Advanced Linux Usage

2017-11-28

Martin Dahlö
martin.dahlo@scilifelab.uu.se

Valentin Georgiev
valentin.georgiev@icm.uu.se

Jacques Dainat
jacques.dainat@nbis.se

Enabler for Life Sciences

Shell and Bash

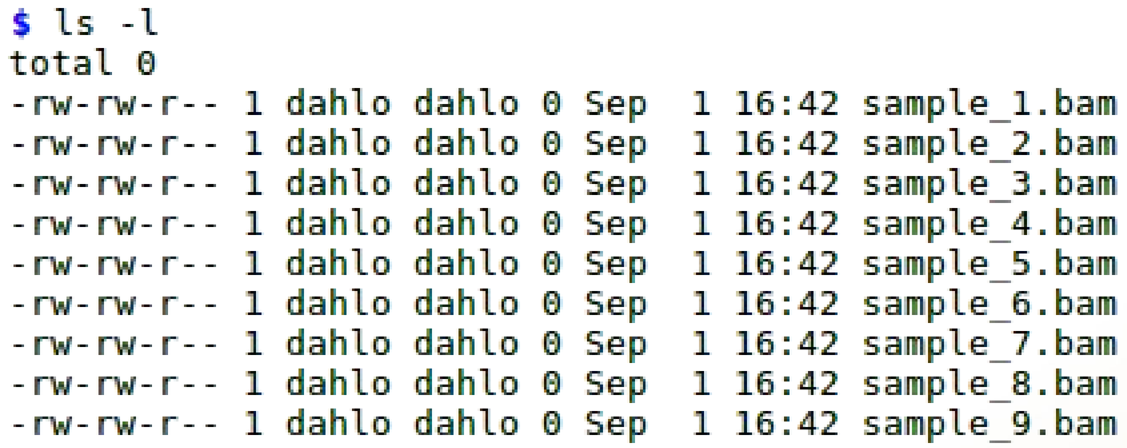
the Shell is a Command Line Interface (CLI)

Bash is one particular shell

tcsh, zsh are also shell programs

Multiple files

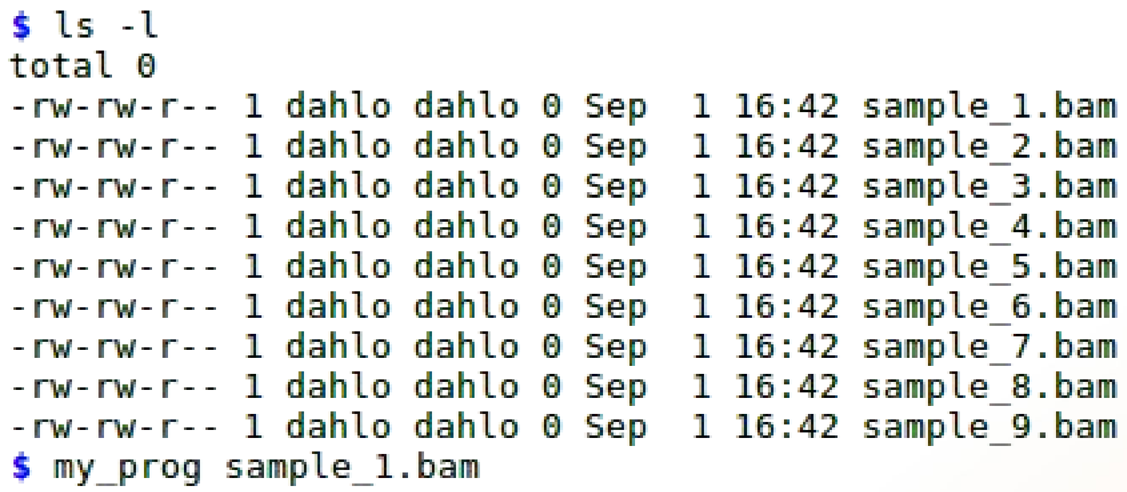
- Same program, many files



```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
```

Multiple files

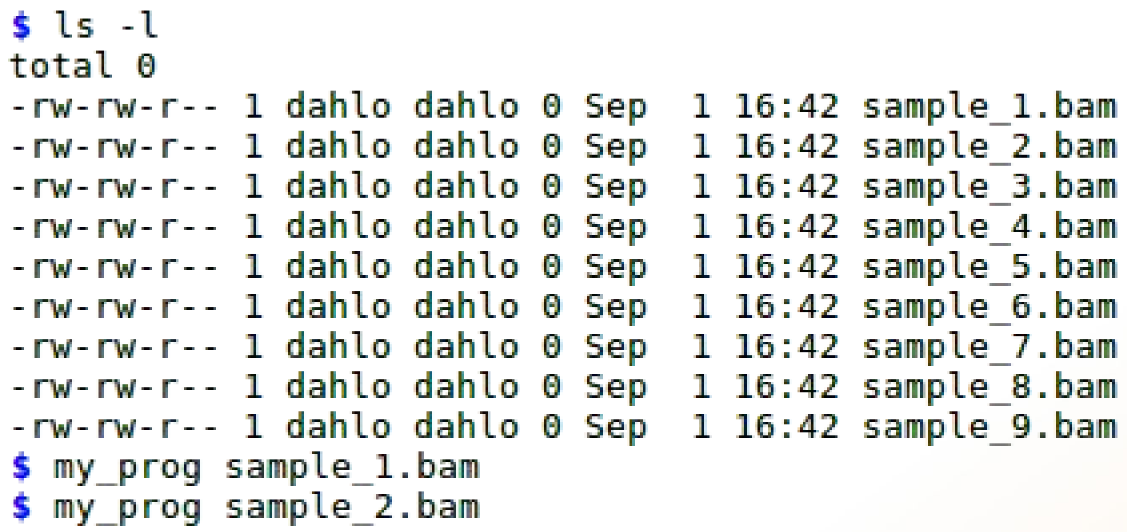
- Same program, many files



```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
```

Multiple files

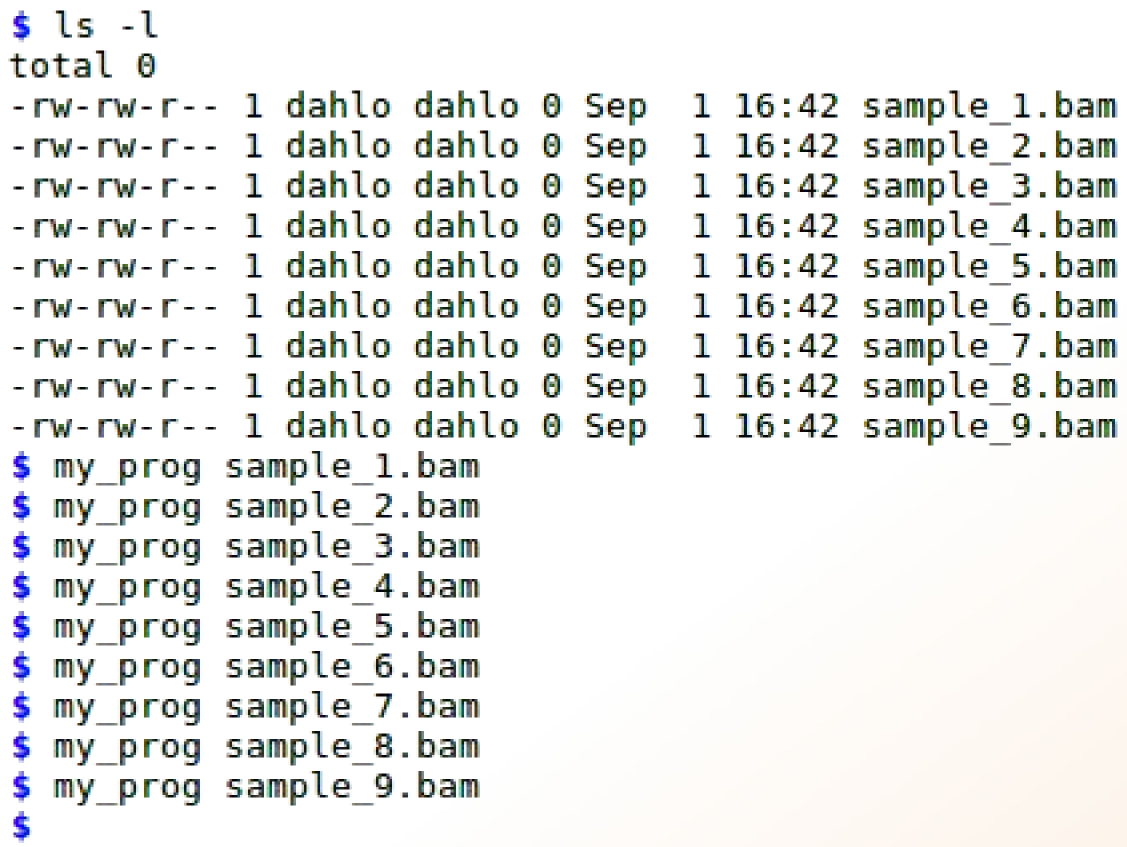
- Same program, many files



```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
```

Multiple files

- Same program, many files



```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
$ my_prog sample_3.bam
$ my_prog sample_4.bam
$ my_prog sample_5.bam
$ my_prog sample_6.bam
$ my_prog sample_7.bam
$ my_prog sample_8.bam
$ my_prog sample_9.bam
$
```

Multiple files

- Same program, many files
  - 10 files? Ok
  - 1000 files? Not ok..

