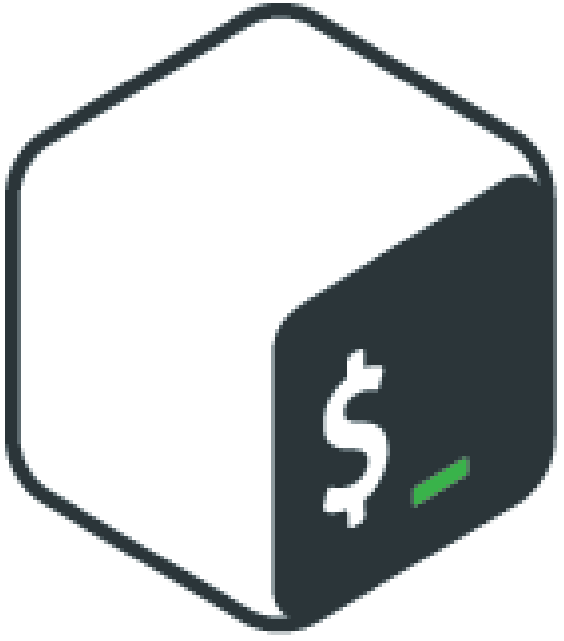
BINF2111 – Introduction to Bioinformatics Computing

BASH 101 – Bash around and find out?



BASH

THE BOURNE-AGAIN SHELL

Richard Allen White III, PhD
RAW Lab

Lecture 8 – Thursday Sep 16th, 2021

Learning Objectives

- Review quiz and bonus

- Bash variables review

- Bash variables strings

- Sting formats bioinfomatics (fasta, fasta-like files)

- Bash Arrays

- Quiz 8

Quiz 7 answers

bash script.sh or ./script.sh does what?

does what?

Quiz 7 answers

bash script.sh or ./script.sh does what?

does what?

runs the bash script in terminal (44% missed)

./script.sh (wait does this mean?)

Quiz 7 answers

printf '#!/bin/bash\n# This is my first comment\
n' >script.sh

Output is:
#!/bin/bash

This is my first comment

T or F

Quiz 7 answers

```
printf '#!/bin/bash\n# This is my first comment\
n' >script.sh
```

Output is:

```
#!/bin/bash

# This is my first comment
```

T or **F**

Quiz 7 answers

```
printf '#!/bin/bash\n# This is my first comment\
n' >script.sh
```

Output is: (no space)

```
more script.sh
#!/bin/bash
# This is my first comment
```

Quiz 7 answers

Output is:

#!/bin/bash

This is my first comment

What is the printf command?

Quiz 7 answers

Output is:

#!/bin/bash

This is my first comment

What is the printf command?

printf '#!/bin/bash\n \n# This is my first comment' >script.sh

Bonus 6

- Write a bash script that states 150 kg at 178 cm is overweight?

ENJOY!

Bonus 6

- Write a bash script that states 150 kg at 178 cm is overweight?

```
1 #!/bin/bash
2
3 # This script prints a message about your weight if you give it
  your
4
5 # weight in kilos and height in centimeters.
6
7 weight="$1"
8 height="$2"
9 idealweight=$[$height - 110]
10 if [ $weight -le $idealweight ] ; then
11  echo "You should eat a bit more food."
12 else
13  echo "You should eat a bit less food."
14 fi
```

bash -x weight.sh 130 178

BASH Reserved words

! - Pipelines
[[ ]] - Conditional Constructs
{ } - Command Grouping
break - Looping Constructs
case - Conditional Constructs
continue - Looping Constructs
do - Looping Constructs
done - Looping Constructs
elif - Conditional Constructs
else - Conditional Constructs
esac - Conditional Constructs
fi - Conditional Constructs
for - Looping Constructs
function - Shell Functions
if - Conditional Constructs
in - Conditional Constructs
select - Conditional Constructs
then - Conditional Constructs
time - Pipelines
until - Looping Constructs
while - Looping Constructs

Similar from UNIX
& , | , > , < , ! , =
# , $, (,) , ; , {, } , [,] , \

https://www.gnu.org/software/bash/manual/html_node/Reserved-Word-Index.html

BASH Variables (By content)

Apart from dividing variables in local and global variables, we can also divide them in categories according to the sort of content the variable contains.
In this respect, variables come in 4 types:

- String variables

- Integer variables

- Constant variables

- Array variables

String Variables – fasta formats

Sequence files are in a variety of formats commonly called FASTA format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, as single letter codes.

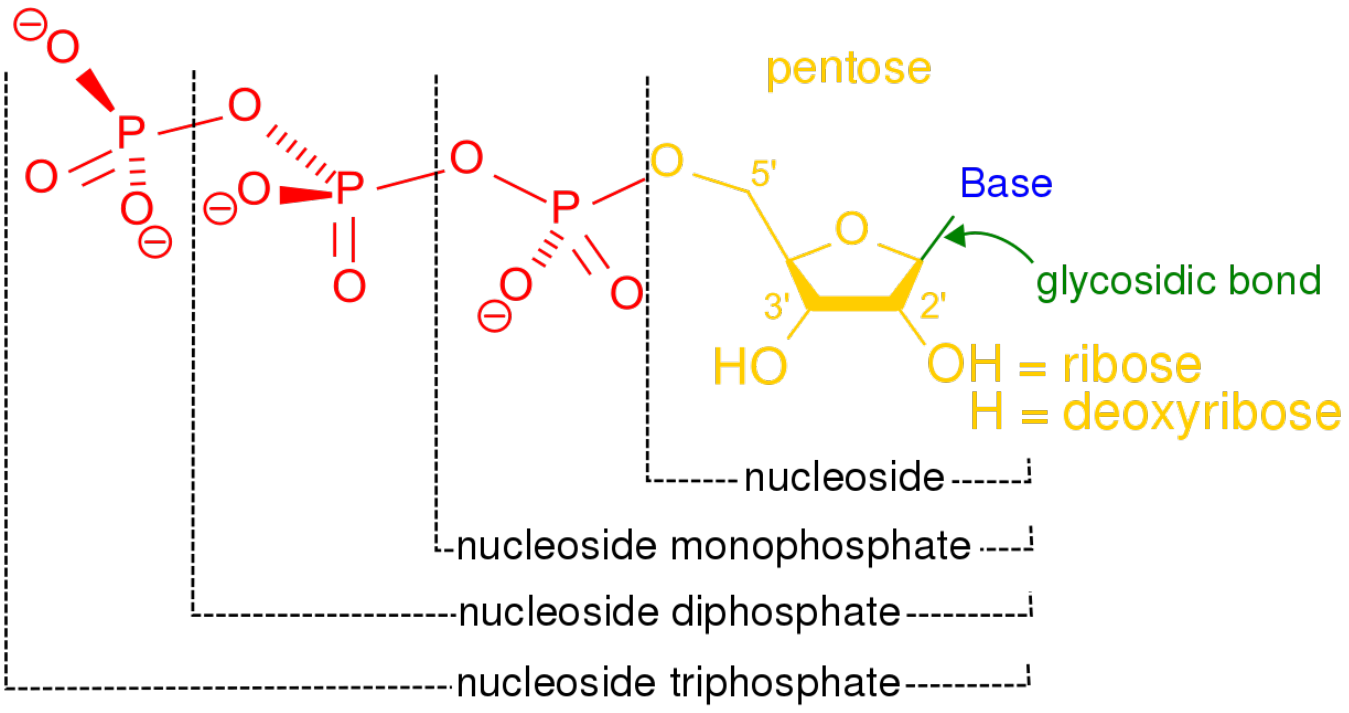
String Variables – fasta formats

Sequence files are in a variety of formats commonly called FASTA format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, as single letter codes.

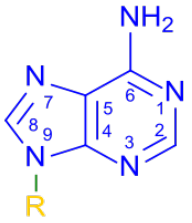
Nucleotides (5 single letters)
A, T, C, G, U

Guanine (G), adenine (A), cytosine (C) and thymine (T) for DNA.
For RNA, uracil (U) (rarely used)

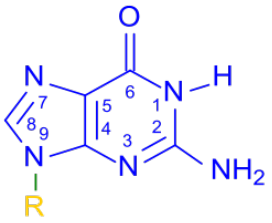
String Variables – fasta formats



Purines

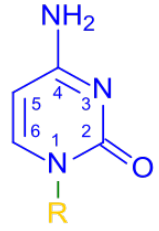


Adenine

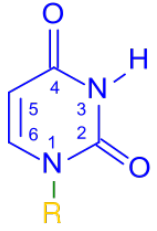


Guanine

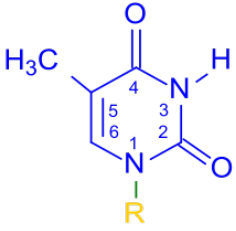
Pyrimidines



Cytosine

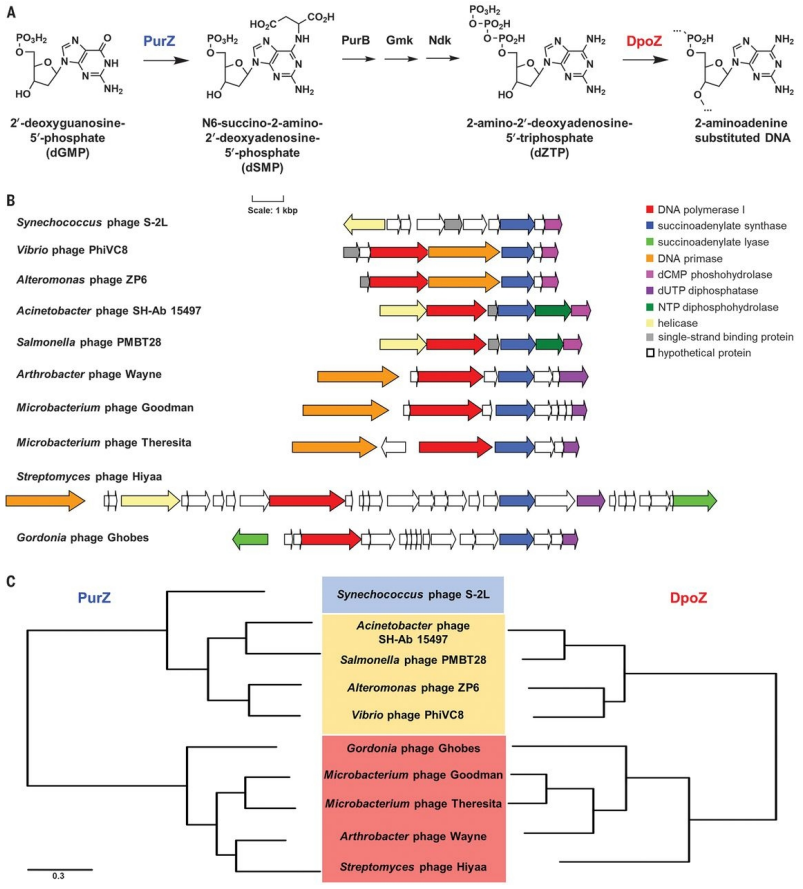


Uracil



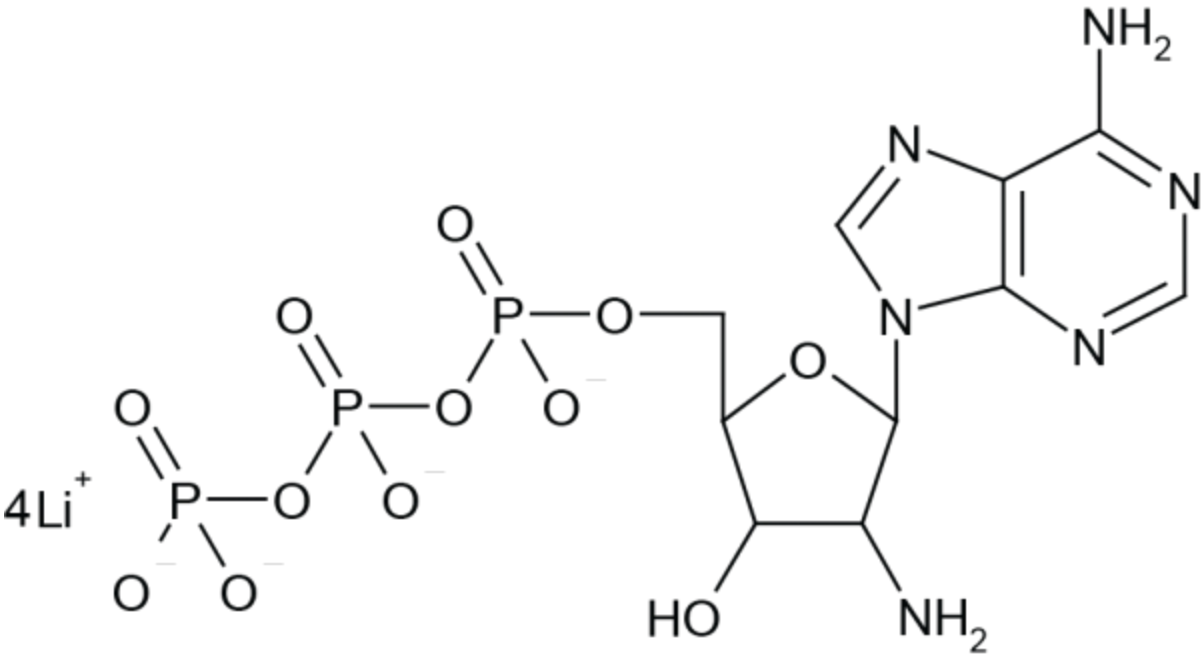
Thymine

String Variables – fasta formats



10.1126/science.abe6542

S-2L and Vibrio phage PhiVC8



2-amino-2′-deoxyadenosine-5′-triphosphate (2-amino adenine)

