LING 408/508: Programming for Linguists

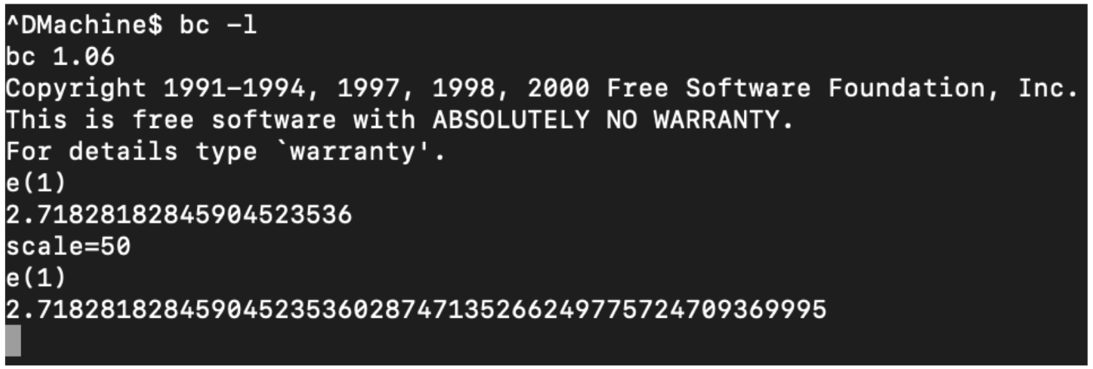
Lecture 7

Today's Topic

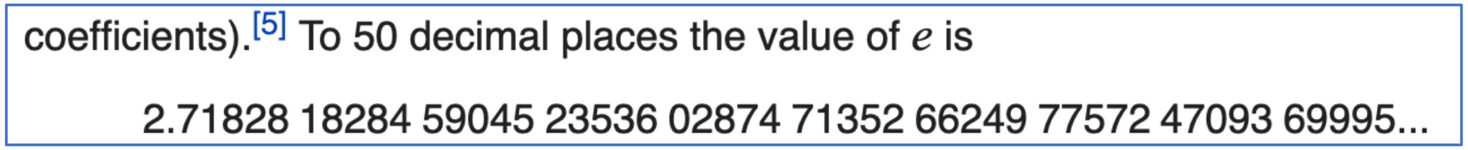
- Homework 3 Review
- asking for user input from the Terminal
- Case Study:
  - Let's write a shell script to calculate BMI
- Ungraded Homework 4

Homework 3 Review

- Question 1
  - use bc to compute the value of the math constant *e* to 50 decimal places



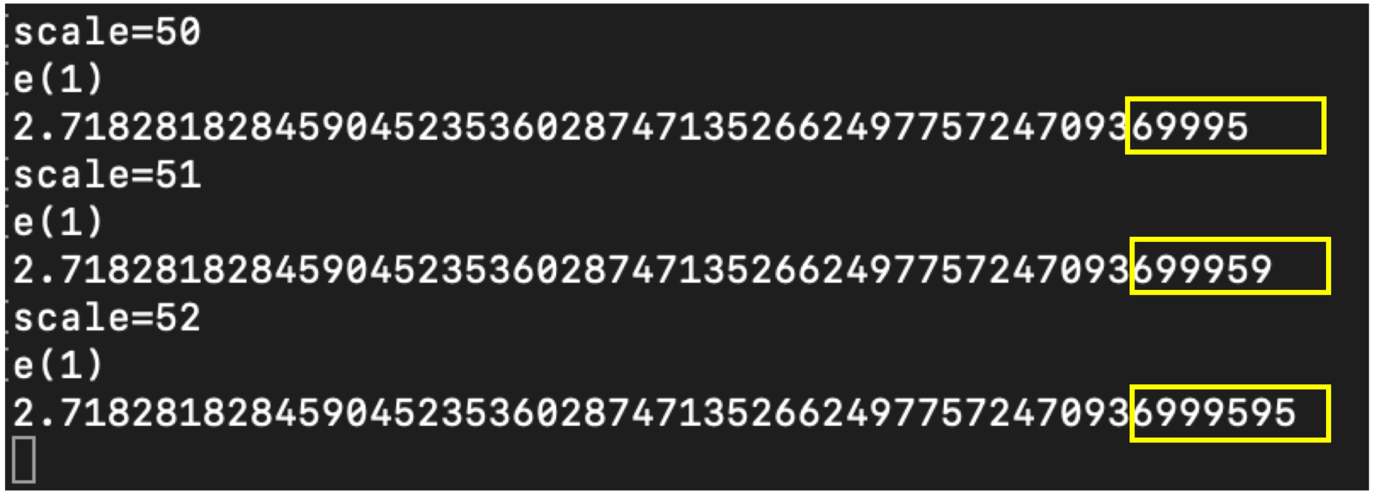
```
^DMachine$ bc -l
bc 1.06
Copyright 1991-1994, 1997, 1998, 2000 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
e(1)
2.71828182845904523536
scale=50
e(1)
2.71828182845904523536028747135266249775724709369995
```