Multiple files

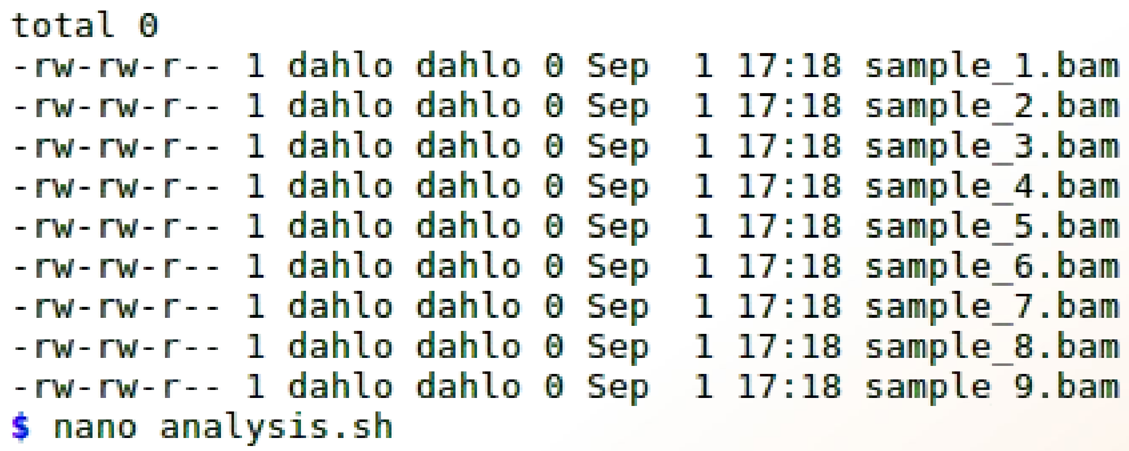
- Same program, many files
  - 10 files? Ok
  - 1000 files? Not ok..
- Reproducibility
  - Self and others

Multiple files

- Same program, many files
  - 10 files? Ok
  - 1000 files? Not ok..
- Reproducibility
  - Self and others

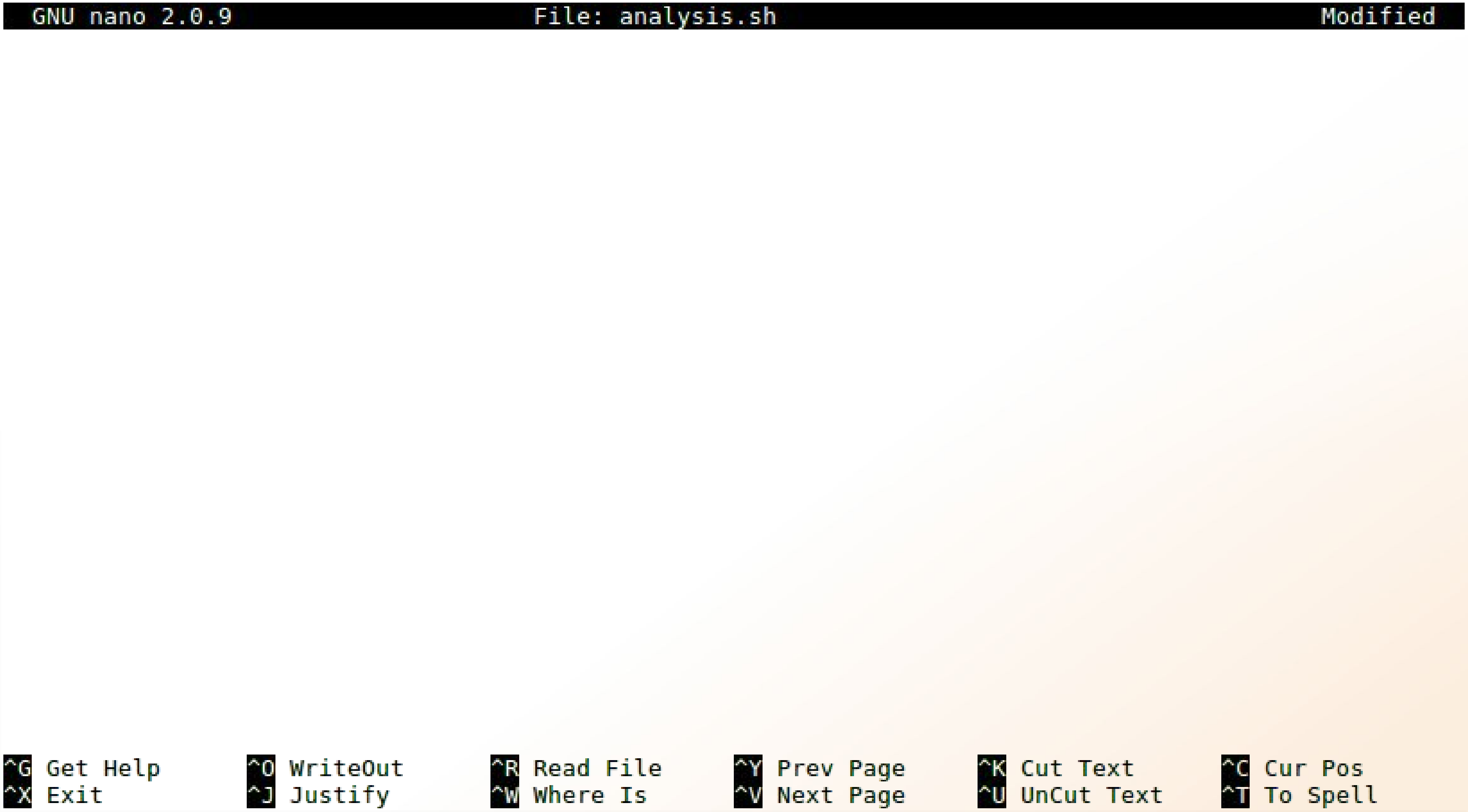
The answer - write a script!