10.1126/science.abe6494

String Variables – fasta (Nucleotides)

Sequence files - .fa, .fasta, .ffn, .fna

BEST - .fna (fasta nucleotide file - contigs/reads)
.ffa - fasta feature nucleotide (nucleotide for protein ORFs)

String Variables – fasta (Nucleotides)

Sequence files - .fa, .fasta, .ffn, .fna

BEST - .fna (fasta nucleotide file - contigs/reads)
.ffa - fasta feature nucleotide (nucleotide for protein ORFs)

fna example

>Illumina_Single_End_Adapter1
GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTG
>Illumina_Single_End_Adapter2
CAAGCAGAAGACGGCATACGAGCTCTTCCGATCT
>Illumina_Single_End_PCR_Primer1
AATGATACGGCGACCACCGAGATCTACACTCTTTCCCTACACGACGCTCTTCCGATCT
>Illumina_Single_End_PCR_Primer2
CAAGCAGAAGACGGCATACGAGCTCTTCCGATCT

String Variables – fasta (Nucleotides)

Sequence files - .fa, .fasta, .ffn, .fna

BEST - .fna (fasta nucleotide file - contigs/reads)
.ffa - fasta feature nucleotide (nucleotide for protein ORFs)

ffa example

>CP001157.1_1 # 101 # 1537 # 1 # ID=1_1;partial=00;start_type=GTG;rbs_motif=GGAG/GAGG;rbs_spacer=5-10bp;gc_cont=0.630
GTGTCCGTGGAACTTTGGCAGCAGTGCGTCGAGCTGCTGCGCGACGAGCTGCCCG

>CP001157.1_2 # 1566 # 2669 # 1 # ID=1_2;partial=00;start_type=ATG;rbs_motif=GGA/GAG/AGG;rbs_spacer=5-10bp;gc_cont=0.627
ATGCATTTCACCATTCAACGCGAAGCCCTGCTGAAACCGCTGCAACTGGTCGCTGG
CGTCGTCGAGCGCC

String Variables – fastq (Nucleotides)

Sequence files - .fq, fastq, fasta/qual (rare)

This is the fasta plus the quality encoded (raw/trim reads)

fastq example

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCC
ATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***
+*''))**55CCF>>>>>>CCCCCCC65
```

String Variables – fastq (Nucleotides)

Sequence files - .fq, fastq, fasta/qual (rare)

This is the fasta plus the quality encoded (raw/trim reads)

The byte representing quality runs from 0x21 (lowest quality; '!' in ASCII) to 0x7e (highest quality; '~' in ASCII). Here are the quality value characters in left-to-right increasing order of quality (ASCII):

```
!"#$%&'()*+,-./0123456789:;<=>
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvw
xyz{|}~
```

American Standard Code for Information Interchange (ASCII)

String Variables – fastq (Nucleotides)

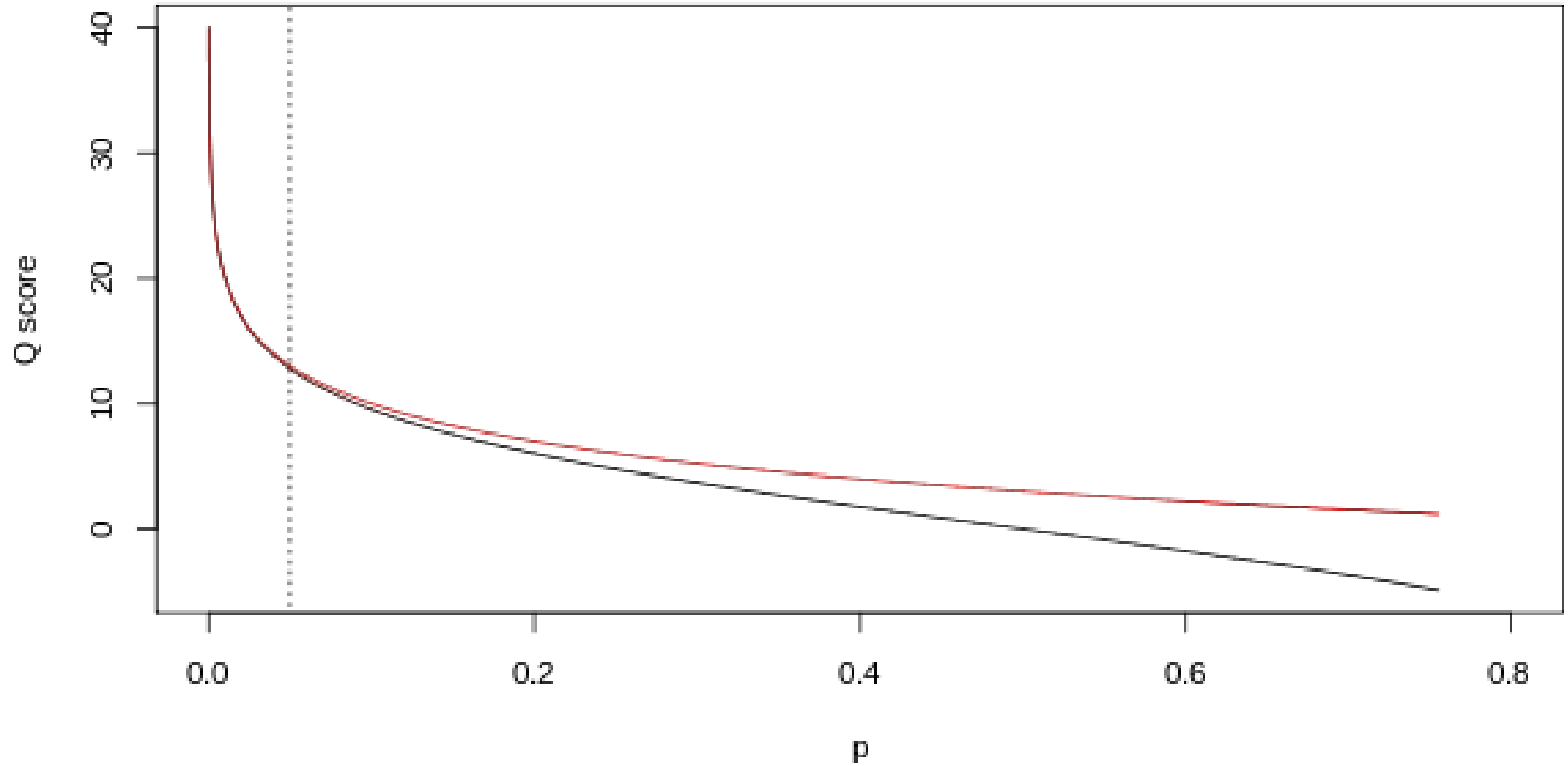
A quality value Q is an integer mapping of p (i.e., the probability that the corresponding base call is incorrect). Two different equations have been in use. The first is the standard Sanger variant to assess reliability of a base call, otherwise known as Phred quality score:

$$\boldsymbol{Q}\ \text{sanger} = -10 \log_{10} p$$

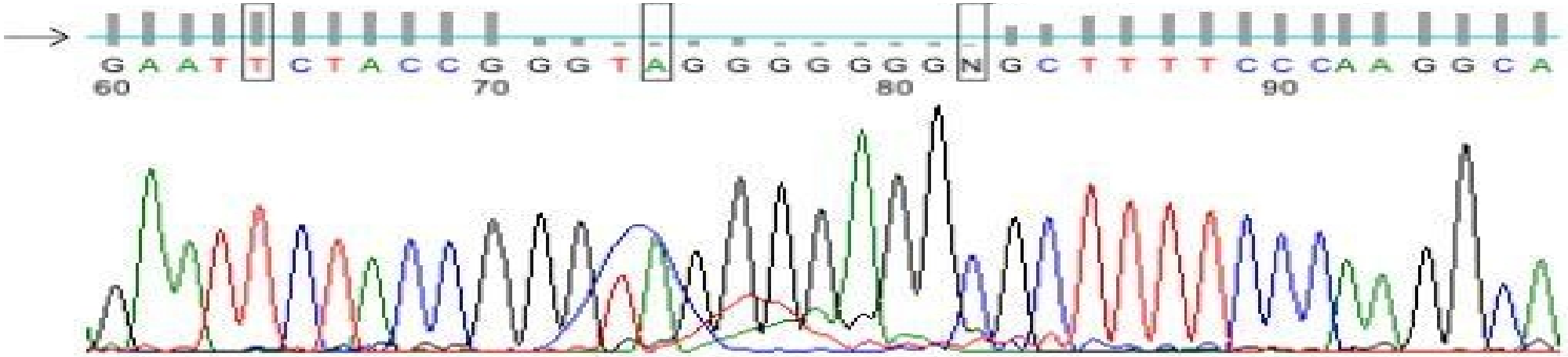
Illumina (solexa) Genome Analyzer earlier used a different mapping, encoding the odds p/(1-p) instead of the probability p:

$$\boldsymbol{Q}\ \textit{solexa (prior to 1.3v)} = -10 \log_{10} p\ /\ 1 - p$$

String Variables – fastq (Nucleotides)



String Variables – fastq (Phred scores)



Phred quality scores are logarithmically linked to error probabilities

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%
60	1 in 1,000,000	99.9999%

String Variables – fasta formats

Sequence files are in a variety of formats commonly called FASTA format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, as single letter codes.

Amino Acids (20 single letters, 3 nucleotide = 1 codon)
R, H, K, D, E, S, T, N, Q, C, G, P, A, V, I, L, M, F, Y, W

22 possible (hydroxyproline and selenomethionine rare)

String Variables – fasta formats

Sequence files are in a variety of formats commonly called FASTA format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, as single letter codes.

Amino Acids (20 single letters, 3 nucleotide = 1 codon)
R, H, K, D, E, S, T, N, Q, C, G, P, A, V, I, L, M, F, Y, W

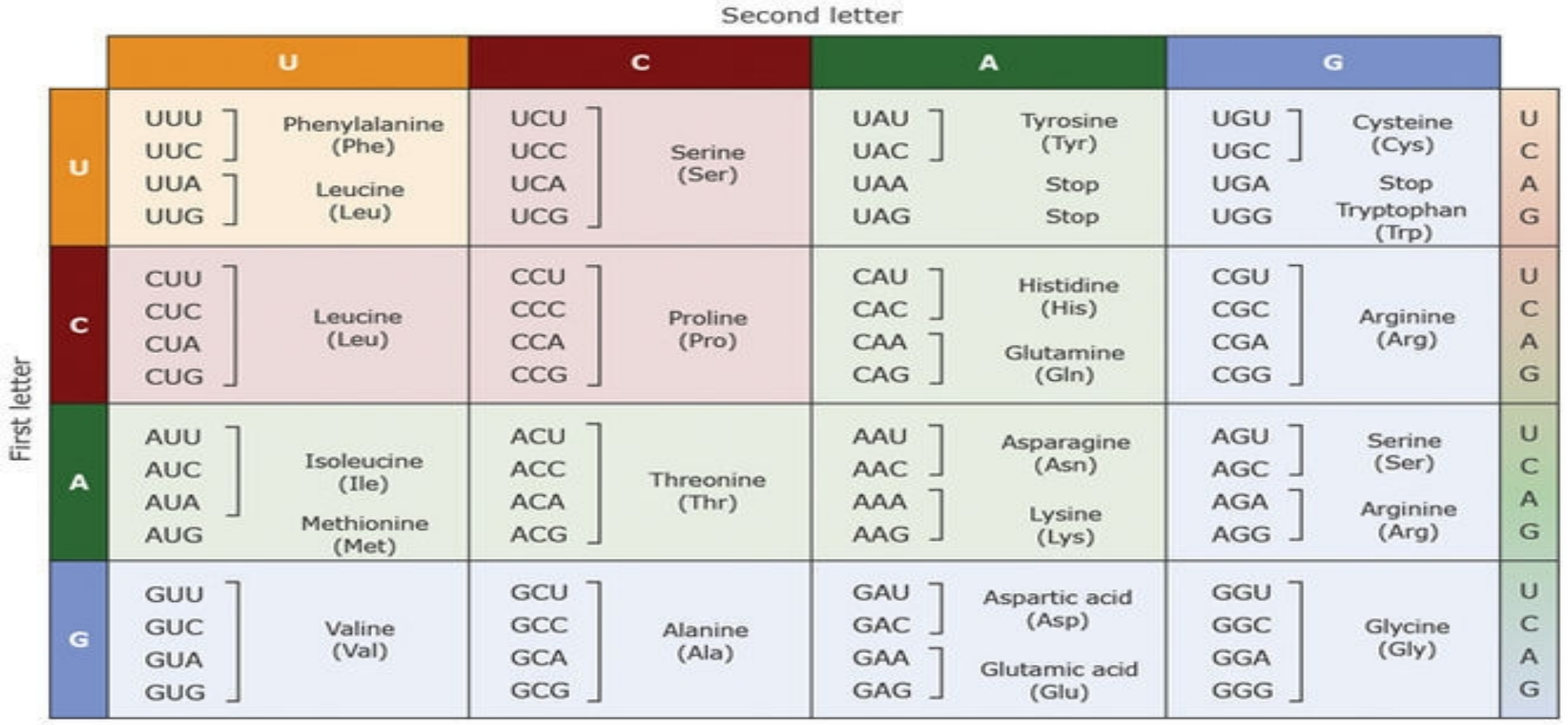
22 possible (hydroxyproline and selenomethionine rare)

String Variables – fasta formats (DNA)

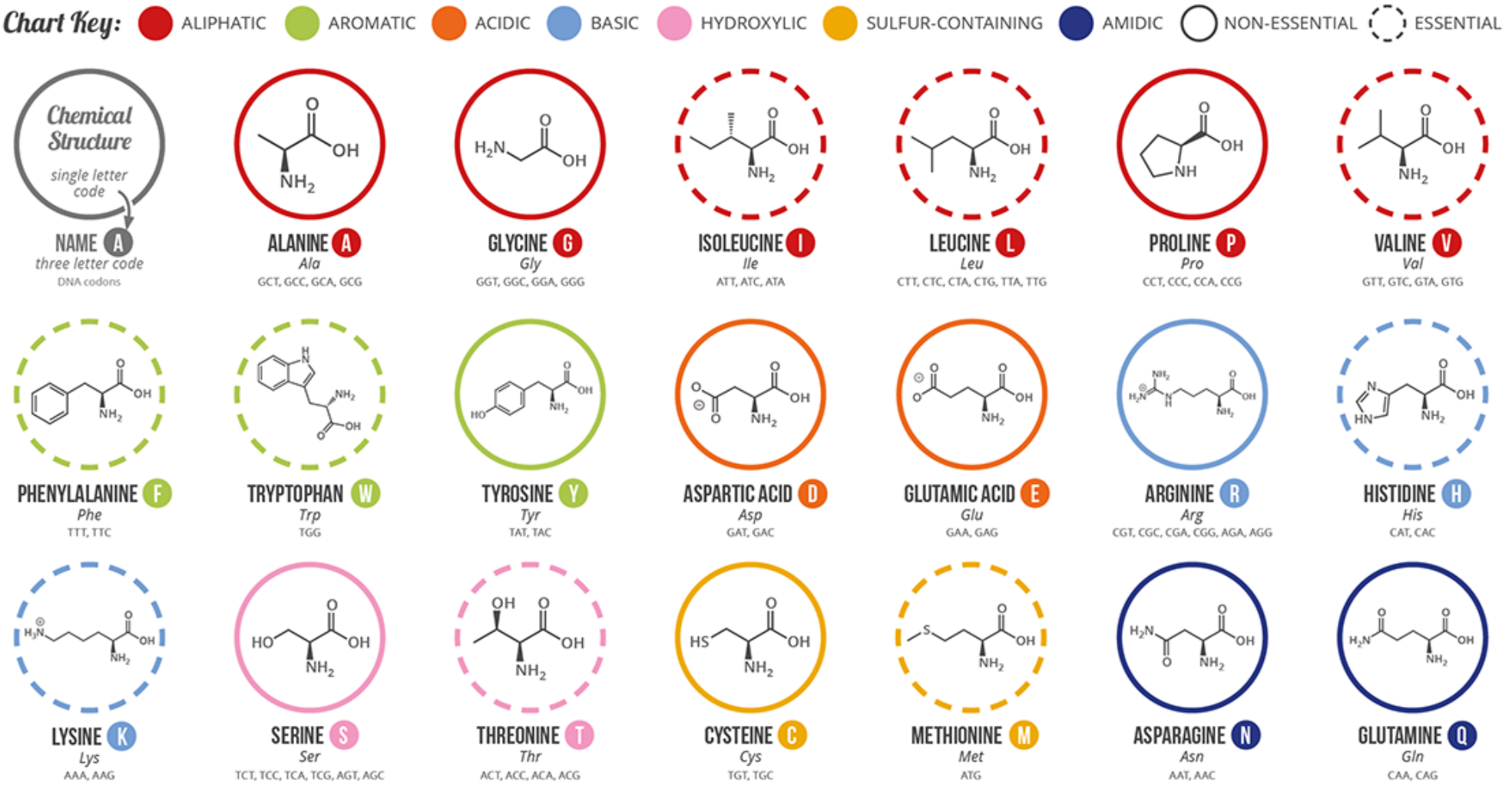
	T		C		A		G		
T	TTT	Phe	TCT	Ser	TAT	Tyr	TGT	Cys	T
	TTC	Phe	TCC	Ser	TAC	Tyr	TGC	Cys	C
	TTA	Leu	TCA	Ser	TAA	STOP	TGA	STOP	A
	TTG	Leu	TCG	Ser	TAG	STOP	TGG	Trp	G
C	CTT	Leu	CCT	Pro	CAT	His	CGT	Arg	T
	CTC	Leu	CCC	Pro	CAC	His	CGC	Arg	C
	CTA	Leu	CCA	Pro	CAA	Gln	CGA	Arg	A
	CTG	Leu	CCG	Pro	CAG	Gln	CGG	Arg	G
A	ATT	Ile	ACT	Thr	AAT	Asn	AGT	Ser	T
	ATC	Ile	ACC	Thr	AAC	Asn	AGC	Ser	C
	ATA	Ile	ACA	Thr	AAA	lys	AGA	Arg	A
	ATG	Met*	ACG	Thr	AAG	Lys	AGG	Arg	G
G	GTT	Val	GCT	Ala	GAT	Asp	GGT	Gly	T
	GTC	Val	GCC	Ala	GAC	Asp	GGC	Gly	C
	GTA	Val	GCA	Ala	GAA	Glu	GGA	Gly	A
	GTG	Val	GCG	Ala	GAG	Glu	GGG	Gly	G

*When within gene; at beginning of gene. ATG signals start of translation.

String Variables – fasta formats (RNA)



String Variables – fasta formats (AA's)



String Variables – faa (amino acids)

Sequence files are in a variety of formats commonly called FASTA format is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, as single letter codes.

Example (.faa)

