دانشگاه صنعتی امیرکبیر

آزمایشگاه سیستم‌عامل

جلسه سوم: آشنایی با دستور نویسی در سیستم‌عامل

مدرس :مینا یوسف‌نژاد

Bash = shell = command language interpreter

راه‌های اجرای دستورات در bash:

- مستقیم
- نوشتن در فایل اجرایی:

/#!bin/bash

Stream

Stream	توضیحات	مثال
Standard in (stdin)	ورودی ها: به طور پیش فرض به کیبورد متصل است	input => python scanf => c cin => c++ read=>bash
Standard out (stdout)	خروجی: به طور پیش فرض به نمایشگر یا ترمینال متصل است	print=>python printf=>c cout=>c++ echo=>bash
Standard error (stderr)	به طور پیش stdoutمانند فرض به ترمینال یا نمایشگر متصل است ولی مستقل مدیریت میشود.	sys.stderr=>python fprintf=>c std::cerr=>c++ >&2 =>bash
stdio => c iostream => c++		

Redirection & piping

- Process > data file: این دستور خروجی یک فرآیند را به یک فایل داده هدایت می‌کند. اگر فایل وجود نداشته باشد، آن را ایجاد می‌کند و اگر وجود داشته باشد، محتوای آن را بازنویسی می‌کند.
- Process >> data file: این دستور خروجی یک فرآیند را به یک فایل داده هدایت می‌کند. اگر فایل وجود نداشته باشد، آن را ایجاد می‌کند و اگر وجود داشته باشد، خروجی را به محتوای موجود آن اضافه می‌کند.
- Process < data file: این دستور محتویات یک فایل داده را می‌خواند و آن را به ورودی یک فرآیند هدایت می‌کند.
- Piping: شما می‌توانید از piping نیز استفاده کنید تا خروجی یک فرآیند به ورودی فرآیند دیگر هدایت شود.

دریافت ورودی از کاربر

```
read -p 'Username: ' uservar
read -sp 'Password: ' passvar
```

عبارت شرطی

```
if []; then
   command1
elif []; then
   command2
else
   command3
fi
```

عبارت چند حالته:

```
case $variable in
   pattern-1)
        commands
        ;;
   pattern-2)
        commands
        ;;
   pattern-3|pattern-4|pattern-5)
        commands
        ;;
   pattern-N)
        commands
        ;;case $variable in
   pattern-1)
        commands
        ;;
   pattern-2)
        commands
        ;;
   pattern-3|pattern-4|pattern-5)
        commands
        ;;
   pattern-N)
        commands
        ;;
   *)
        commands
        ;;
esac
```

حلقه:

```
while [ condition ]
do
   command1
   command2
   command3
done
```

```
#! /bin/bash

counter=0
while [ $counter -lt 10 ]
do
   echo "The counter is $counter"
   let counter=counter+1
done
```

```
for VARIABLE in 1 2 3 4 5 .. N
do
    command1
    command2
    command3
done

for VARIABLE in file1 file2 file3 .. fileN
do
    command1
    command2
    command3
done



for OUTPUT $(Linux-or-Unix-Command-Here)
do
    command1
    command2
    command3
done

for i in $(ls)
do
    echo item: $i
done
```

توابع:

```
function function_name(){
  command1
  command2
  command3
  return 1
}
```

```
function greeting(){
   echo hello $1
   return 2
}

greeting john
echo $?
```

متغیرهای خاص یا سیستمی

$USER	این متغیر نام کاربر فعلی که وارد سیستم شده است را نگه‌داری می‌کند.	برای شخصی‌سازی پیام‌ها یا دستورات بر اساس کاربر جاری.
$0	نام اسکریپت یا برنامه‌ای که در حال حاضر در حال اجرا است را نشان می‌دهد. اگر در ترمینال دستور داده شود، نام دستور را نشان می‌دهد.	برای نمایش نام خود اسکریپت یا برای انجام عملیات بر اساس نام اسکریپت.
$1, $2, ..., $9	این متغیرها به ترتیب مقادیر آرگومان‌های ورودی به اسکریپت را نگه‌داری می‌کنند. به عنوان مثال، $1 اولین آرگومان، $2 دومین آرگومان و ... است.	برای دریافت و پردازش ورودی‌های داده شده به اسکریپت.
$#	این متغیر تعداد آرگومان‌های ورودی به اسکریپت را نشان می‌دهد.	برای بررسی تعداد آرگومان‌های ورودی و مدیریت ورودی‌های نادرست.
$$	فرایند جاری را نشان می‌دهد. هر (PIDاین متغیر شناسه) با یک شناسه فرایند منحصر به Bashاسکریپت یا دستور در فرد اجرا می‌شود.	برای انجام عملیات‌هایی مانند ایجاد فایل‌های موقتی با استفاده از PID.
$@	این متغیر تمام آرگومان‌های ورودی را به عنوان یک لیست جداگانه نشان می‌دهد. اگر آن را در یک حلقه استفاده کنید، هر آرگومان به طور جداگانه پردازش می‌شود.	برای دسترسی به تمام آرگومان‌های ورودی به راحتی.

آزمایش 1

برنامه‌ای بنویسید که از کاربر نام و سن او را بپرسد و بر اساس سن، یک پیام چاپ کند.

- از دستور `read` برای دریافت ورودی‌ها استفاده کنید.
- اگر سن کمتر از ۱۸ باشد، پیام "شما نوجوان (teenager)هستید." چاپ شود.
- اگر سن بین ۱۸ تا ۳۰ باشد، پیام "شما جوان (young)هستید." چاپ شود.
- اگر سن بالاتر از ۳۰ باشد، پیام "شما بزرگسال (adult) هستید." چاپ شود.

آزمایش 2

برنامه‌ای بنویسید که یک عدد از کاربر بگیرد و بررسی کند که آیا آن عدد زوج است یا فرد.

- از ``if-else`` برای بررسی زوج یا فرد بودن عدد استفاده کنید.

آزمایش 3

برنامه‌ای بنویسید که تا زمانی که کاربر عددی کمتر از ۱۰۰ وارد نکرده است، از او اعداد دریافت کند.

- با استفاده از ``while`` این کار را پیاده‌سازی کنید.
- اگر عدد وارد شده کمتر از ۱۰۰ بود، برنامه پایان یابد.