Basic script



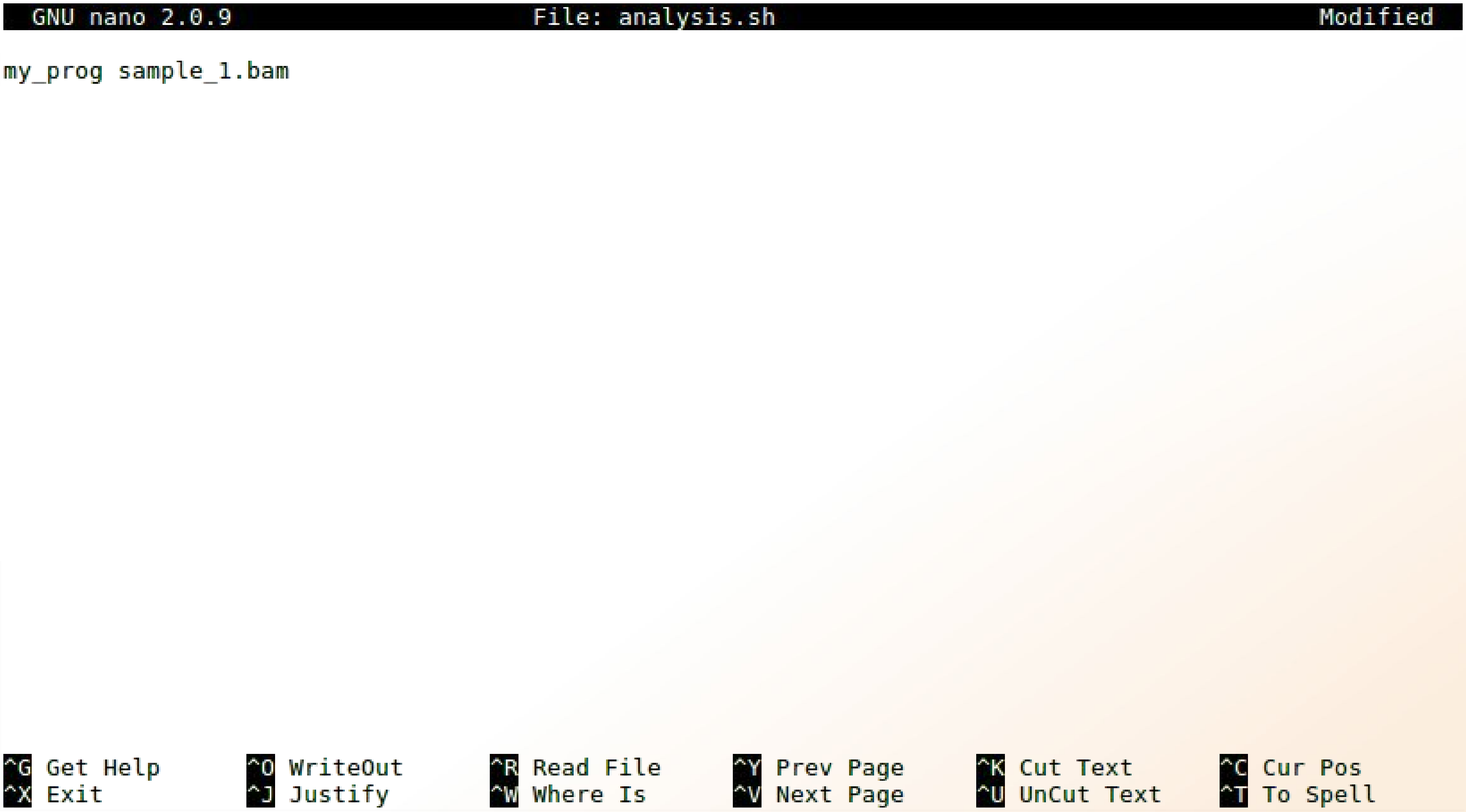
```
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_9.bam
$ nano analysis.sh
```

Basic script



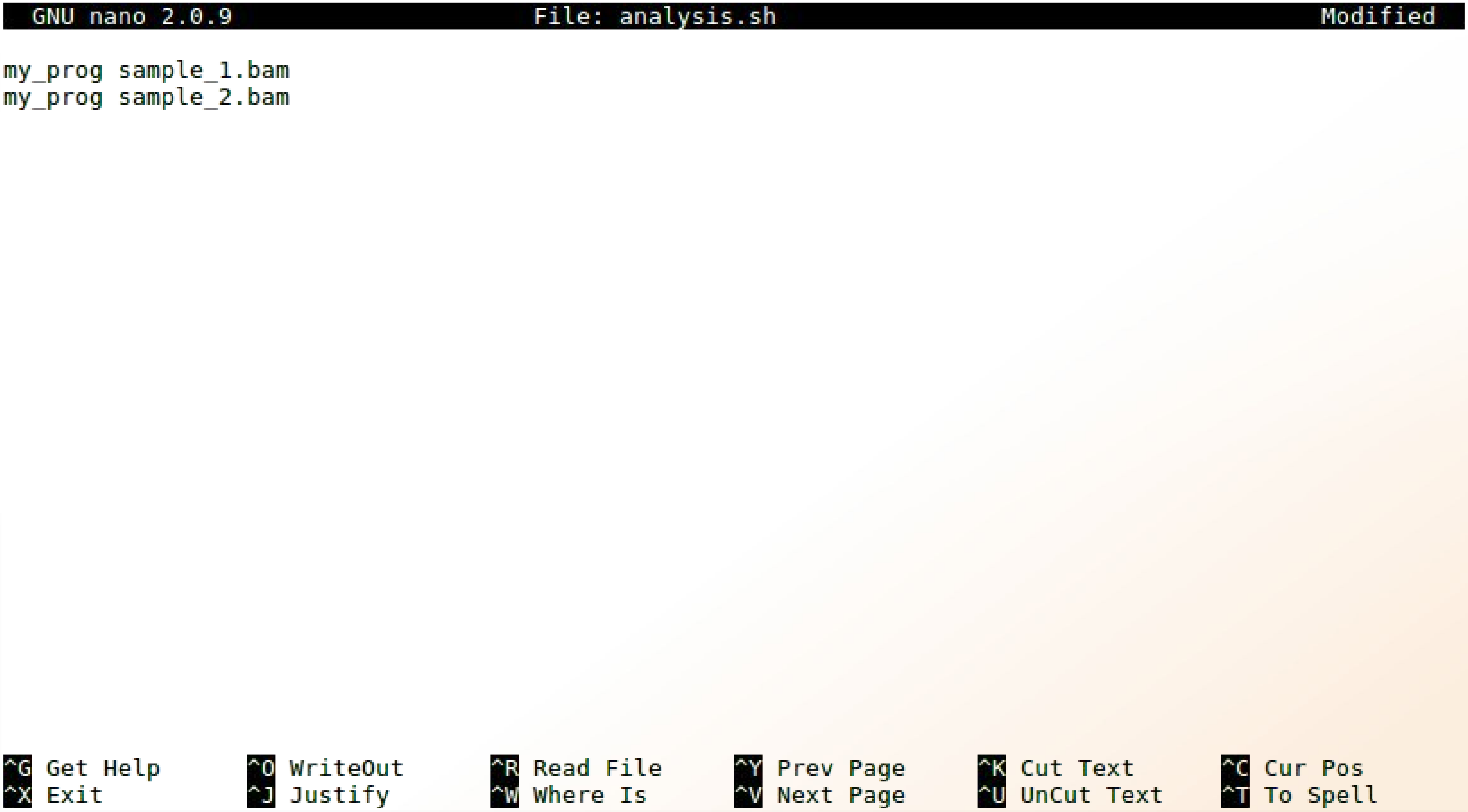
```
  GNU nano 2.0.9                 File: analysis.sh                                    Modified
















^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is      ^V Next Page     ^U UnCut Text    ^T To Spell
```