```
>CP001157.1_1 # 101
MSVELWQQCVELLRDELPAQQFNTWIRPLQVEADGDEL*
>CP001157.1_2 # 1566
MKFPDYERVLPRGGDKKVLGDRQLLREAFSRTAILSNEK*
>CP001157.1_3 # 2688
MSLGRVTVTAVRNLHPVTLNPSPRINILYGPNGSGKTSLL*
```

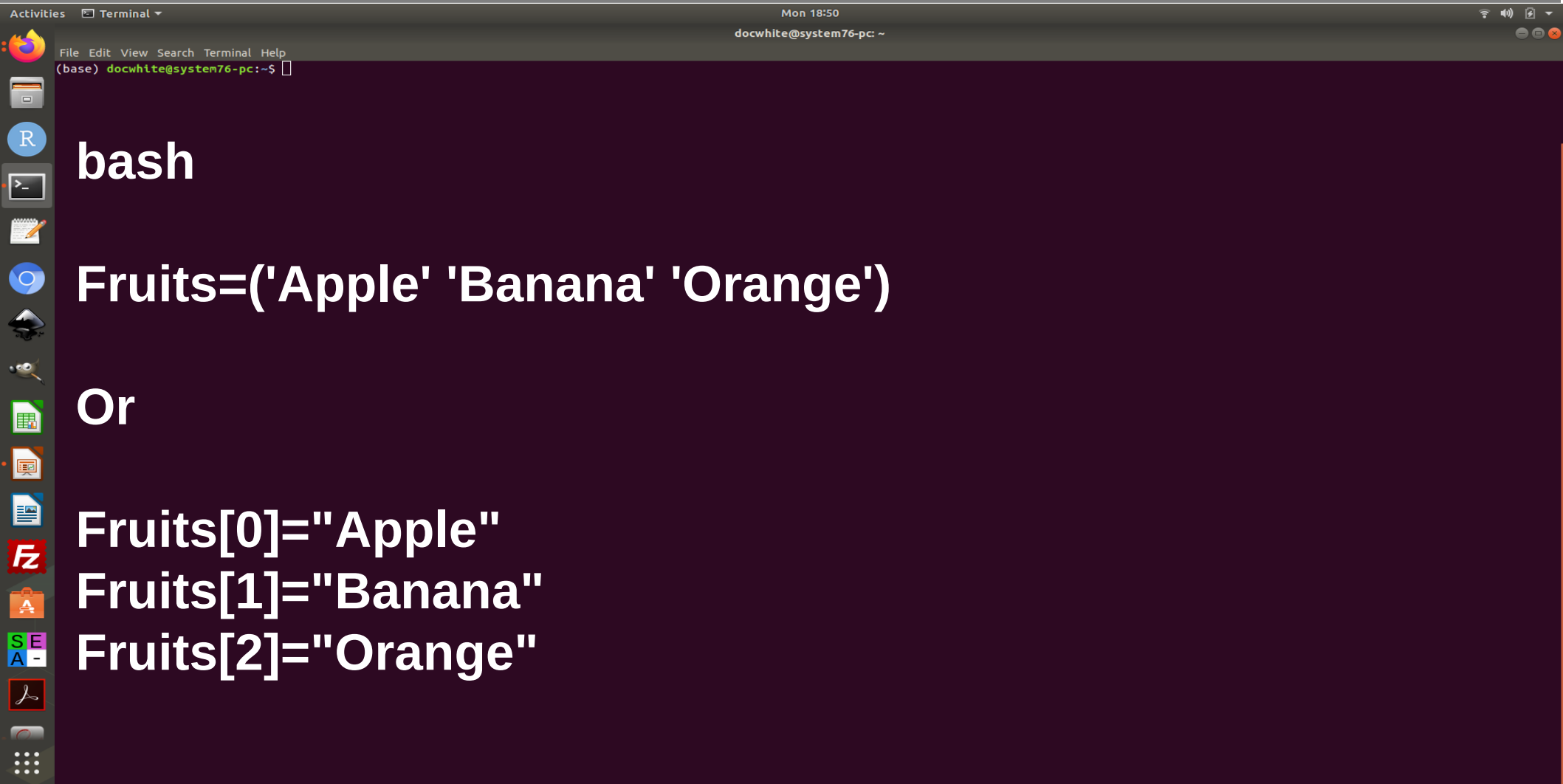
Array Variables in Bash

An array is a variable containing multiple values. Any variable may be used as an array.

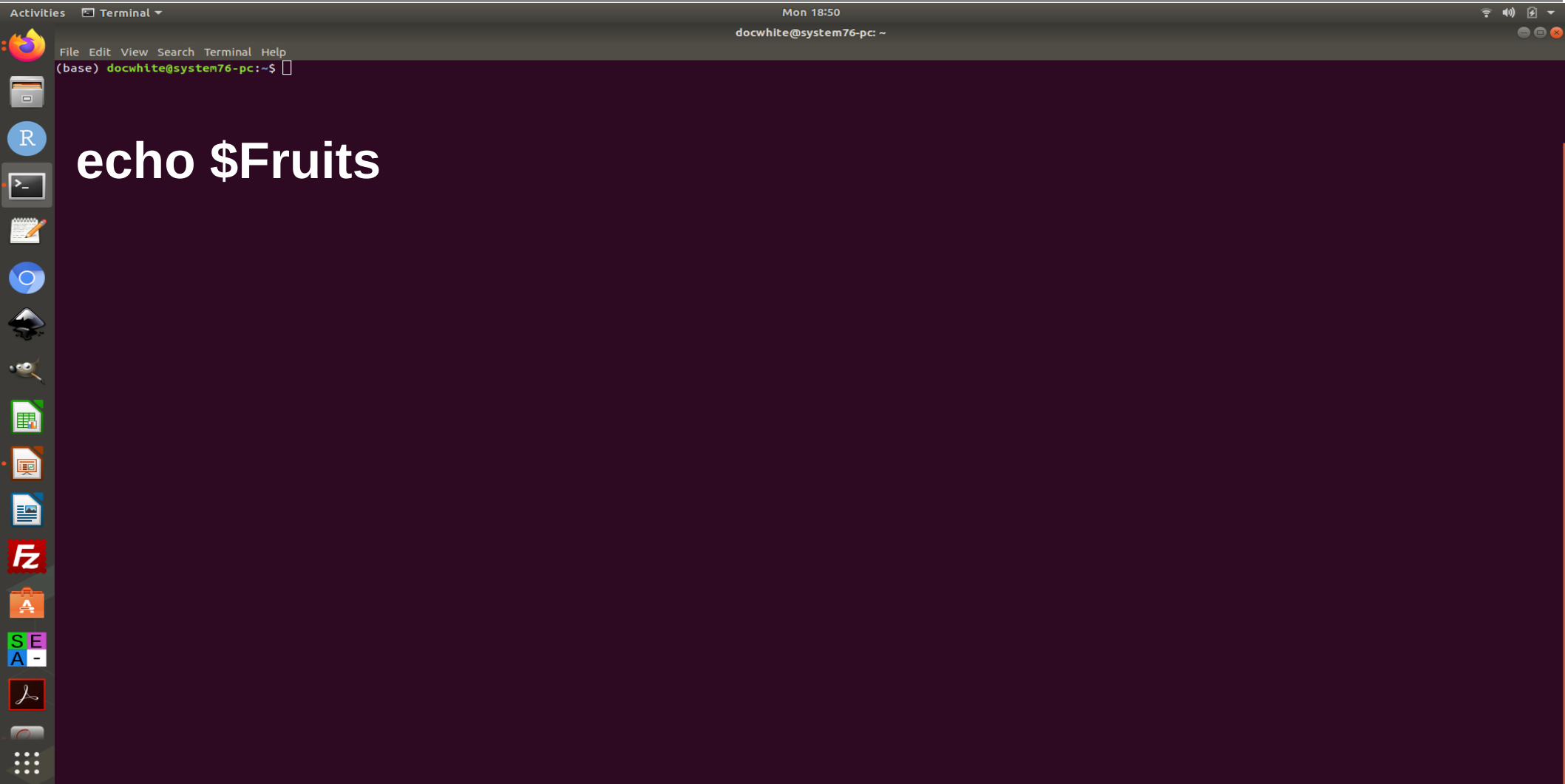
There is no maximum limit to the size of an array, nor any requirement that member variables be indexed or assigned contiguously.

Arrays are zero-based: the first element is indexed with the number 0.

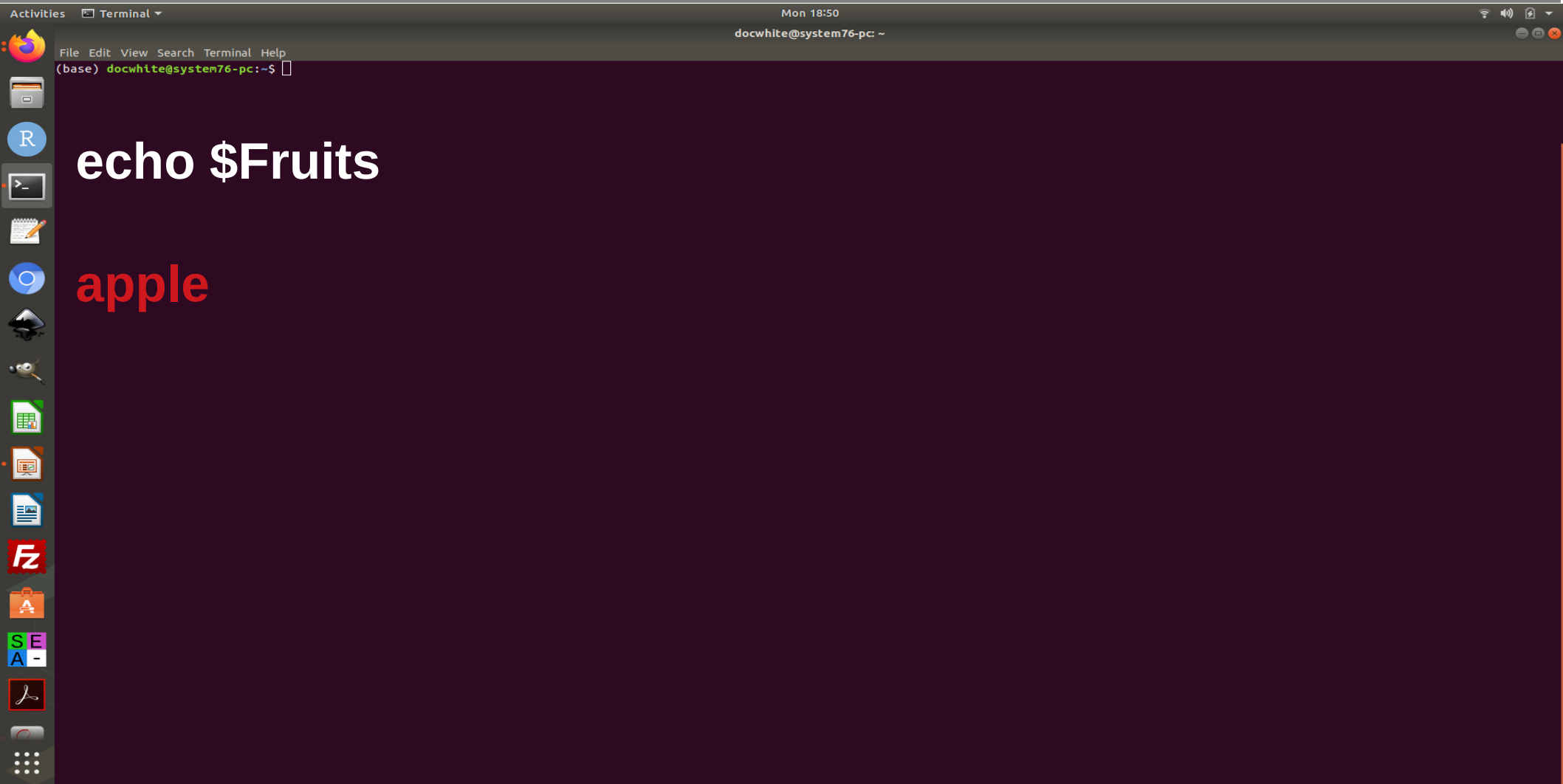
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