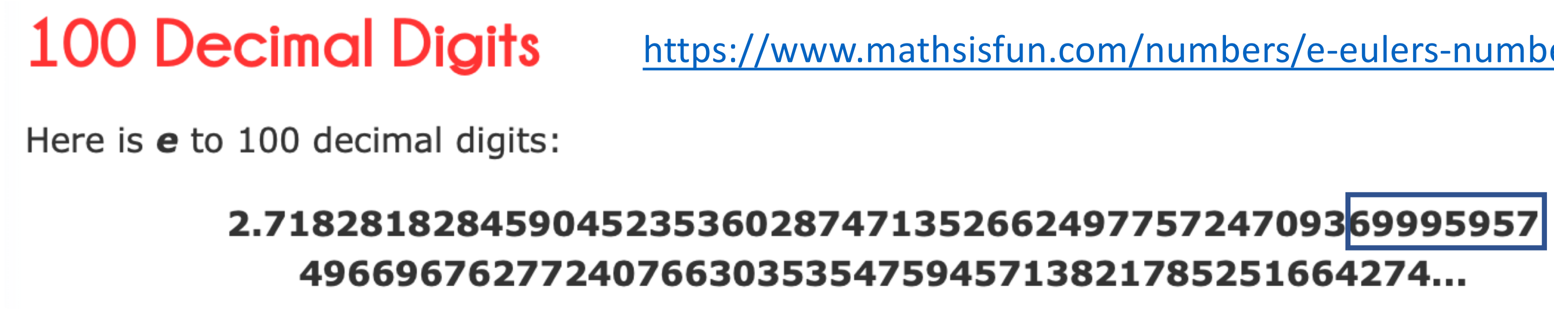
coefficients).[5] To 50 decimal places the value of *e* is

2.71828 18284 59045 23536 02874 71352 66249 77572 47093 69995...

Homework 3 Review



```
scale=50
e(1)
2.71828182845904523536028747135266249775724709369995
scale=51
e(1)
2.718281828459045235360287471352662497757247093699959
scale=52
e(1)
2.7182818284590452353602874713526624977572470936999595
```



100 Decimal Digits

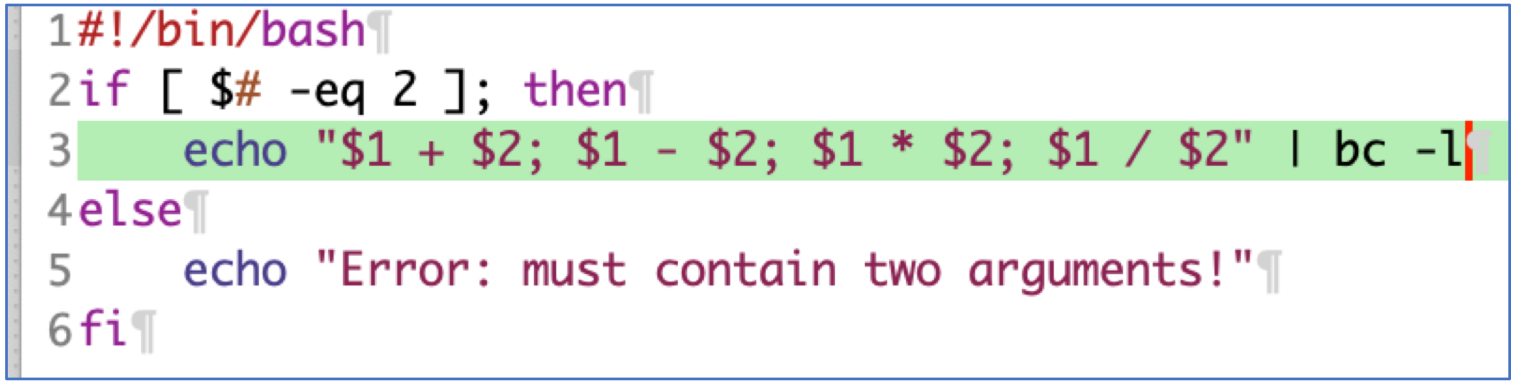
https://www.mathsisfun.com/numbers/e-eulers-number.html

Here is ***e*** to 100 decimal digits:

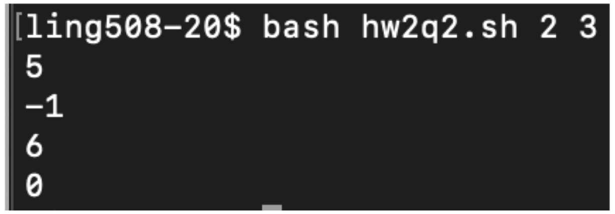
2.71828182845904523536028747135266249775724709369995957
49669676277240766303535475945713821785251664274...