Basic script



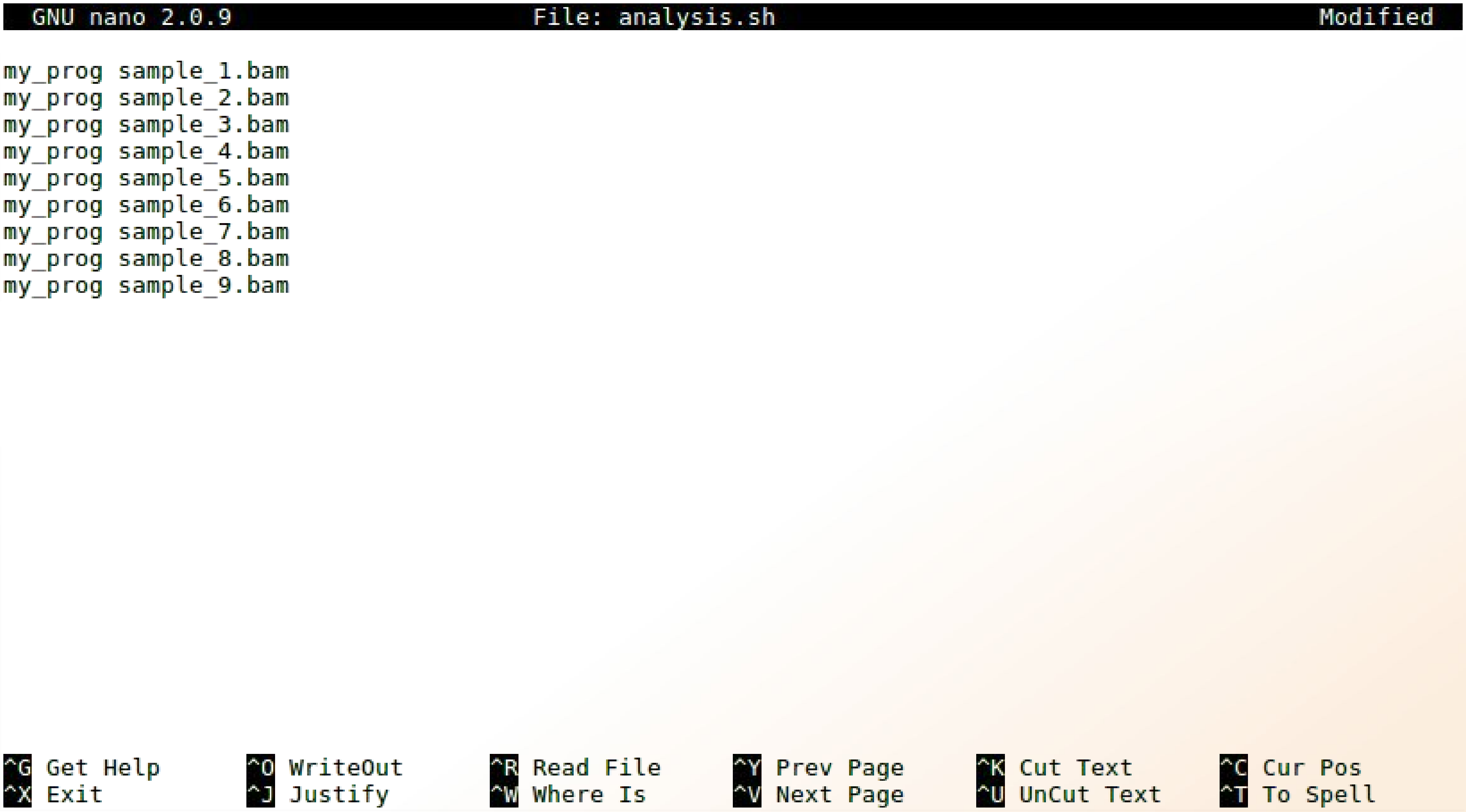
```
  GNU nano 2.0.9                 File: analysis.sh                              Modified

my_prog sample_1.bam
```

Basic script



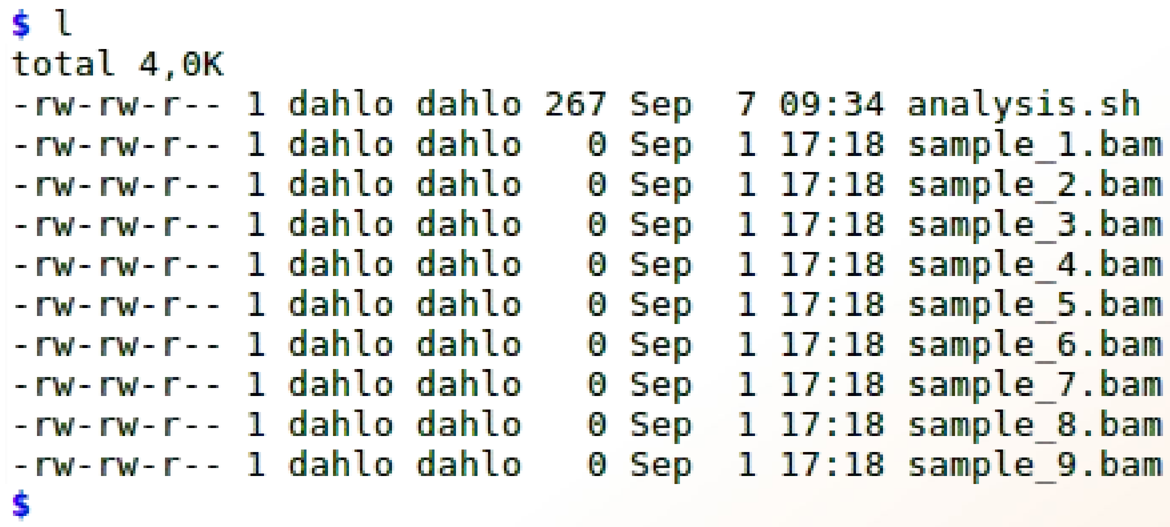
```
  GNU nano 2.0.9                File: analysis.sh                              Modified

my_prog sample_1.bam
my_prog sample_2.bam
```

Basic script



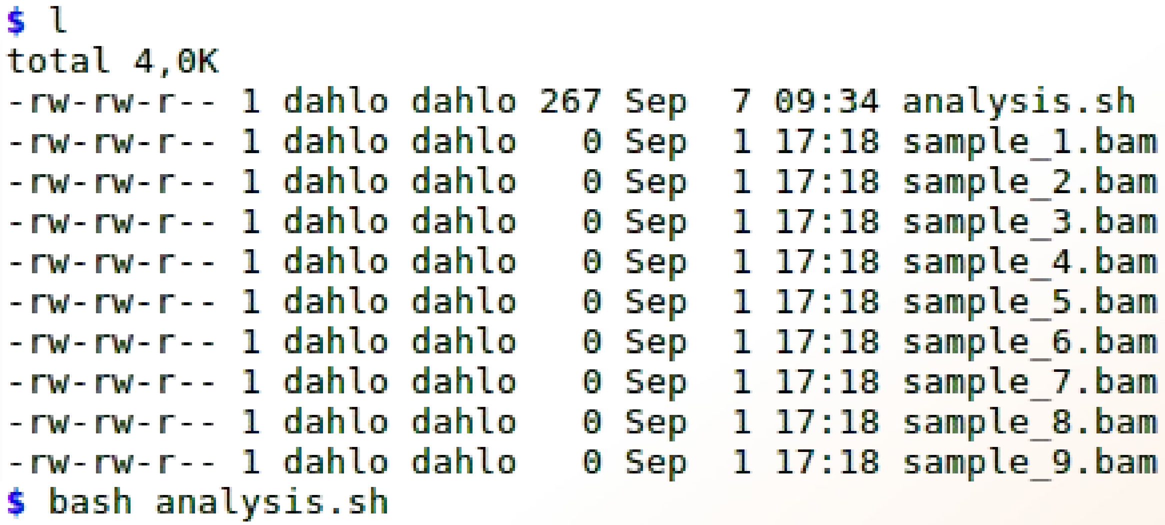
```
  GNU nano 2.0.9                   File: analysis.sh                              Modified

my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
```

Basic script



```
$ l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep  7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_9.bam
$
```

Basic script



```
$ l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep  7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_9.bam
$ bash analysis.sh
```

Variables

- Assigning

```
my_variable=5
my_variable=”nice text”
```

Variables

- Assigning

  ```
  my_variable=5
  my_variable=”nice text”
  ```

- Using

  ```
  $my_variable
  ```

Variables

- Assigning

```
my_variable=5
my_variable=”nice text”
```

- Using

```
$my_variable
```

```
$ my_variable=”Pia”
```

Variables

- Assigning

```
my_variable=5
my_variable="nice text"
```

- Using

```
$my_variable
```

```
$ my_variable="Pia"
$ echo "Hello, $my_variable! "
```