bash
Fruits=('Apple' 'Banana' 'Orange')
Or
Fruits[0]="Apple"
Fruits[1]="Banana"
Fruits[2]="Orange"

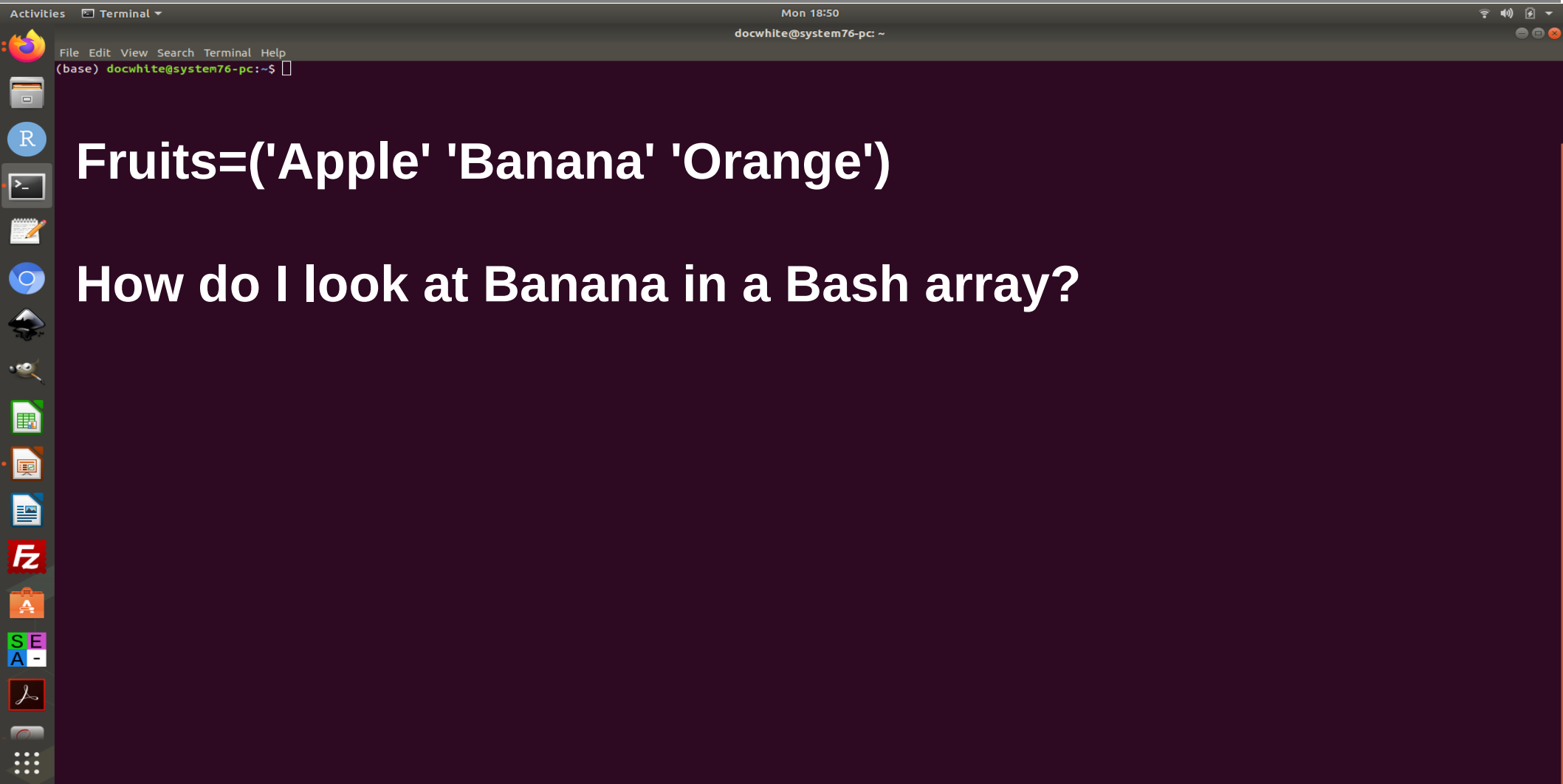
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
echo $Fruits

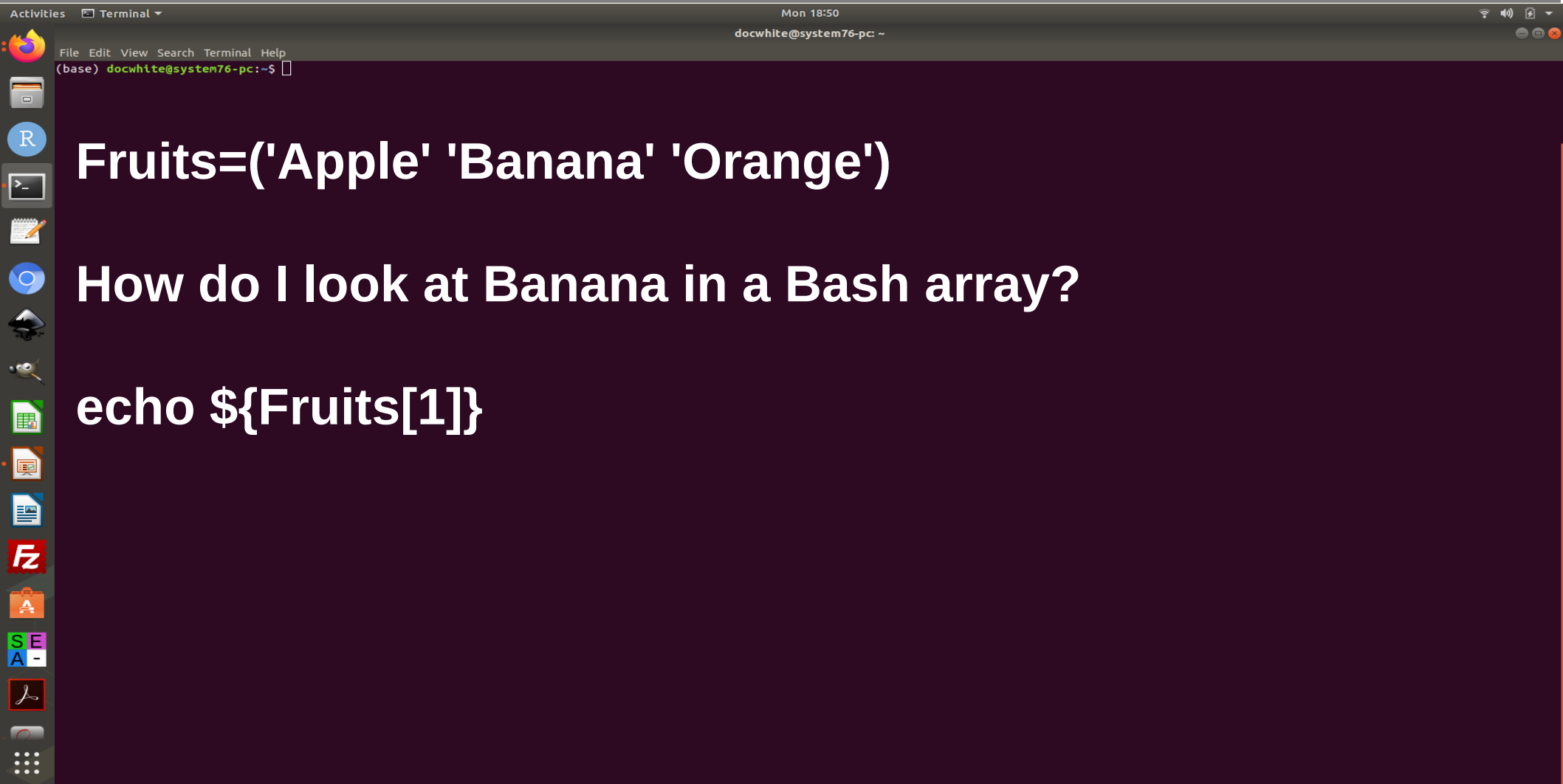
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
echo $Fruits
apple

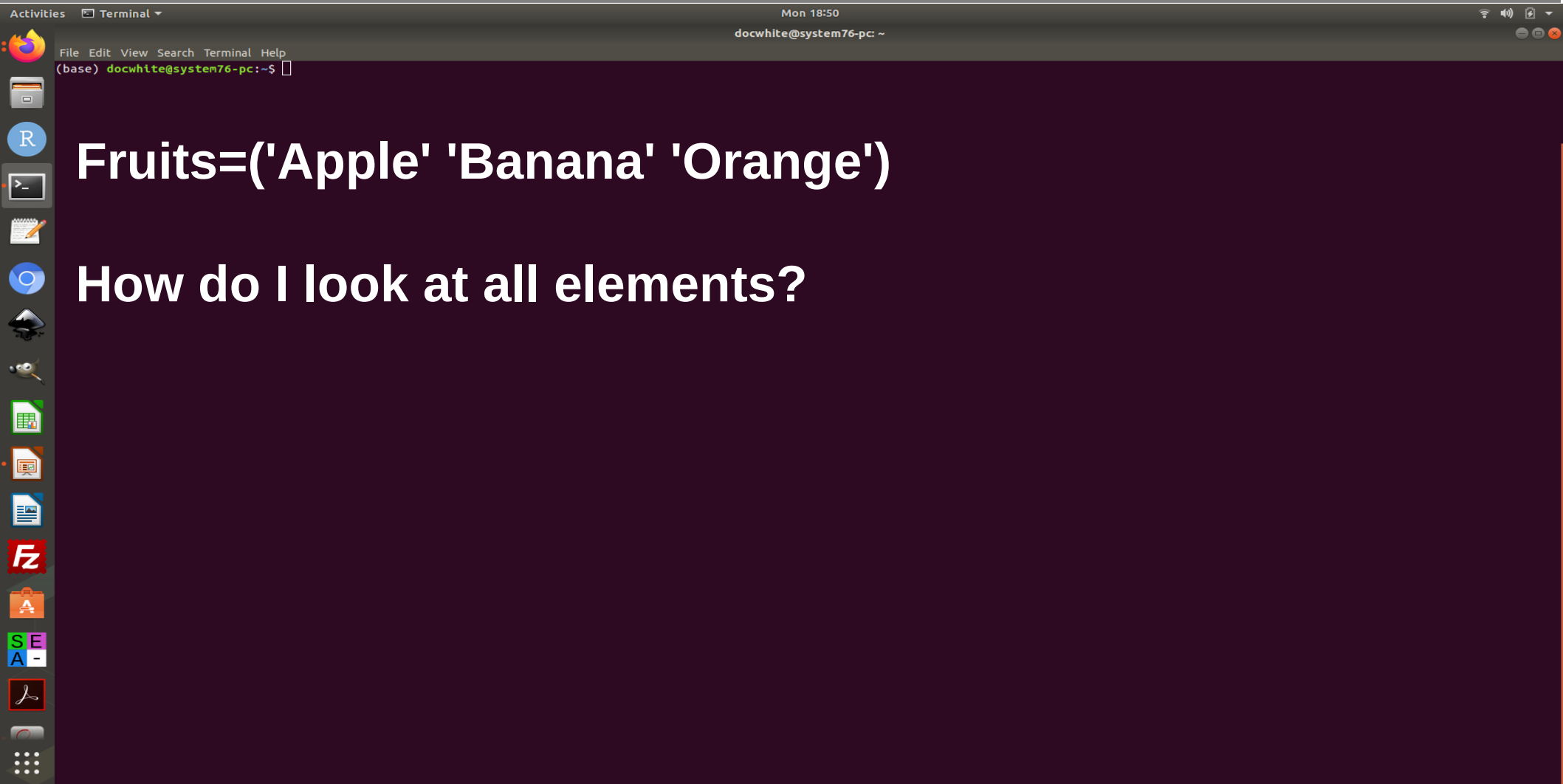
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do I look at Banana in a Bash array?

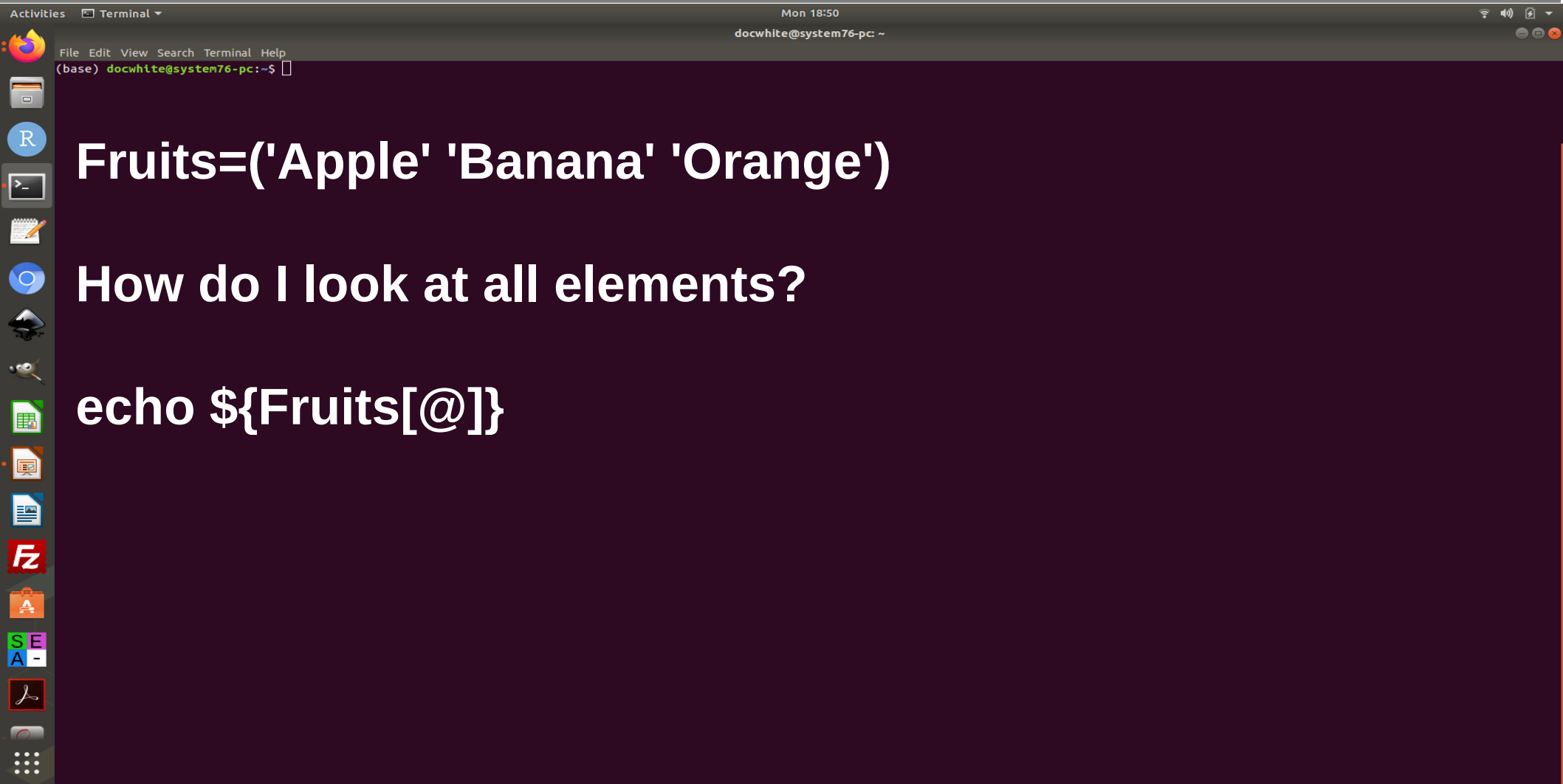
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do I look at Banana in a Bash array?
echo ${Fruits[1]}

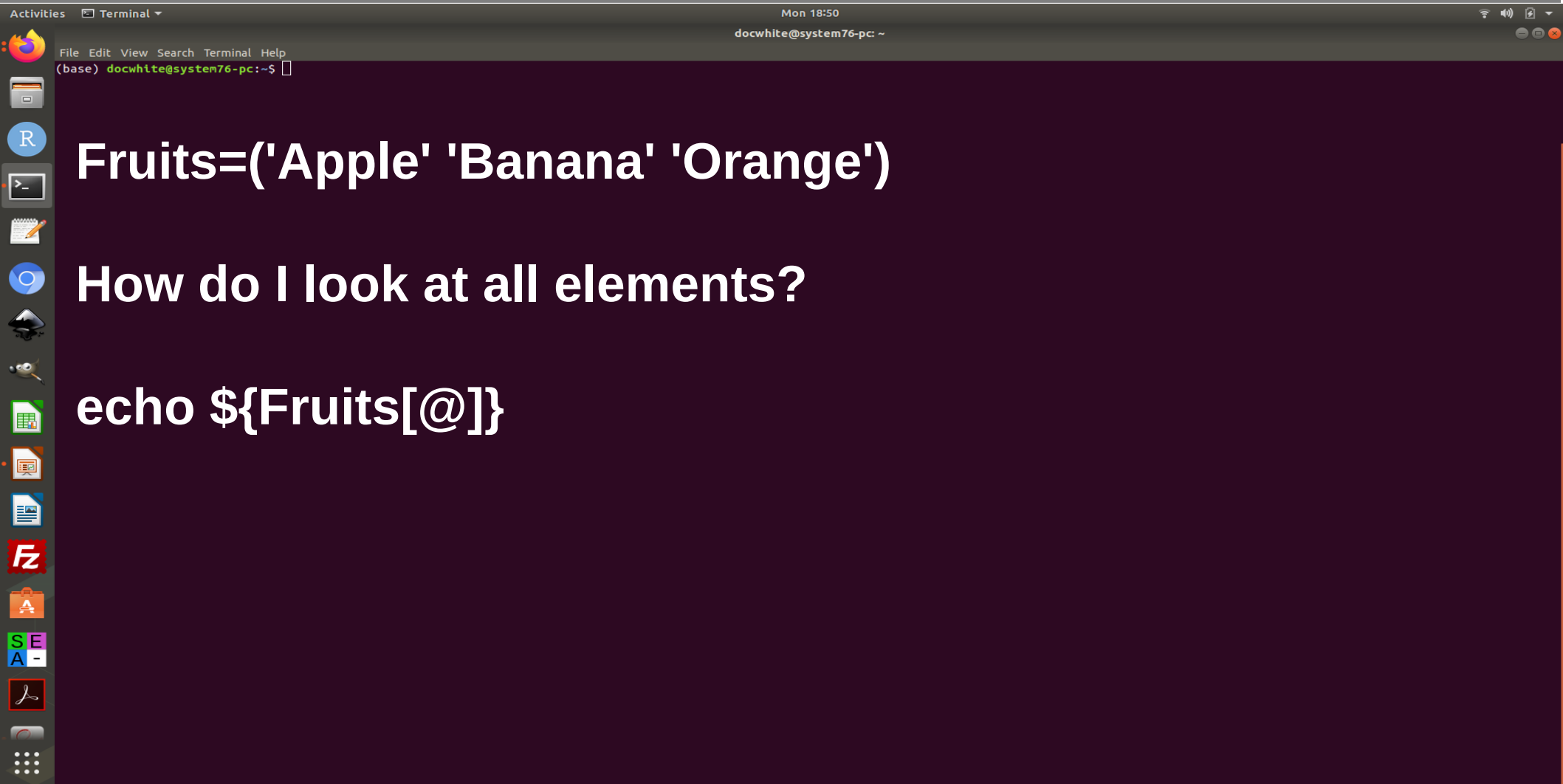