Homework 3 Review

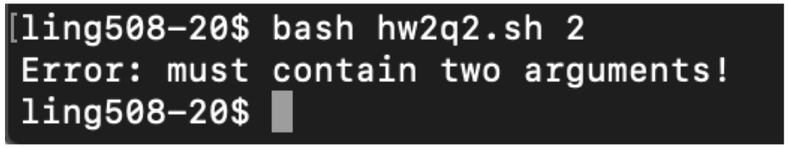
- Question 2:
  - bash script calls bc to print out the result of adding, substracting, multiplying and dividing the two numbers.



```bash
#!/bin/bash
if [ $# -eq 2 ]; then
    echo "$1 + $2; $1 - $2; $1 * $2; $1 / $2" | bc -l
else
    echo "Error: must contain two arguments!"
fi
```



```
[ling508-20$ bash hw2q2.sh 2 3
5
-1
6
0
```



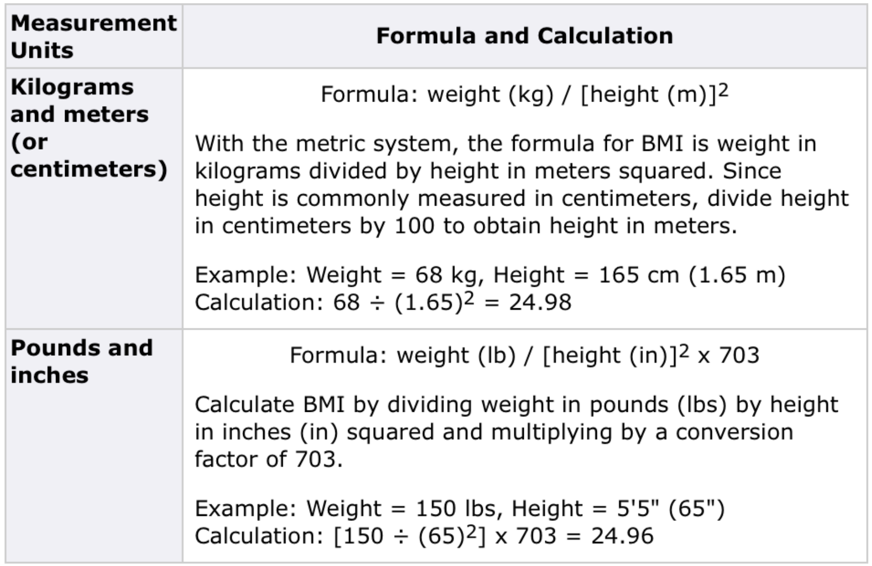
```
[ling508-20$ bash hw2q2.sh 2
Error: must contain two arguments!
ling508-20$
```

Input

- At a terminal:
  - `read –p “Name: ” name`
  - `read –p “Enter X and Y: ” x y`
  - `echo $x`
  - `echo $y`

First Non-trivial Program

- Let’s write a simple shell-script BMI calculator
  - it can solicit input from the terminal or take command line arguments

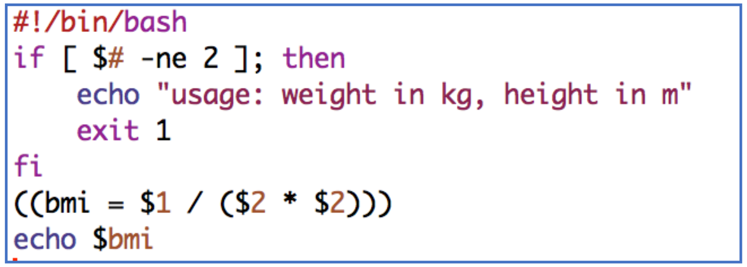


Measurement Units	Formula and Calculation
Kilograms and meters (or centimeters)	Formula: weight (kg) / [height (m)]^2 With the metric system, the formula for BMI is weight in kilograms divided by height in meters squared. Since height is commonly measured in centimeters, divide height in centimeters by 100 to obtain height in meters. Example: Weight = 68 kg, Height = 165 cm (1.65 m) Calculation: 68 ÷ (1.65)^2 = 24.98
Pounds and inches	Formula: weight (lb) / [height (in)]^2 x 703 Calculate BMI by dividing weight in pounds (lbs) by height in inches (in) squared and multiplying by a conversion factor of 703. Example: Weight = 150 lbs, Height = 5'5" (65") Calculation: [150 ÷ (65)^2] x 703 = 24.96