Variables

- Assigning

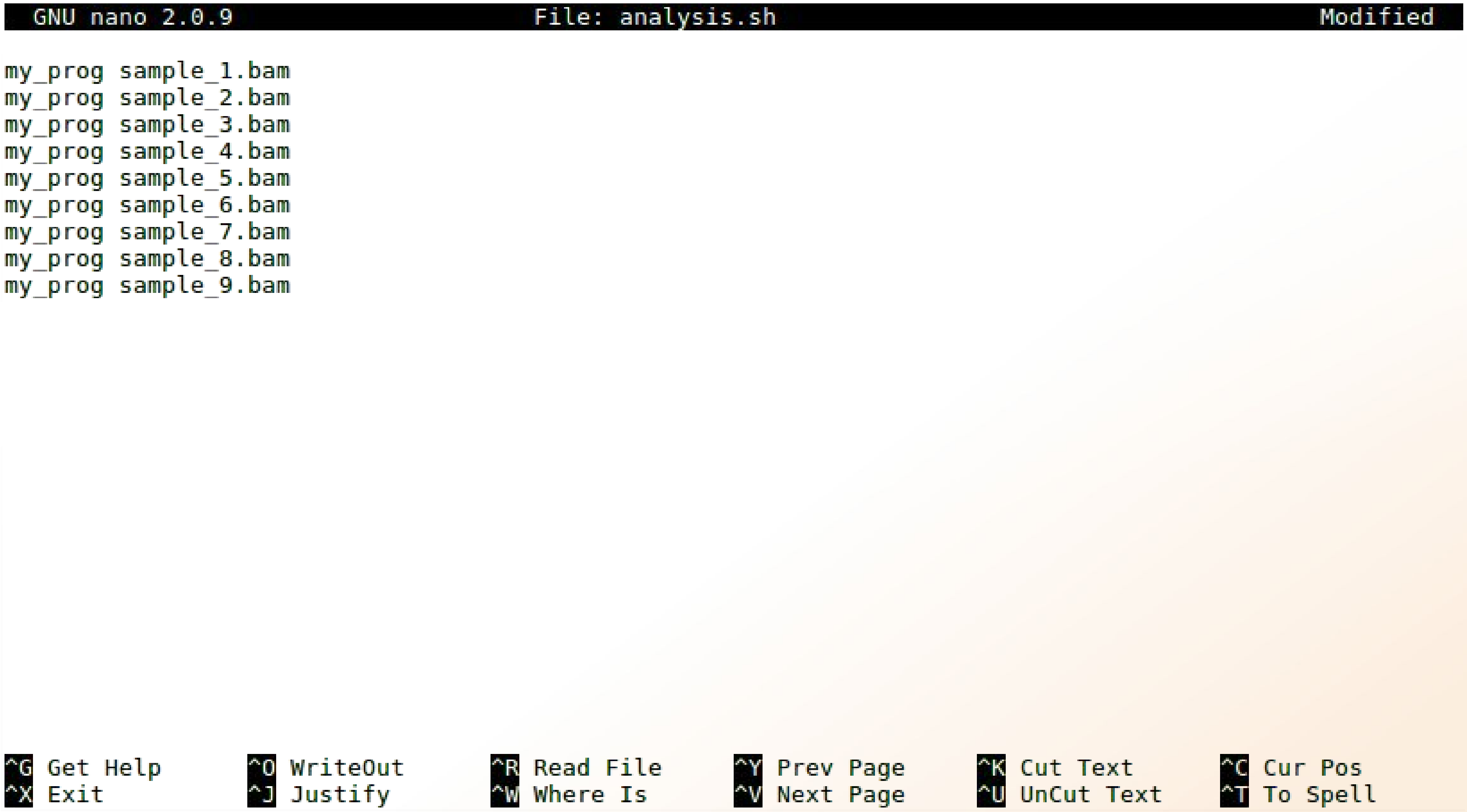
```
my_variable=5
my_variable=”nice text”
```

- Using

```
$my_variable
```

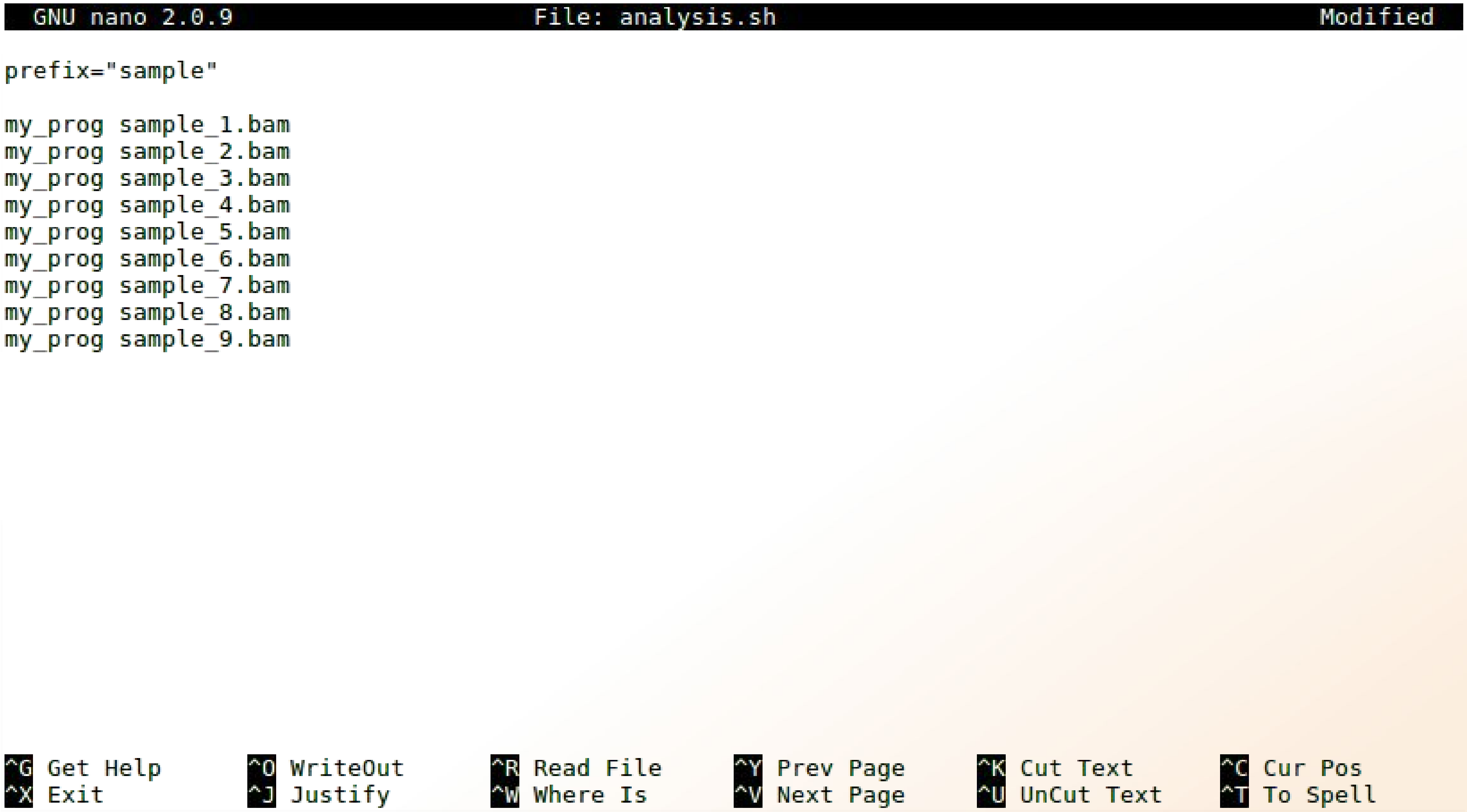
```
$ my_variable=”Pia”
$ echo “Hello, $my_variable! ”
Hello, Pia!
```

