere egiaztatzen du

```
if [ $# -lt 1 ]
then
    echo "errorea: ez da aurkitu parametrorik"
    exit 1
fi
for mota in $*
do
    for i in *.$mota
    do
        if [ -f $i ]
        then
            echo "===== $i fitxategia ====="
            head $i
        fi
    done
done
```

Adibideak

7.- Zenbaki bat irakurri, eta 10 baino handiagoa bada mezu bat pantailaratu eta begizta bat egin 10-eraino

```
echo –e "Sartu zenbaki bat:\c"
read n
if test $n -gt 10  #baita: if [ $n –gt 10 ]
then
    echo "10 baino handiagoa da"
    while [ $n –ge 10 ]
    do
        echo $n
        n=`expr $n – 1`
    done
fi
```

Adibideak

8.- Parametro gisa pasatako fitxategien azken 10 lerroak listatu, aurretik buru bat jarriz. Parametroak daudela egiaztatzen da, baita fitxategien existentzia ere

```
if [ $# -lt 1 ]
then
    echo "errorea: parametroak behar dira"
    exit 1
fi
for i in $*
do
    if [ -f $i ]
    then
        echo "===== $i fitxategia ====="
        tail $i
    fi
done
```

Adibideak

9.- Uneko katalogoa eta lehen mailako azpikatalogoak aztertu, bertan dauden fitxategi zein katalogoen izenak pantailaratuz

```
for i in *
do
    if [ -d $i ]
    then
        echo "Katalogoa: $i/"
        for j in $i/*
        do
            if [ -d $j ]
            then
                echo "   Azpi_katalogoa: $j/"
            fi
            if [ -f $j ]
            then
                echo "   Azpi_fitxategia: $j"
            fi
        done
    fi
    if [ -f $i ]
    then
        echo "Fitxategia: $i"
    fi
done
```