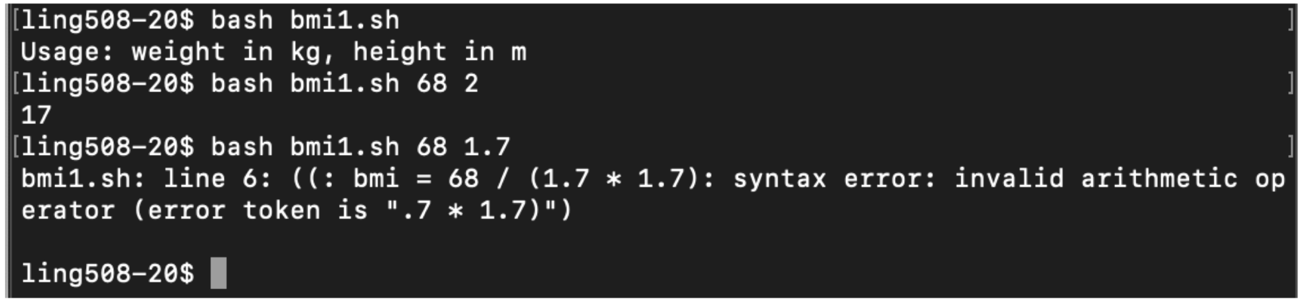
try the metric one first

First Non-trivial Program

- First pass (let's see what happens):



```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in m"
    exit 1
fi
((bmi = $1 / ($2 * $2)))
echo $bmi
```



```
ling508-20$ bash bmi1.sh
Usage: weight in kg, height in m
ling508-20$ bash bmi1.sh 68 2
17
ling508-20$ bash bmi1.sh 68 1.7
bmi1.sh: line 6: ((: bmi = 68 / (1.7 * 1.7): syntax error: invalid arithmetic op
erator (error token is ".7 * 1.7)")

ling508-20$ 
```

BMI calculator

- Did you notice bash can only do integers?
  - can use bc
  - recall scale = # of decimal places
  - `echo "scale=2;2/3" | bc`

  ```
  .66
  ```

  - but test comparisons (`-gt` etc.) would then be a pain in the butt
  - can re-scale the formula:

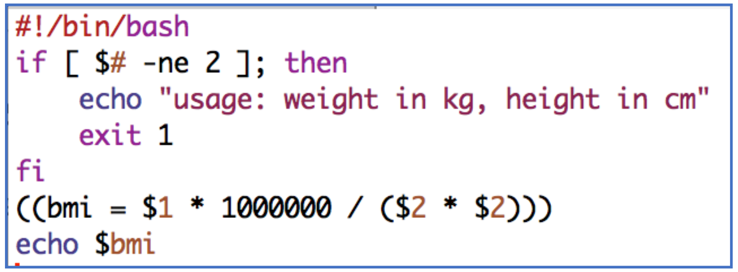
  Example:

  - weight in kg * 1,000,000 / (height in cm)**2
  - `echo $((68* 1000000 / (165 * 165)))`

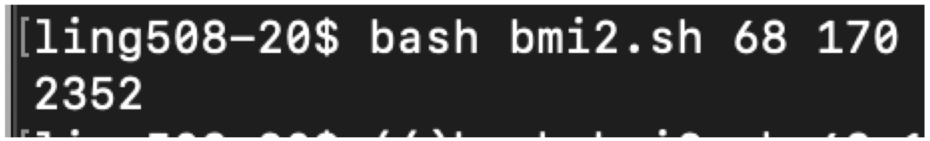
  ```
  2497
  ```
  (24.97)

First Non-trivial Program

- Some other possibilities:
  - scale to cm: 100 cm = 1 meter, so 100 * 100 = 10000
  - suppose we scale again by 100 (to get 2 decimal places), so 10000 * 100 = 1000000



```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in cm"
    exit 1
fi
((bmi = $1 * 1000000 / ($2 * $2)))
echo $bmi
```

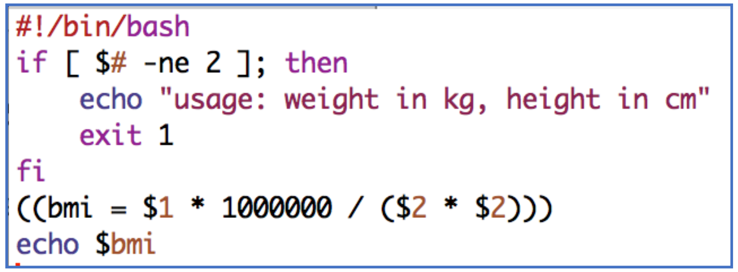


```
[ling508-20$ bash bmi2.sh 68 170
2352
```