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do I look at all elements?

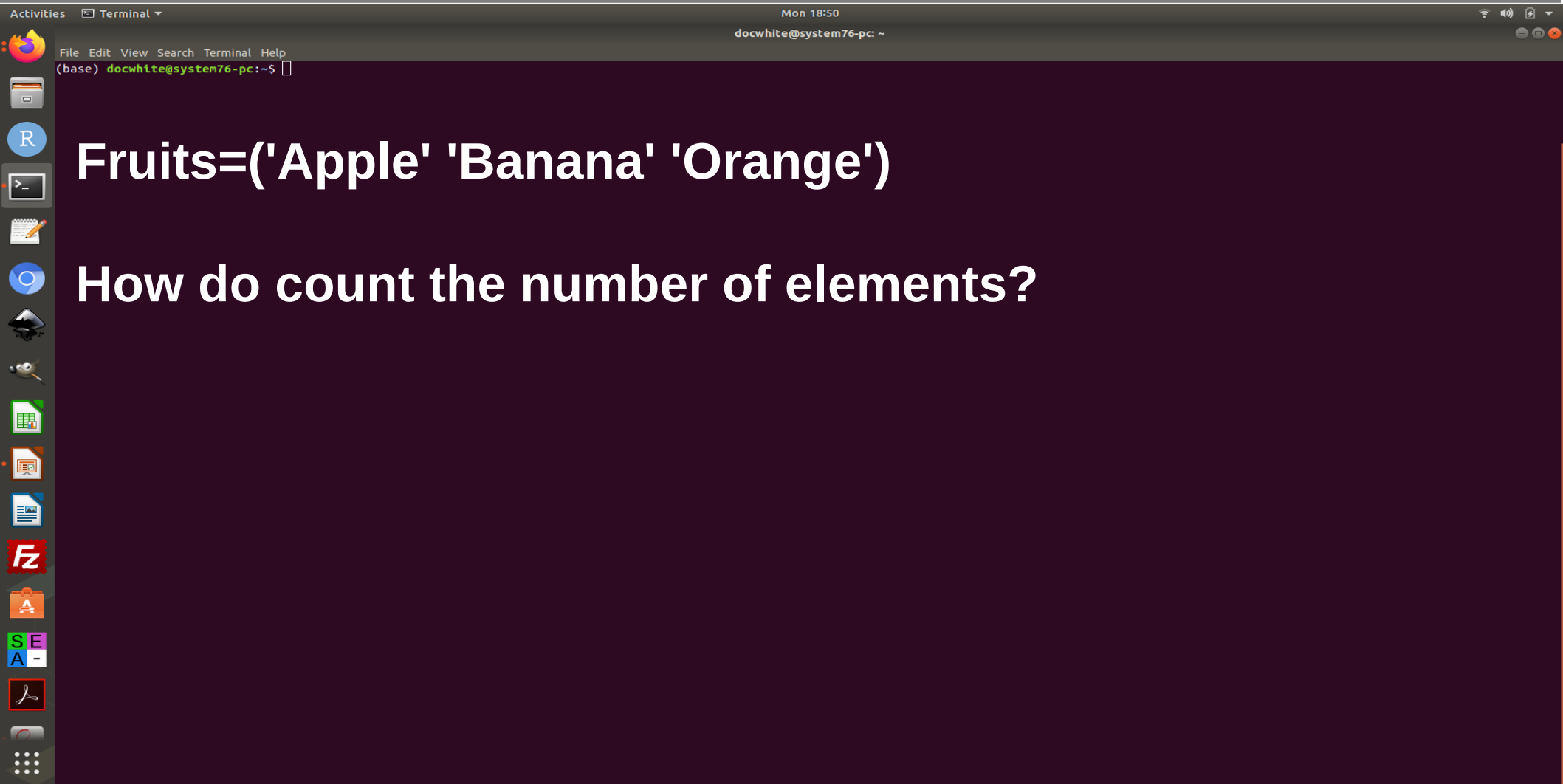
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do I look at all elements?
echo ${Fruits[@]}

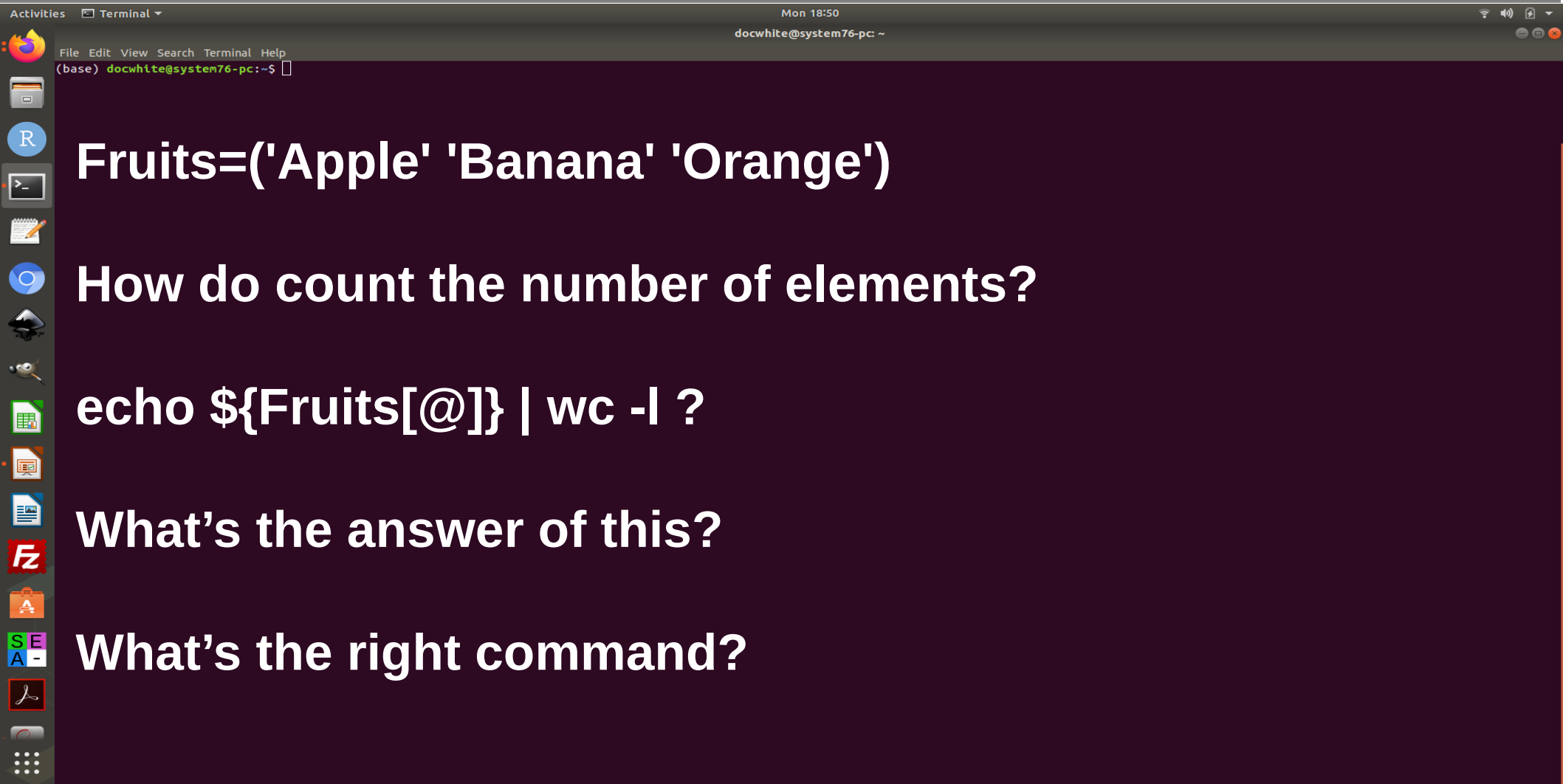
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do I look at all elements?
echo ${Fruits[@]}

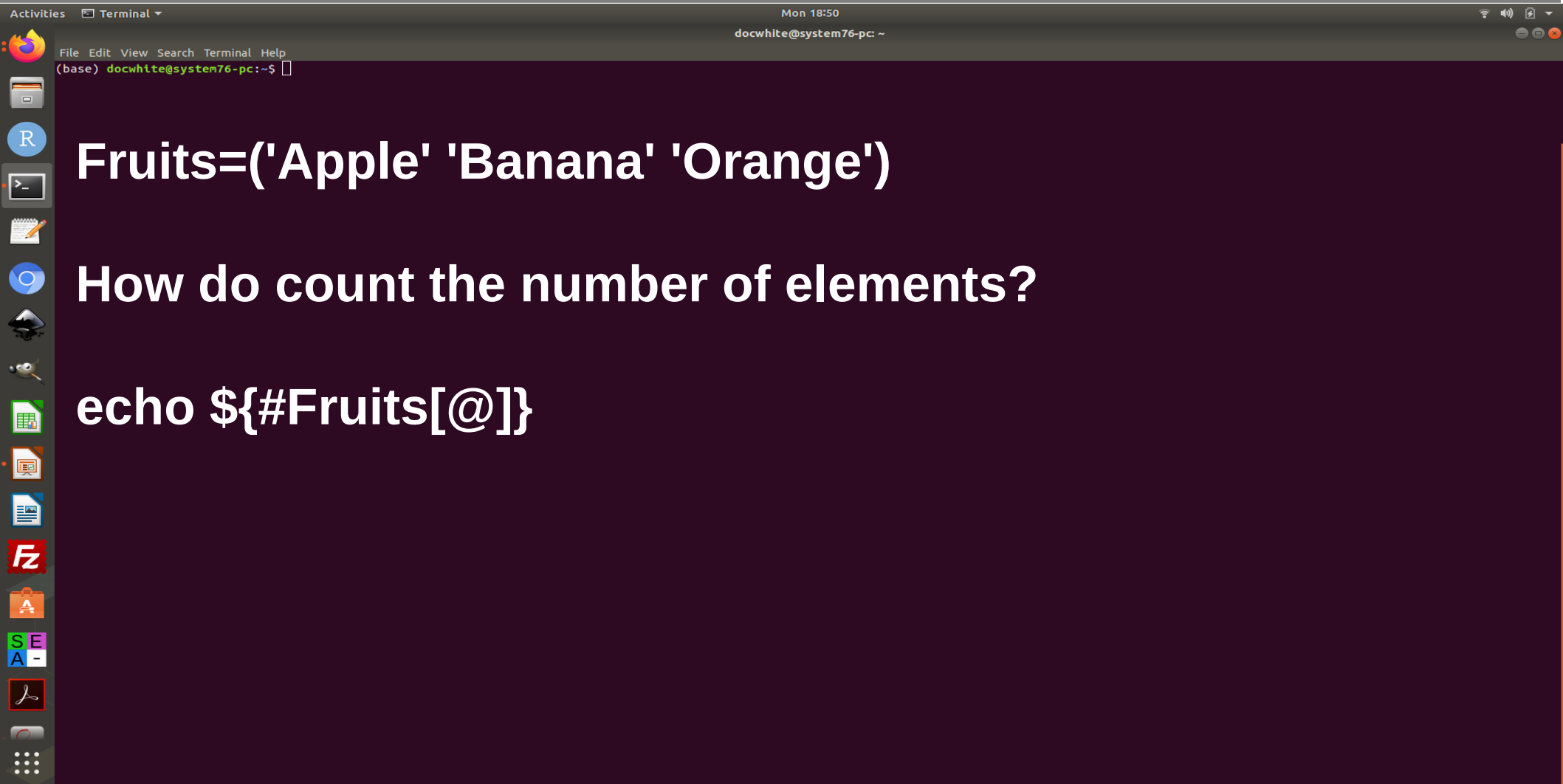
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do count the number of elements?

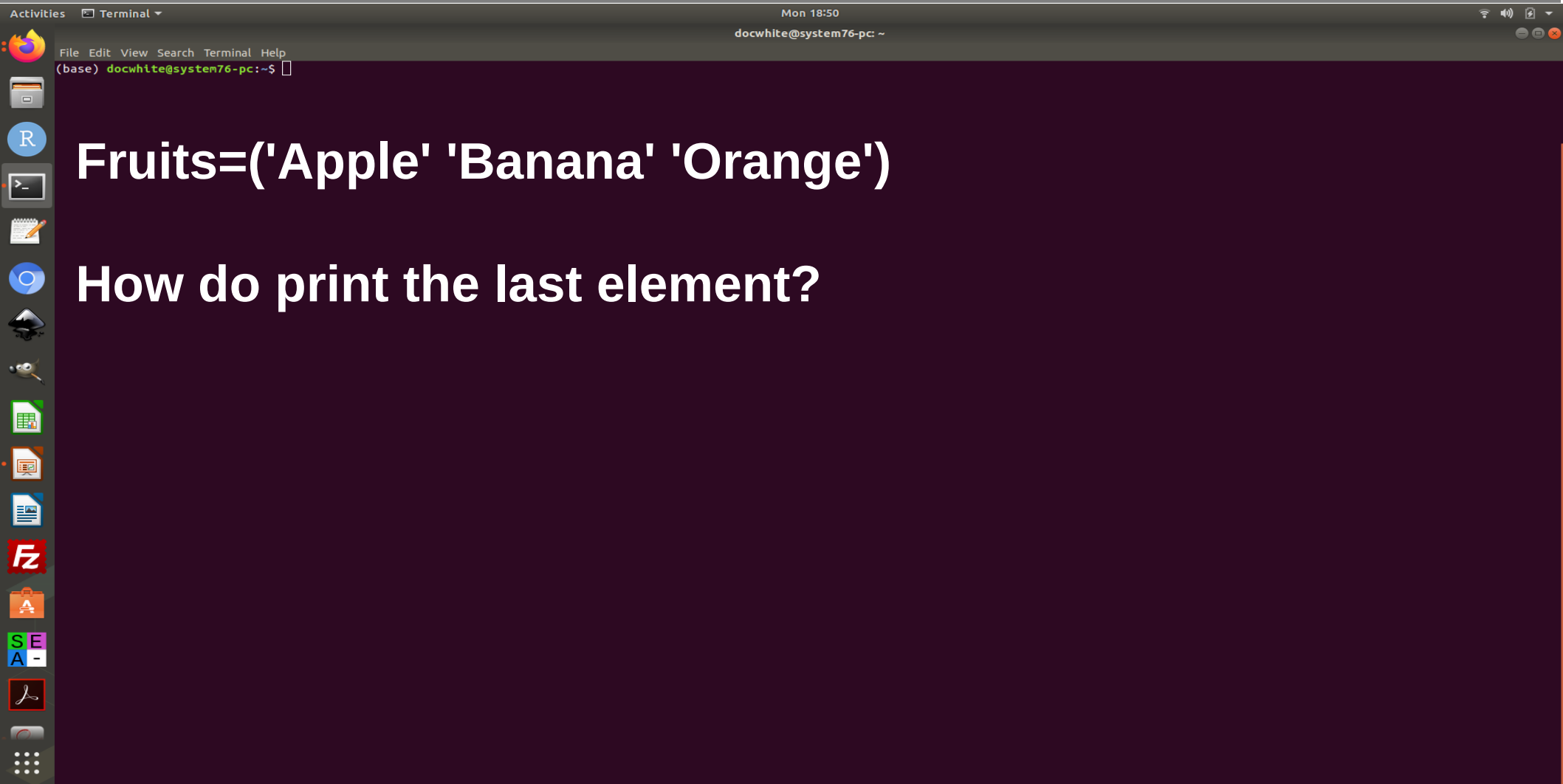
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do count the number of elements?
echo ${Fruits[@]} | wc -l ?
What's the answer of this?