Variables



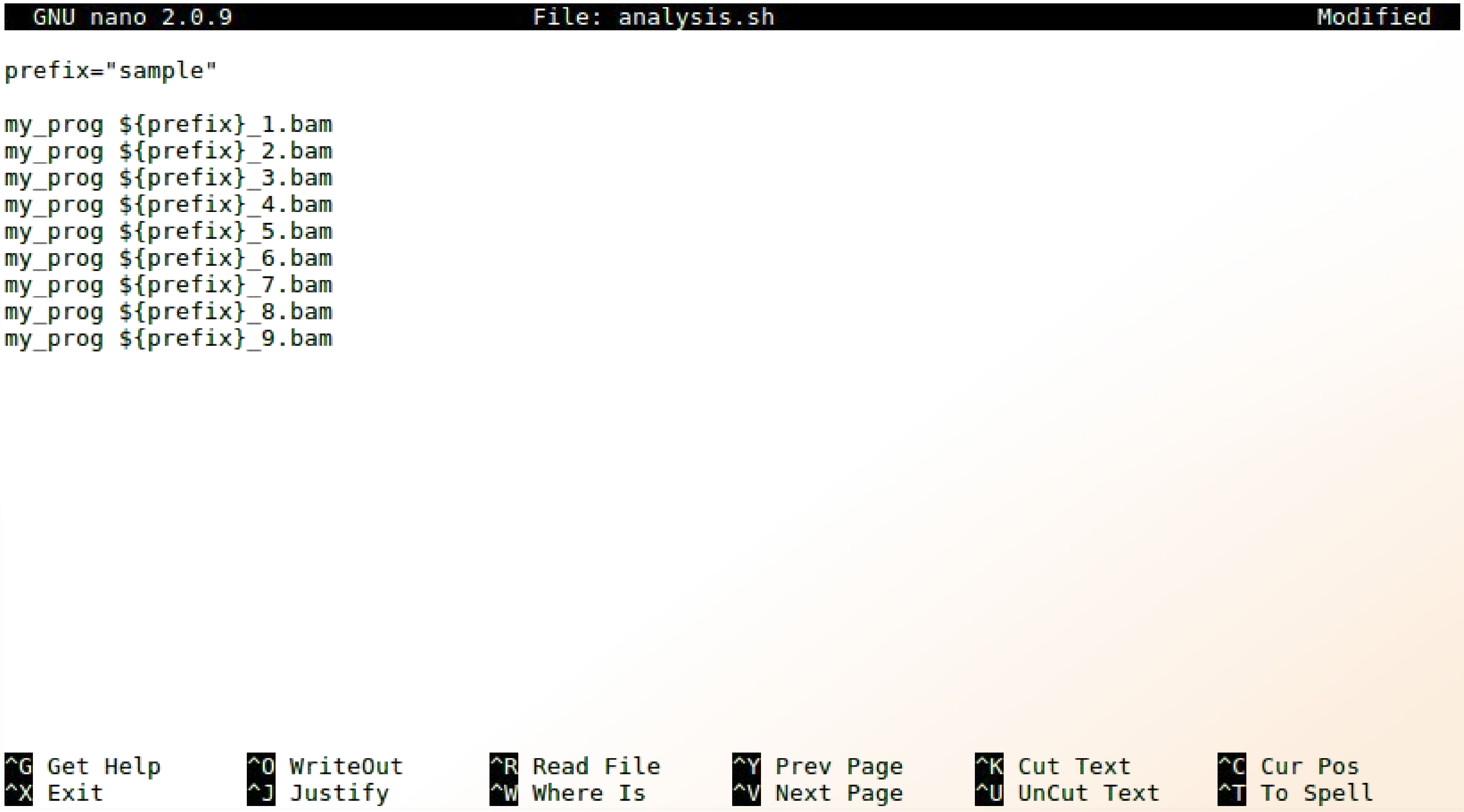
```
  GNU nano 2.0.9              File: analysis.sh                                Modified

my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
```

Variables



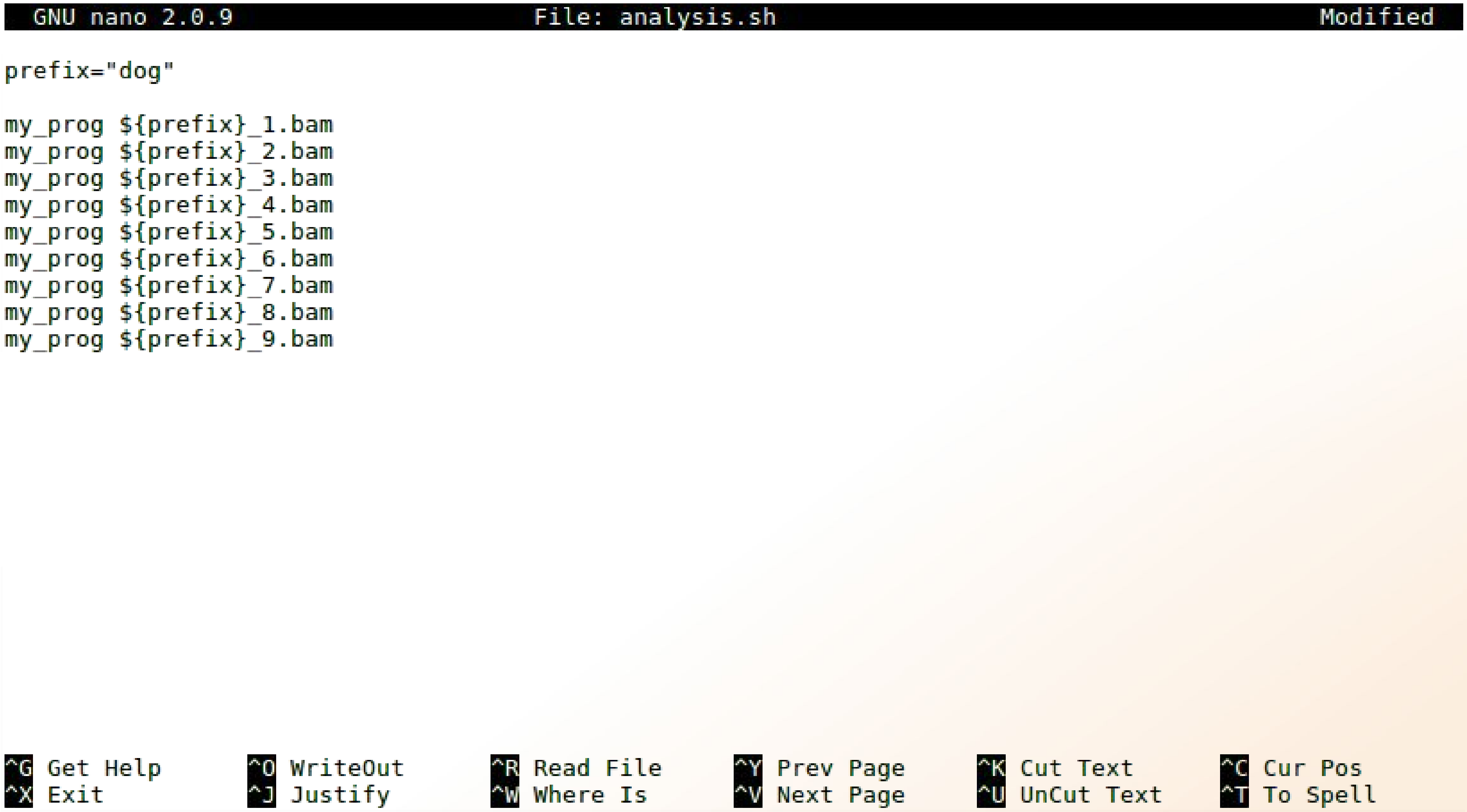
```
  GNU nano 2.0.9                File: analysis.sh                                Modified

prefix="sample"

my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
```

Variables



```
  GNU nano 2.0.9                File: analysis.sh                          Modified

prefix="sample"

my_prog ${prefix}_1.bam
my_prog ${prefix}_2.bam
my_prog ${prefix}_3.bam
my_prog ${prefix}_4.bam
my_prog ${prefix}_5.bam
my_prog ${prefix}_6.bam
my_prog ${prefix}_7.bam
my_prog ${prefix}_8.bam
my_prog ${prefix}_9.bam
```

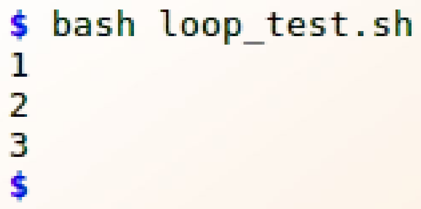
Variables



```
  GNU nano 2.0.9                   File: analysis.sh                                          Modified

prefix="dog"

my_prog ${prefix}_1.bam
my_prog ${prefix}_2.bam
my_prog ${prefix}_3.bam
my_prog ${prefix}_4.bam
my_prog ${prefix}_5.bam
my_prog ${prefix}_6.bam
my_prog ${prefix}_7.bam
my_prog ${prefix}_8.bam
my_prog ${prefix}_9.bam
```

Loops

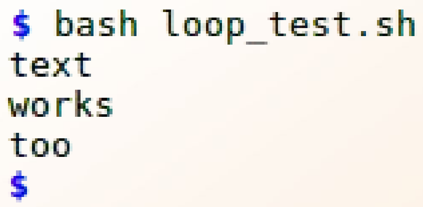
```
for variable in 1 2 3;
do
    echo $variable
done
```



```
$ bash loop_test.sh
1
2
3
$
```

Loops

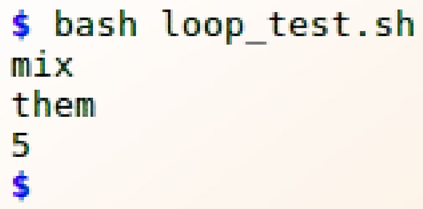
```
for variable in text works too;
do
    echo $variable
done
```



```
$ bash loop_test.sh
text
works
too
$
```