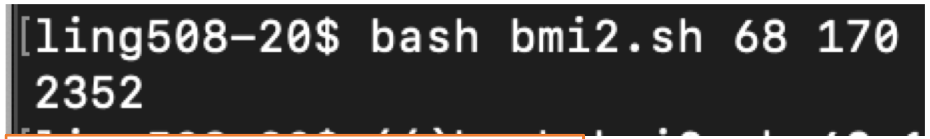
68 kg; 170 cm tall
BMI: 2352 is really 23.52

First Non-trivial Program

- Some other possibilities:

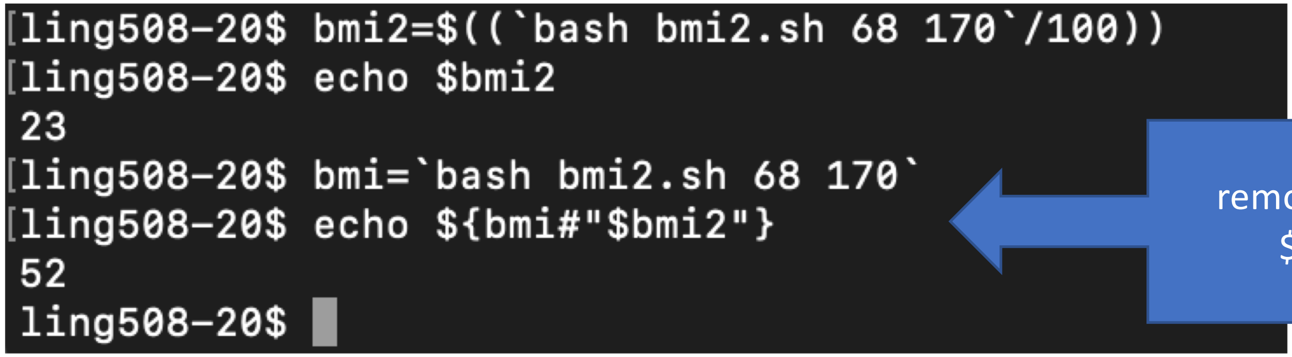


```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in cm"
    exit 1
fi
((bmi = $1 * 1000000 / ($2 * $2)))
echo $bmi
```



```
[ling508-20$ bash bmi2.sh 68 170
2352
```

68 kg; 170 cm tall
BMI: 2352 is really 23.52

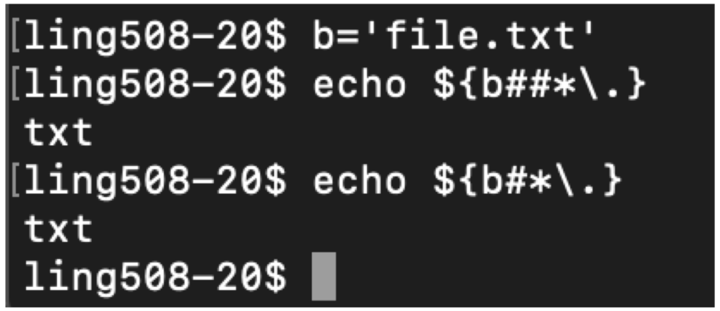


```
[ling508-20$ bmi2=$((`bash bmi2.sh 68 170`/100))
[ling508-20$ echo $bmi2
23
[ling508-20$ bmi=`bash bmi2.sh 68 170`
[ling508-20$ echo ${bmi#"$bmi2"}
52
ling508-20$ 
```

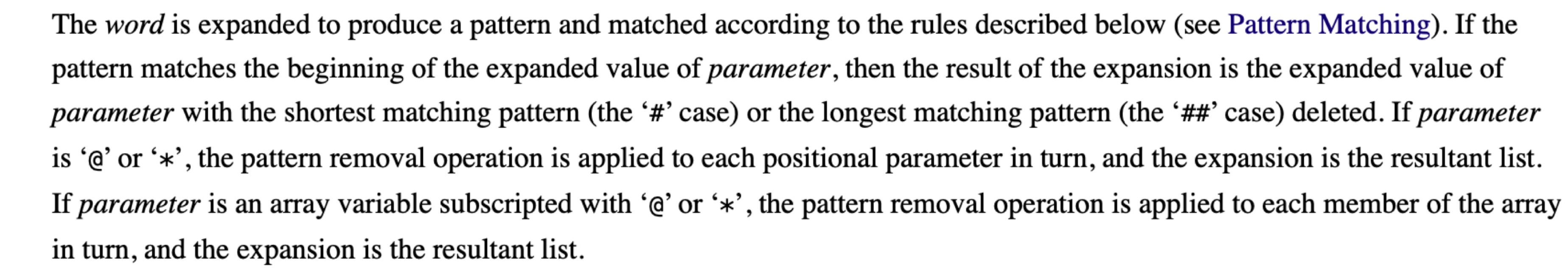
remove prefix
$bmi2

Delete Prefix

https://www.gnu.org/software/bash/manual/html_node/Shell-Parameter-Expansion.html#Shell-Parameter-Expansion



```
[ling508-20$ b='file.txt'
[ling508-20$ echo ${b##*\.}
txt
[ling508-20$ echo ${b#*\.}
txt
ling508-20$ 
```