What's the right command?

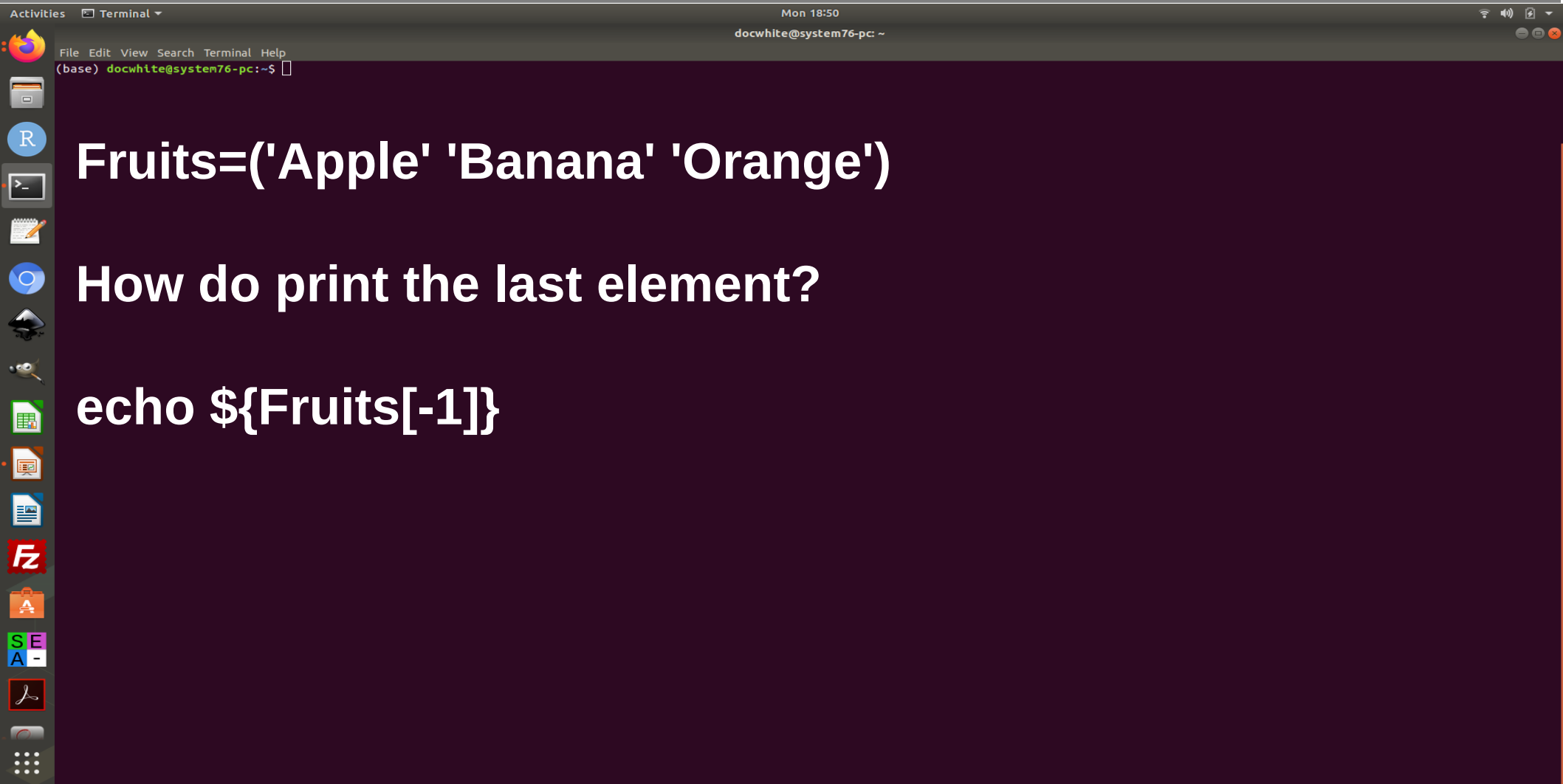
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do count the number of elements?
echo ${#Fruits[@]}

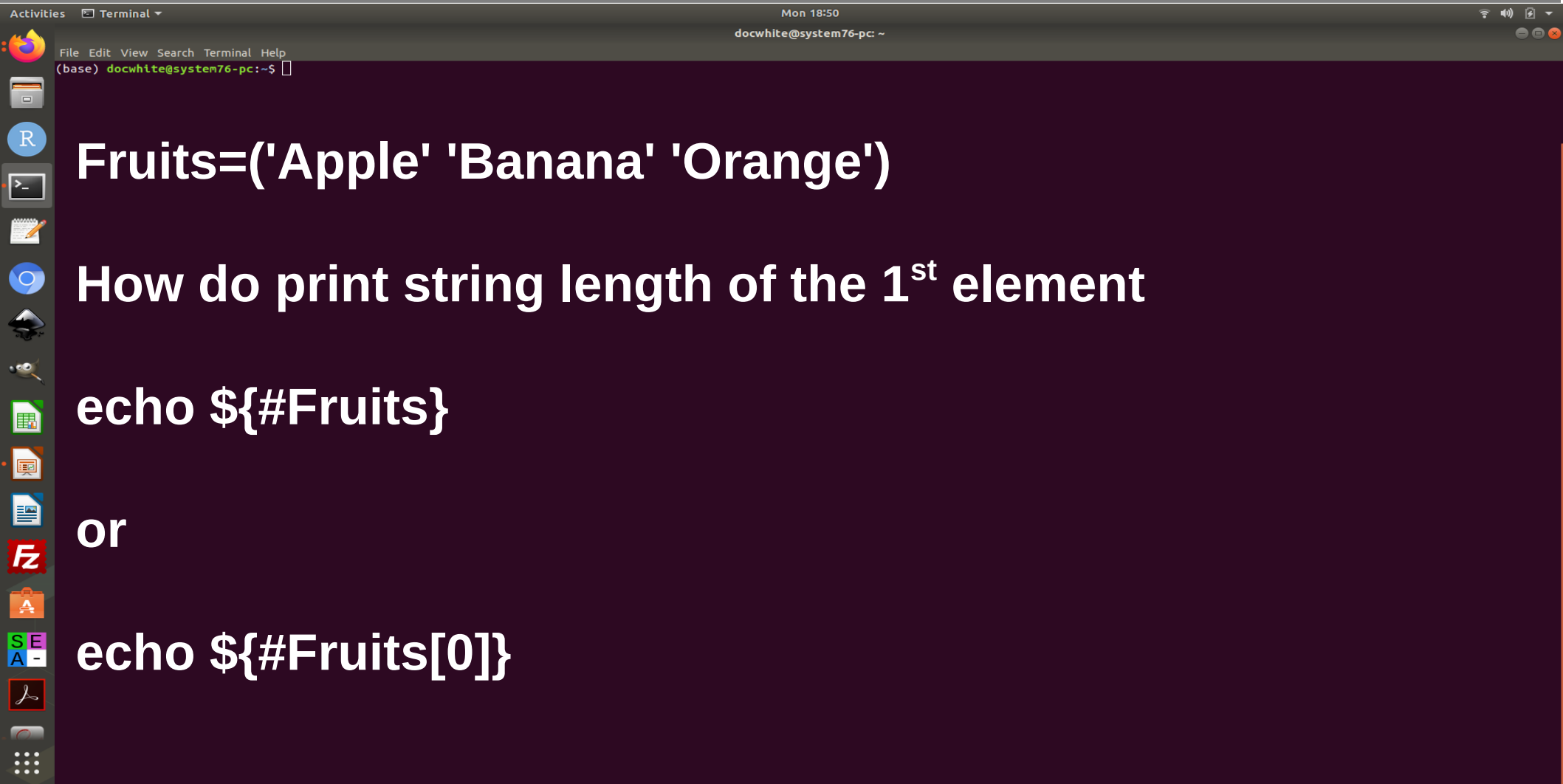
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do print the last element?

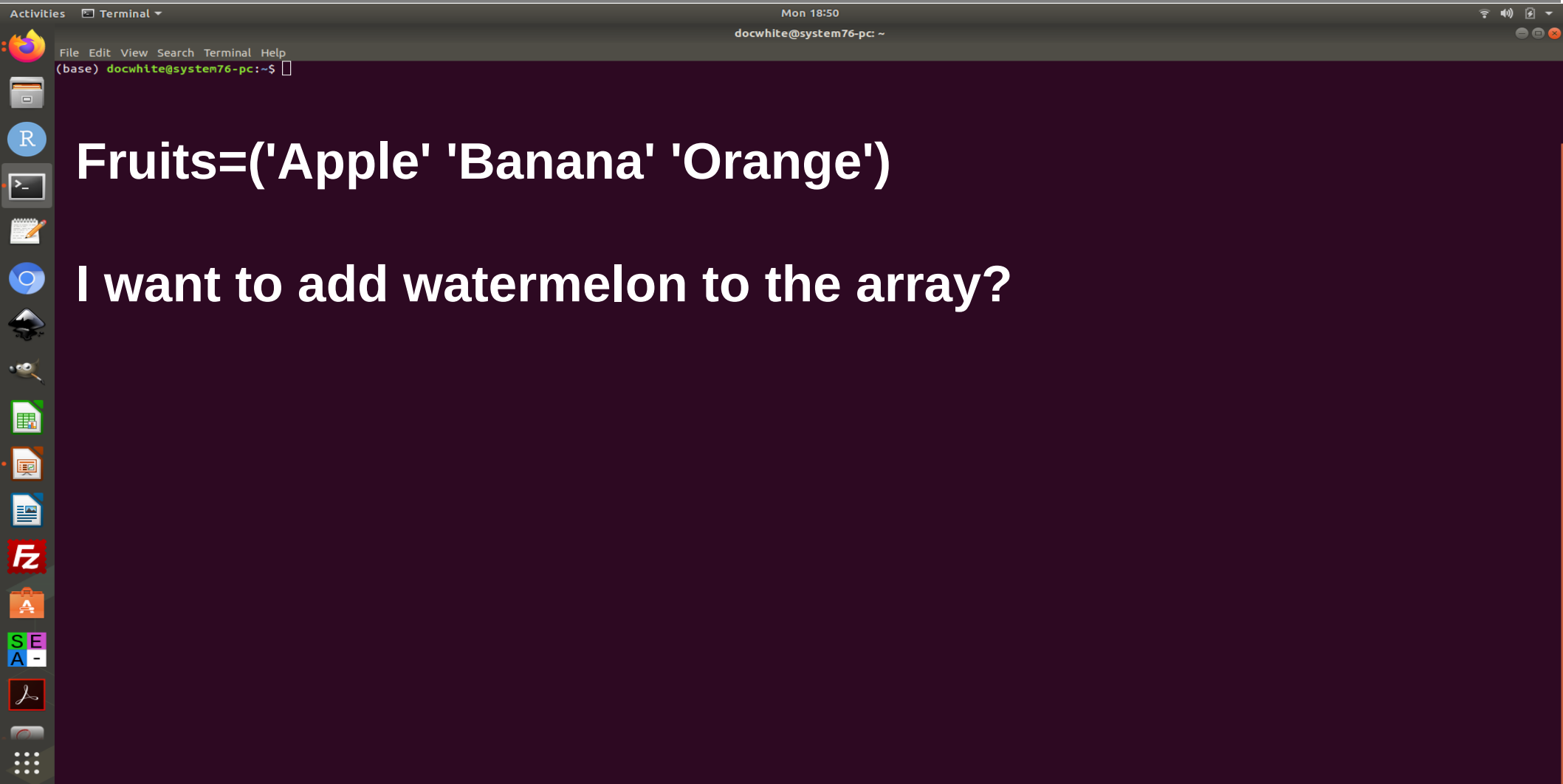
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do print the last element?
echo ${Fruits[-1]}

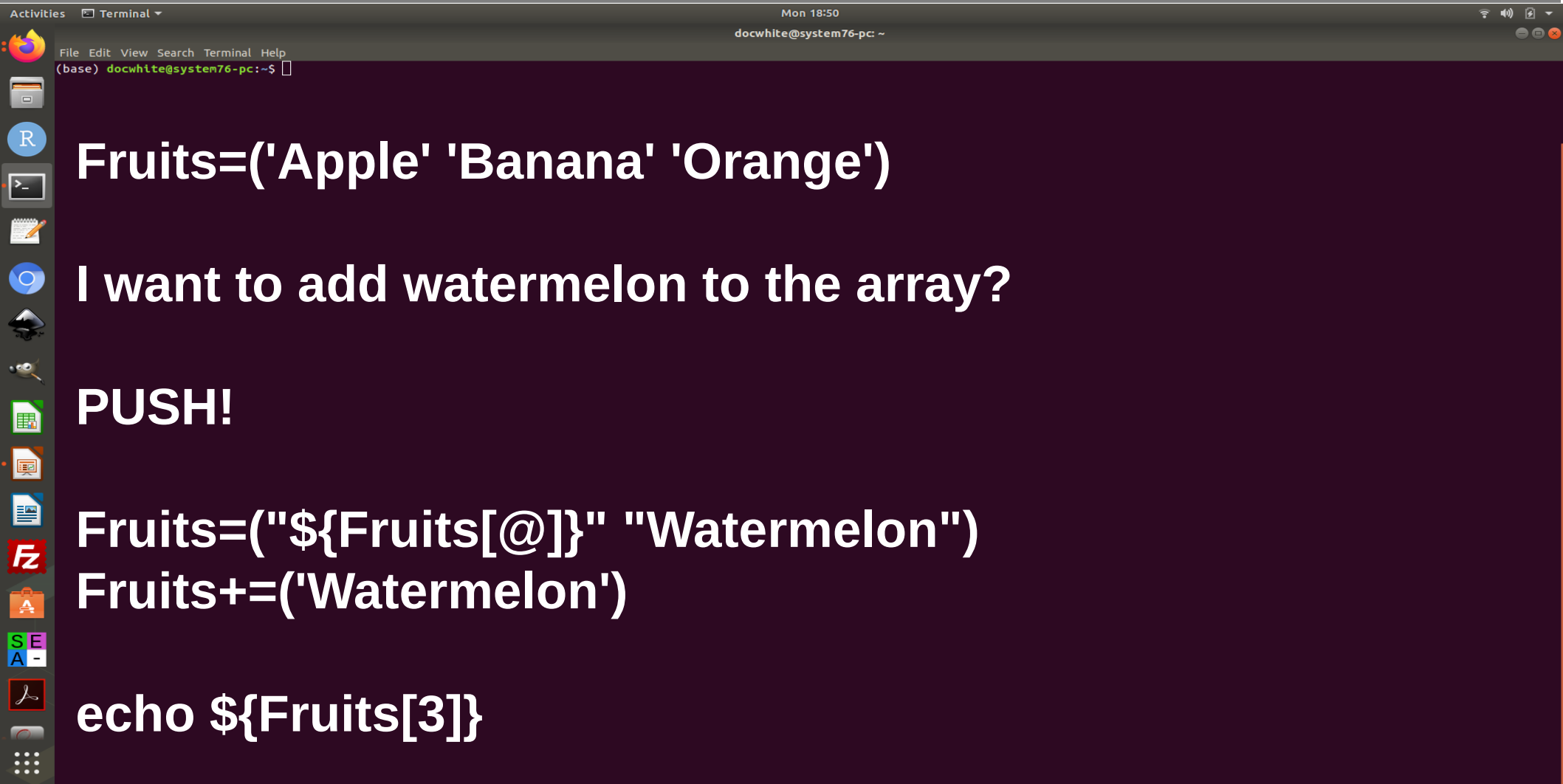
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
How do print string length of the 1st element
echo ${#Fruits}
or
echo ${#Fruits[0]}

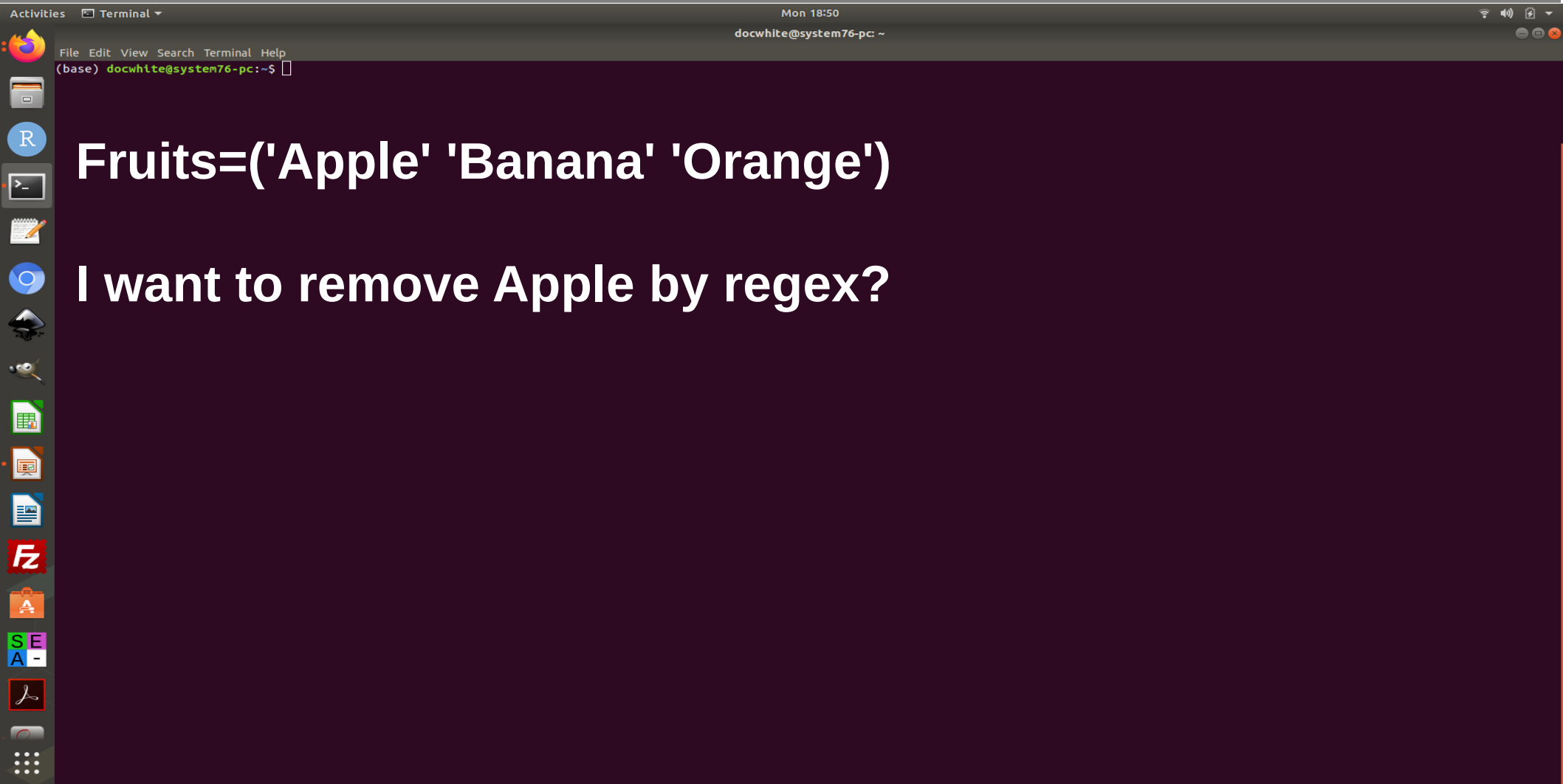
