```
----Lecture 06

Importance of playing

--command line color
bash Bourne again shell
bashrc (resource control)
Talk about .bashrc and how you can make it color of the prompt change
.bash_aliases

--echo command print to terminal and move to next line
echo “Hello World”
--variables
test=”Hello World”
echo $test
what happens if I type
echo test

read puts into a variable
read yourname  # no space
# for comments

-- Exercise 6.1: read a name and echo it
read yourname
echo $yourname

-- How to do I prompt for  my name and read it and type it?
echo "what is your name?"; read yourname ; echo "you typed:"$yourname

This is lame!?%@@E
-- make sh file with the following contents
echo "what is your name?"
read yourname
echo "you typed:"$yourname

execute it
bash sth.sh
bash -xu sth.sh  print traces of execution.

shebang
#! /bin/bash
# print your name

--- Exercise6.2:
Write a script to ask your name and your family name and prints both.

-- further use of echo
We use pgrep to find process id number and then look for limits can you do both together
 $ pgrep chrome
 $cat /proc/3158/limits
Make a guess
$cat /proc/$(pgrep chrome)/limits
Would {} or [] work instead of ()

--  use of grave accent (push twice)
another way is to use ` ` as it expands comi
mands
$ cat /proc/`pgrep chrome`/limits
similarly
 2046  echo `pwd`
 2047  echo $(pwd)


--  difference between “” '' without
obviously cant use ! Or ;  They have specail meanings
echo Another Brick In The Wall
echo "Another Brick In The Wall"
echo 'Another Brick In The Wall'
What is I add ! Or ;
it can think it s a command use “” and ''

-- Double quote allows expansion of variables and commands and single quote does not
```

```
Try
 test="Hello World"
 echo test
 echo $test
--but ' ' does not open
 echo '$test'

Double Quotes
Use when you want to enclose variables or use shell expansion inside a string.
All characters within are interpreted as regular characters except for $ or ` which will be expanded
on the shell.
Single Quotes
All characters within single quotes are interpreted as a string character.


--  ` is something different does not
 echo `hello world I am here`



-- Integer arithmetic via expr
 icy=53
 others=23
 echo `expr $icy + $others `
 echo `expr $icy - $others `
 echo `expr $icy \* $others `

echo `expr $icy / $others `
echo `expr $icy % $others `


--Excersie 6.3:
Modify the program to read the icy and others from the command line and print the values properly

-- Exercise 6.4
Write a program to ask for date of birth of people and calculate how many days they lived. Year=365
days Months are 30 days and todays date is hardcoded.

– Shell awareness
Change the spaces and see what happens. Be careful with spaces.
Paranthesis are with \( \)

----- you can also do $((
icy=23
others=24
echo $((icy + others))
echo $((icy+others))
echo $((icy*others))

-- Exercise 6.5: write the above arithmetic with $((

---For floating point arithmetic
you need to use other methods  bc, we will see that later
priceofapple=1.2
priceoforange=2.5
c=`echo $priceofapple + $priceoforange |bc`
echo $c

-- tables
Speaking of the price I like to make s shopping list
$ echo -e "no \t item \t price \n 1 \t orange \t 1.1 \n 2 \t apple \t 1.50 \n 3 \t banana \t 2.1 "

no       item    price
 1       orange          1.1
 2       apple   1.50
 3       banana          2.1

Not really good spaces are not OK let us print it properly

-- use printf
left alignment of text
f for floating point
```

```
printf "my weakly shopping list:\n"
printf "%-5s %-15s %-6s\n" no  item  price
printf "%-5s %-15s %-3.2f\n" 1 apple 1.31
printf "%-5s %-15s %-3.2f\n" 2 orange  1.61
printf "%-5s %-15s %-3.2f\n" 3 banana 2.35
printf "%-5s %-15s %-3.2f\n" 4 sthstrange 43.5446

-- Exercise 6.6:  print your next week diary

-- Exercise 6.7:
1) remind yourself what commands touch and >> mean
2) Write a shel script which
i)  asks for a file name, say for example user types foo
ii) creates a txt file name with the name foo (this would be making a file called foo.txt)
ii) adds a first line of the following form
This file is: foo.txt
```