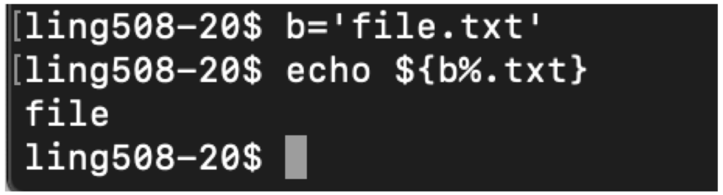


`${parameter#word}`

`${parameter##word}`

The *word* is expanded to produce a pattern and matched according to the rules described below (see Pattern Matching). If the pattern matches the beginning of the expanded value of *parameter*, then the result of the expansion is the expanded value of *parameter* with the shortest matching pattern (the '#' case) or the longest matching pattern (the '##' case) deleted. If *parameter* is '@' or '*', the pattern removal operation is applied to each positional parameter in turn, and the expansion is the resultant list. If *parameter* is an array variable subscripted with '@' or '*', the pattern removal operation is applied to each member of the array in turn, and the expansion is the resultant list.

Delete Suffix



```
[ling508-20$ b='file.txt'
[ling508-20$ echo ${b%.txt}
file
ling508-20$
```

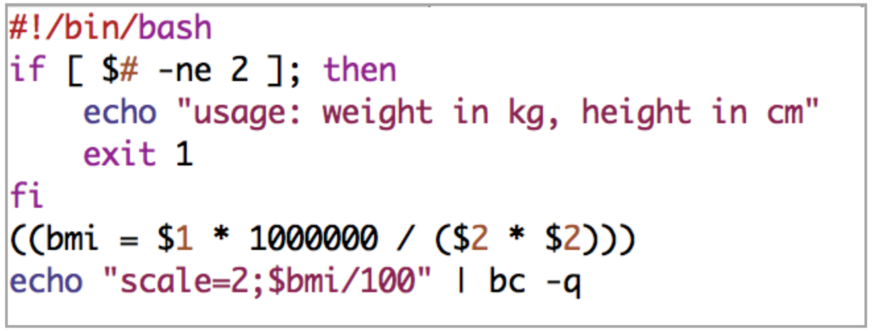
`${`*`parameter`*`%`*`word`*`}`

`${`*`parameter`*`%%`*`word`*`}`

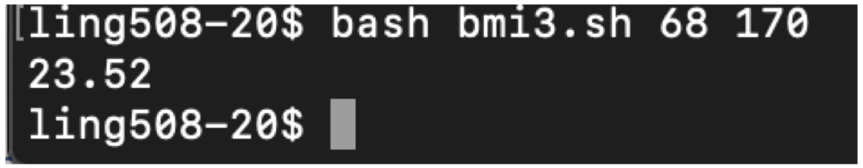
The *word* is expanded to produce a pattern and matched according to the rules described below (see Pattern Matching). If the pattern matches a trailing portion of the expanded value of *parameter*, then the result of the expansion is the value of *parameter* with the shortest matching pattern (the ‘`%`’ case) or the longest matching pattern (the ‘`%%`’ case) deleted. If *parameter* is ‘`@`’ or ‘`*`’, the pattern removal operation is applied to each positional parameter in turn, and the expansion is the resultant list. If *parameter* is an array variable subscripted with ‘`@`’ or ‘`*`’, the pattern removal operation is applied to each member of the array in turn, and the expansion is the resultant list.