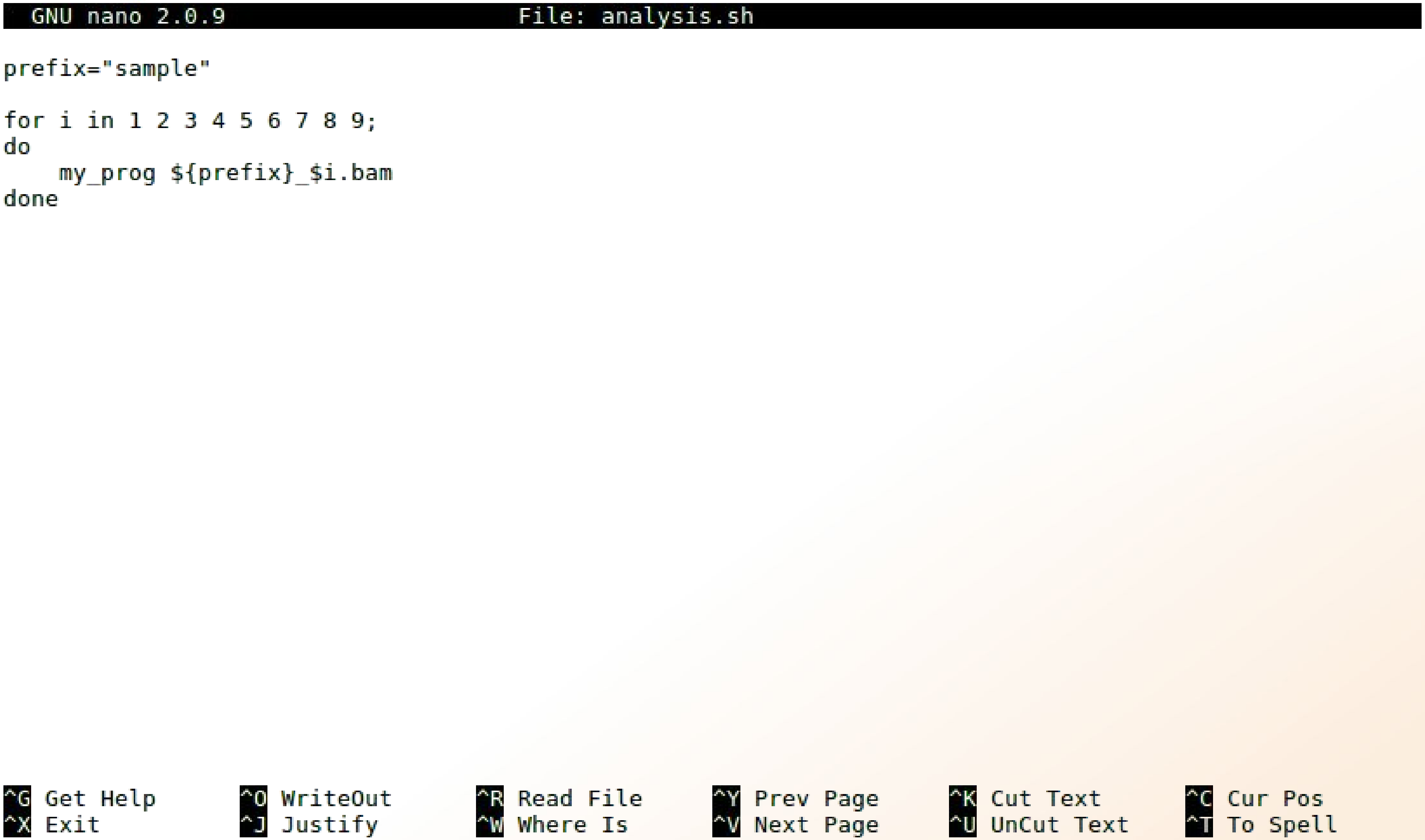
Loops

```
for variable in mix them 5;
do
    echo $variable
done
```



```
$ bash loop_test.sh
mix
them
5
$
```

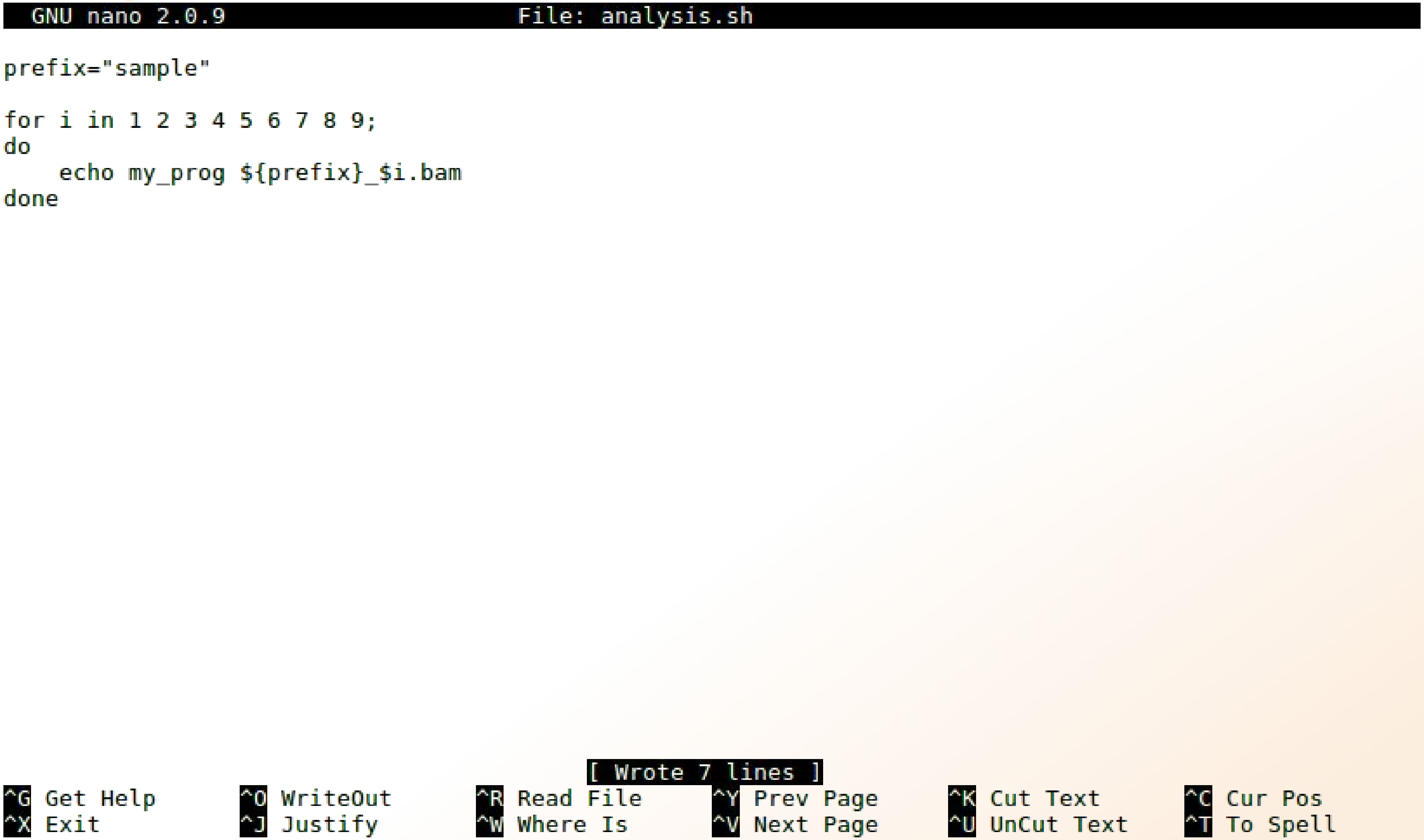
Loops



```
  GNU nano 2.0.9                    File: analysis.sh

prefix="sample"

for i in 1 2 3 4 5 6 7 8 9;
do
    my_prog ${prefix}_$i.bam
done
```

```
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

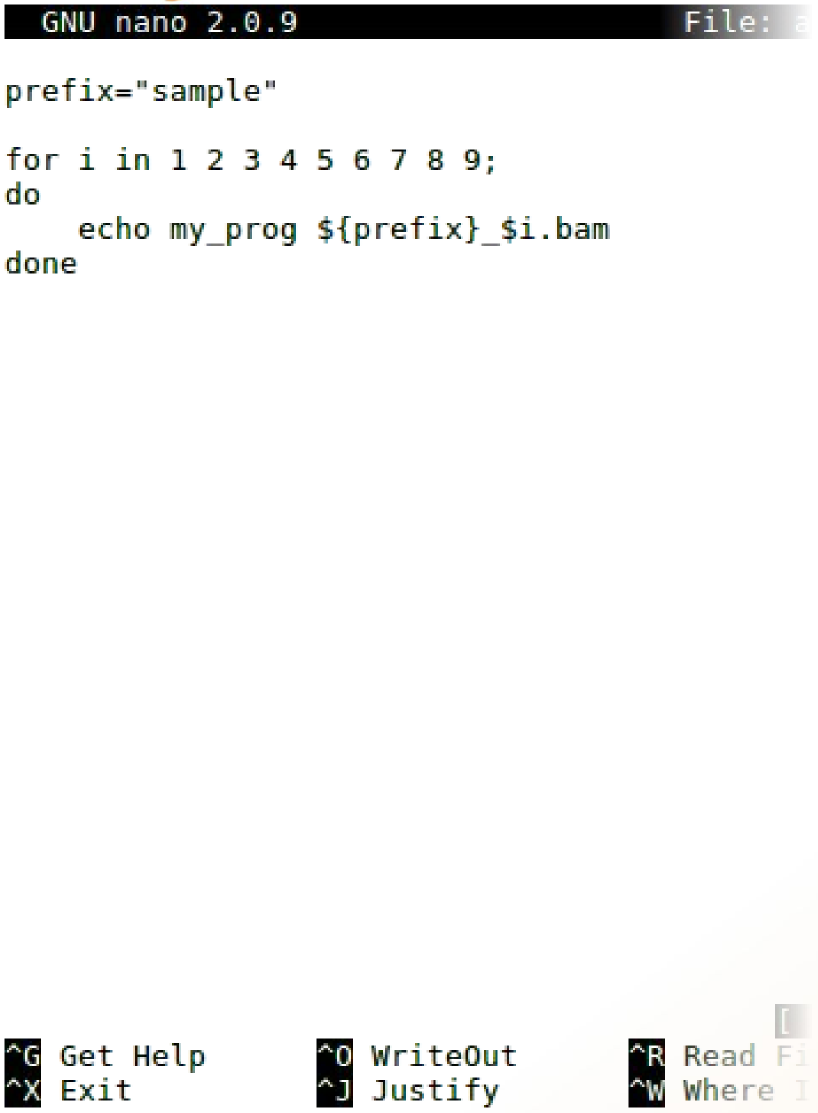
Loops