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
I want to add watermelon to the array?

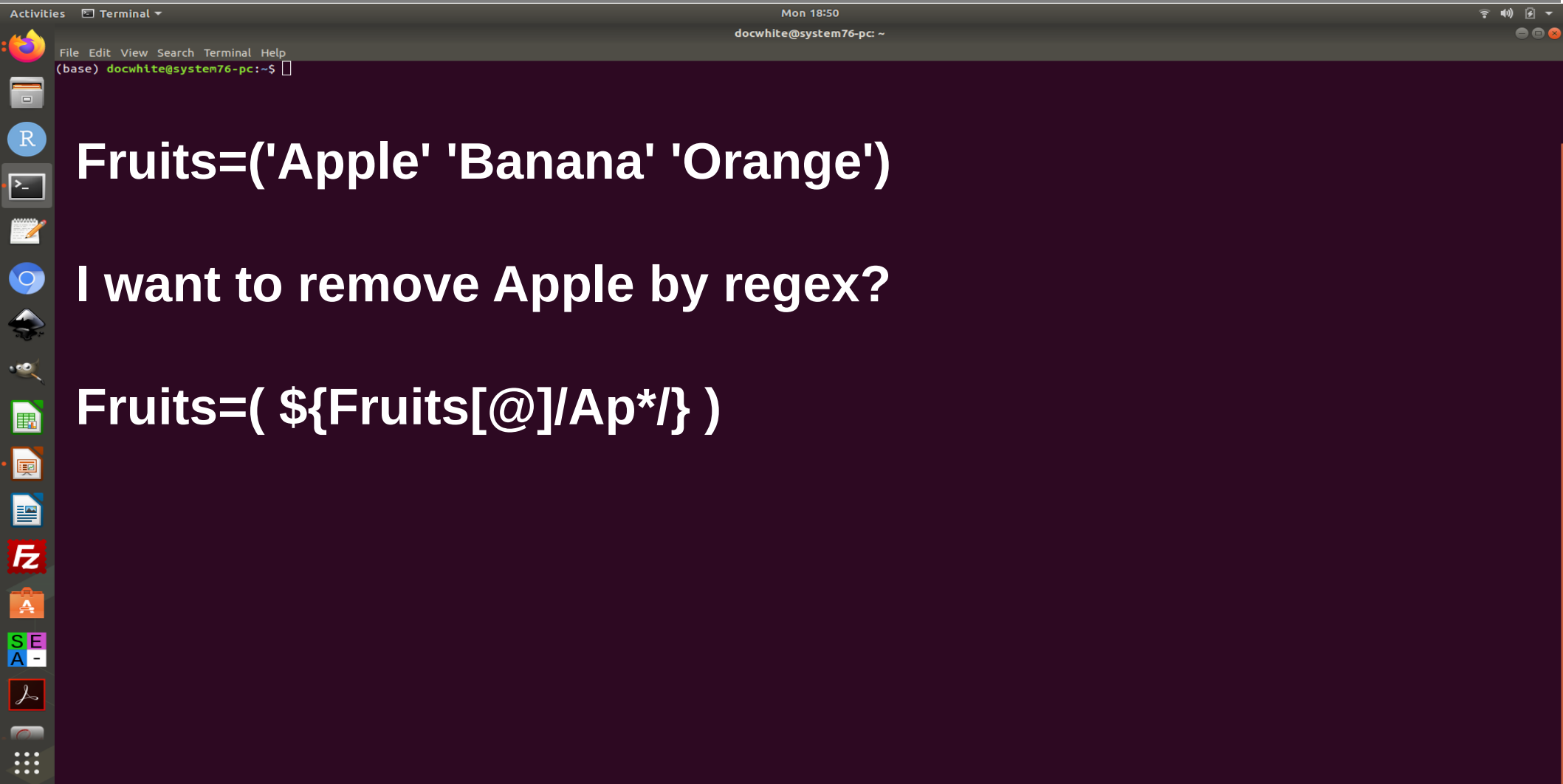
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
I want to add watermelon to the array?
PUSH!
Fruits=("${Fruits[@]}" "Watermelon")
Fruits+=('Watermelon')
echo ${Fruits[3]}

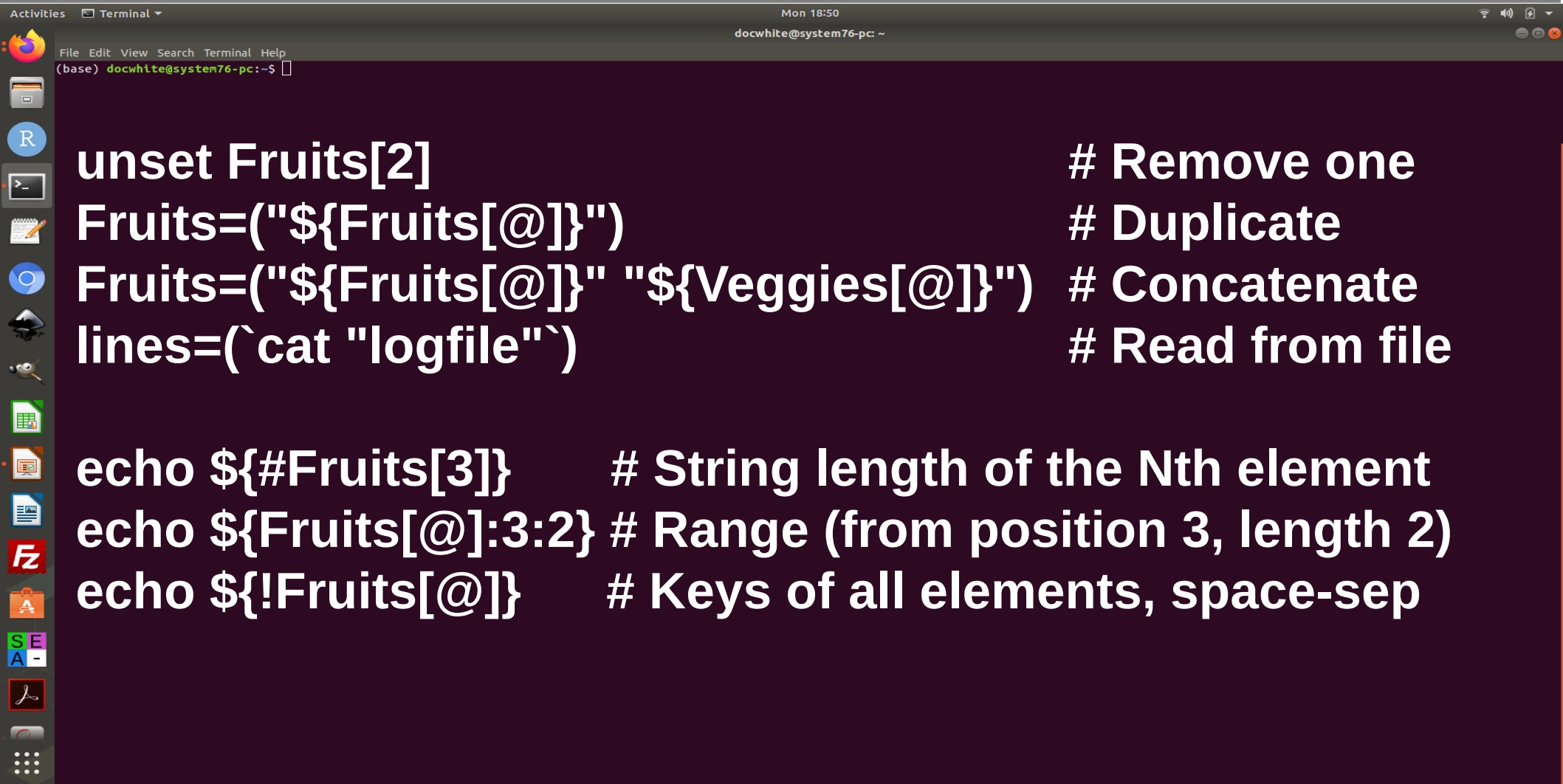
BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
I want to remove Apple by regex?

BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
Fruits=('Apple' 'Banana' 'Orange')
I want to remove Apple by regex?
Fruits=(${Fruits[@]/Ap*/})

BASH arrays


Activities
Terminal
Mon 18:50
docwhite@system76-pc: ~
File Edit View Search Terminal Help
(base) docwhite@system76-pc:~$
unset Fruits[2] # Remove one
Fruits=("${Fruits[@]}") # Duplicate
Fruits=("${Fruits[@]}" "${Veggies[@]}") # Concatenate
lines=(`cat "logfile"`) # Read from file
echo ${#Fruits[3]} # String length of the Nth element
echo ${Fruits[@]:3:2} # Range (from position 3, length 2)
echo ${!Fruits[@]} # Keys of all elements, space-sep

Quiz 8

- On canvas now

Bonus 7

- Write a BASH code to iterate through the Fruit array?

Fruits=('Apple' 'Banana' 'Orange', ‘Mango’)

ENJOY!

Bonus 7

- Write a BASH code to iterate through the Fruit array?

Fruits=('Apple' 'Banana' 'Orange', ‘Mango’)

```
for i in "${Fruits[@]}"; do
  echo $i
done
```