First Non-trivial Program

- Some other possibilities:



```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in cm"
    exit 1
fi
((bmi = $1 * 1000000 / ($2 * $2)))
echo "scale=2;$bmi/100" | bc -q
```

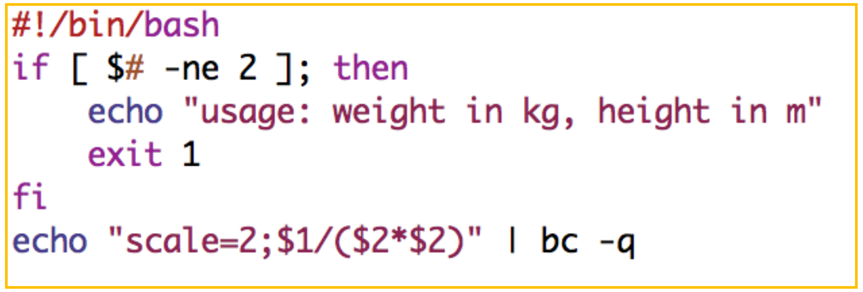


```
ling508-20$ bash bmi3.sh 68 170
23.52
ling508-20$ 
```

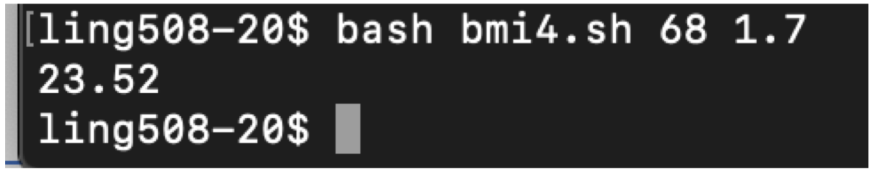
bc –q
means quiet mode
not needed can simply use:
bc

First Non-trivial Program

- Some other possibilities:



```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in m"
    exit 1
fi
echo "scale=2;$1/($2*$2)" | bc -q
```

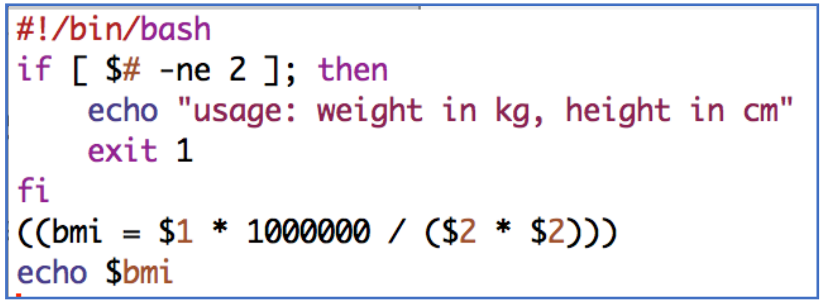


```
[ling508-20$ bash bmi4.sh 68 1.7
23.52
ling508-20$
```

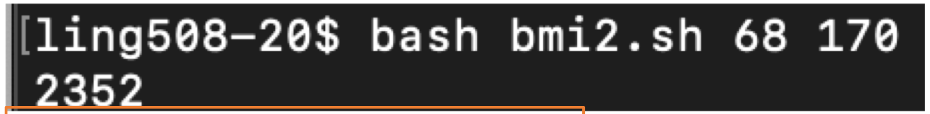
`bc –q`
means quiet mode
not needed can simply use:
`bc`

Upgraded Homework 4

- Modify the shell script so it computes BMI (without using bc) and prints to two decimal places



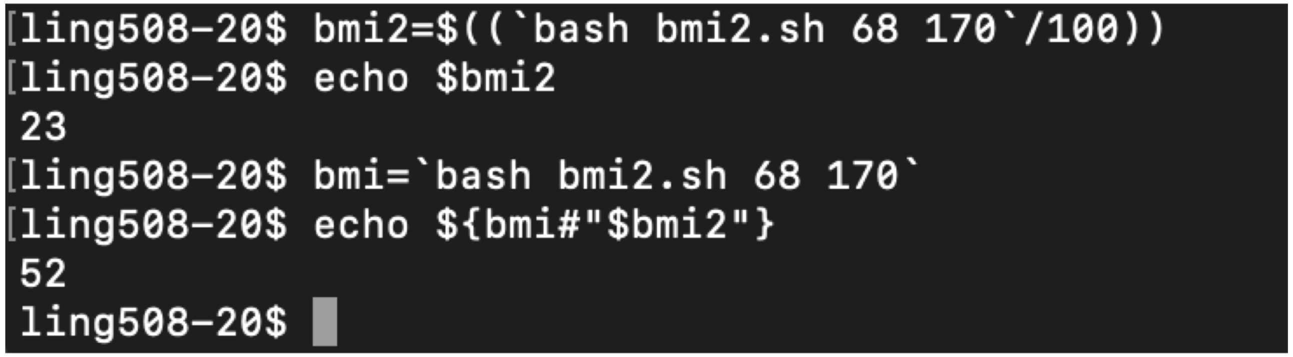
```
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "usage: weight in kg, height in cm"
    exit 1
fi
((bmi = $1 * 1000000 / ($2 * $2)))
echo $bmi
```



```
[ling508-20$ bash bmi2.sh 68 170
2352
```

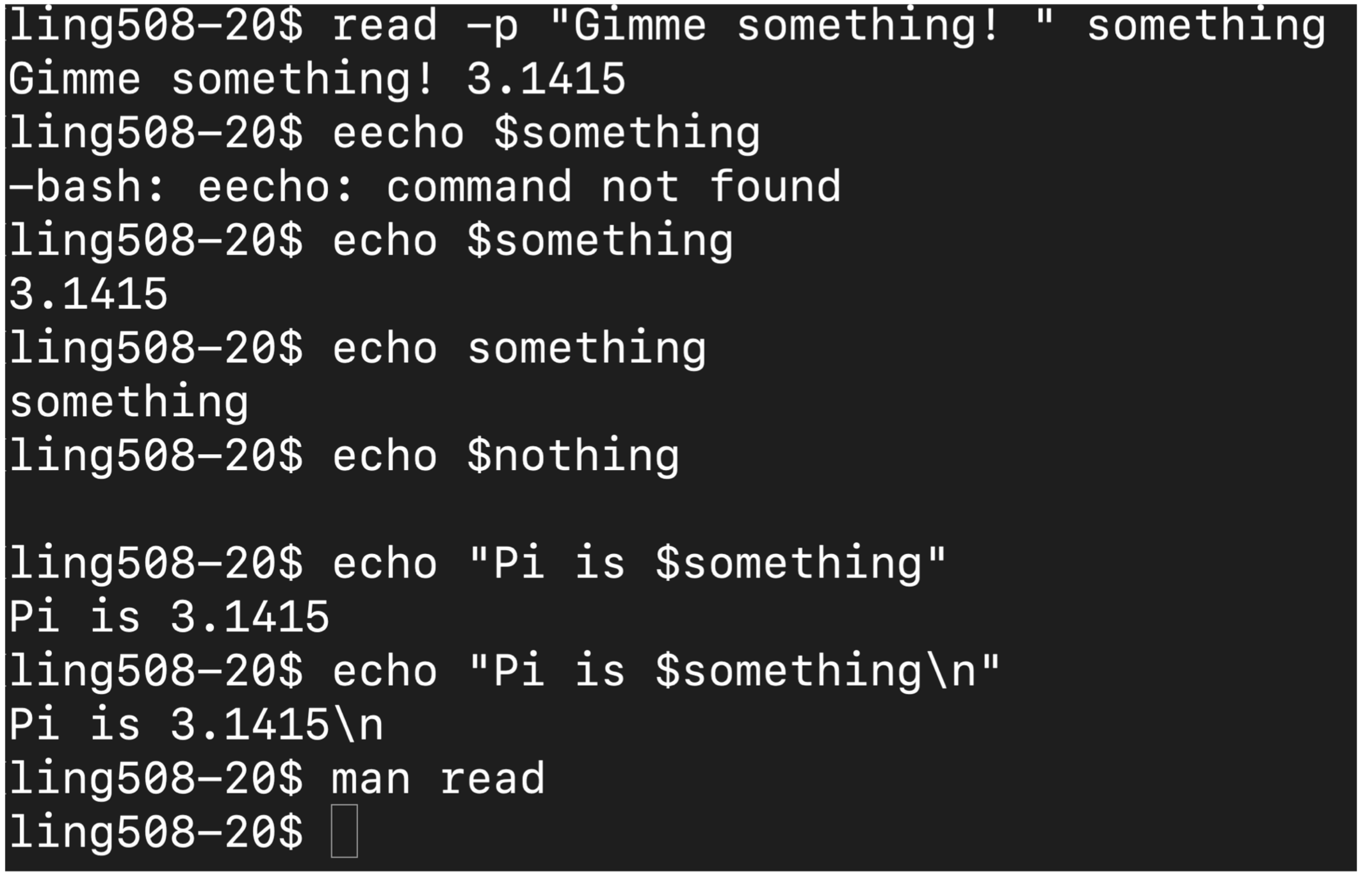
68 kg; 170 cm tall

BMI: 2352 is really 23.52



```
[ling508-20$ bmi2=$((`bash bmi2.sh 68 170`/100))
[ling508-20$ echo $bmi2
23
[ling508-20$ bmi=`bash bmi2.sh 68 170`
[ling508-20$ echo ${bmi#"$bmi2"}
52
ling508-20$ 
```

Appendix: today's Terminal



```
ling508-20$ read -p "Gimme something! " something
Gimme something! 3.1415
ling508-20$ eecho $something
-bash: eecho: command not found
ling508-20$ echo $something
3.1415
ling508-20$ echo something
something
ling508-20$ echo $nothing

ling508-20$ echo "Pi is $something"
Pi is 3.1415
ling508-20$ echo "Pi is $something\n"
Pi is 3.1415\n
ling508-20$ man read
ling508-20$ 
```