```
  GNU nano 2.0.9                    File: analysis.sh

prefix="sample"

for i in 1 2 3 4 5 6 7 8 9;
do
    echo my_prog ${prefix}_$i.bam
done
```

```
                                          [ Wrote 7 lines ]
^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is      ^V Next Page     ^U UnCut Text    ^T To Spell
```

Loops

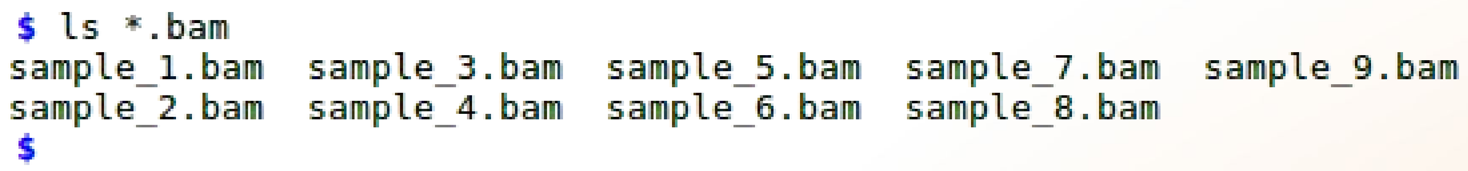


```
  GNU nano 2.0.9                      File: a

prefix="sample"

for i in 1 2 3 4 5 6 7 8 9;
do
    echo my_prog ${prefix}_$i.bam
done
```

```
                                                  [
^G Get Help        ^O WriteOut        ^R Read Fi
^X Exit            ^J Justify         ^W Where I
```

```
$ bash analysis.sh
my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
$
```

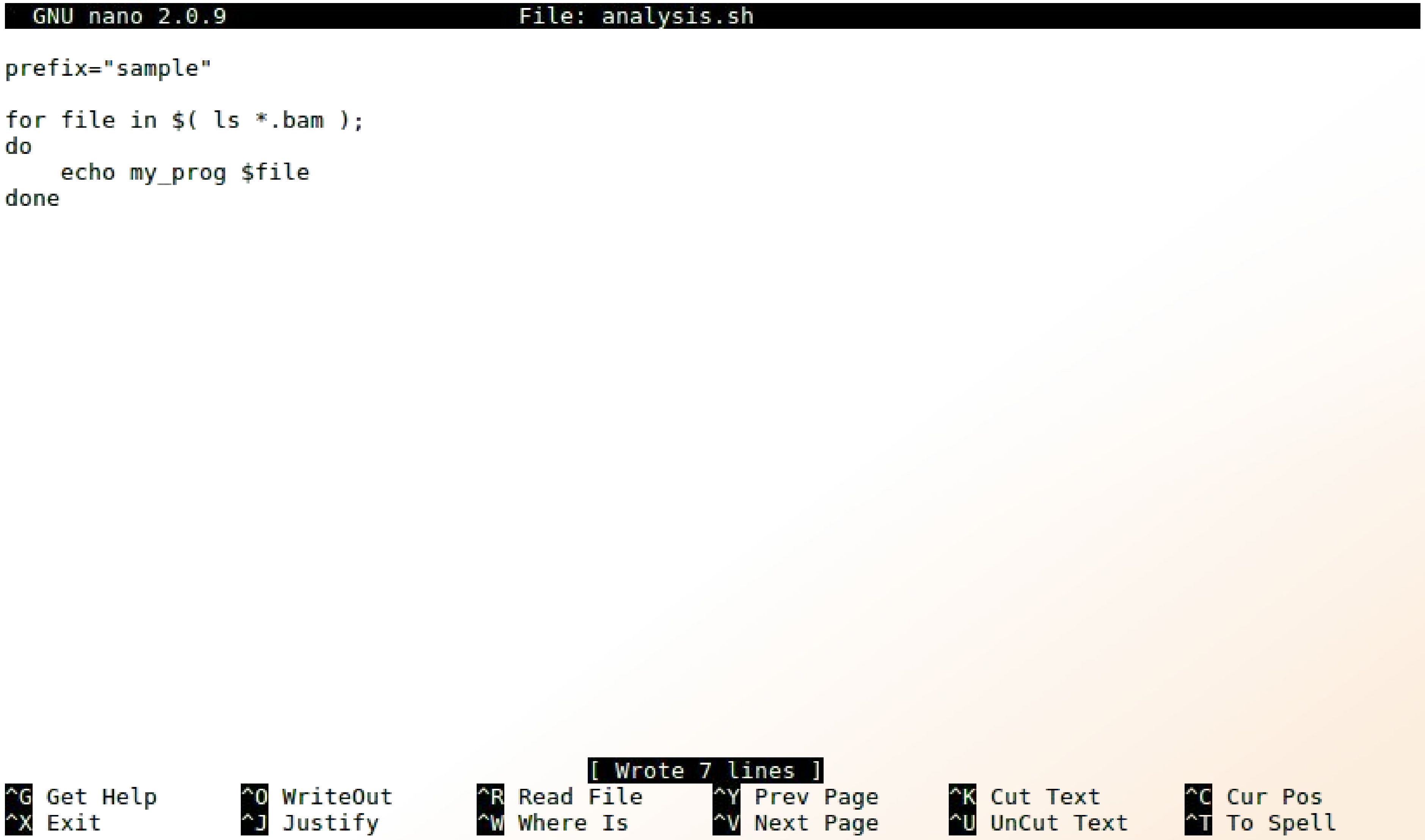
Loop over files



```
$ ls *.bam
sample_1.bam  sample_3.bam  sample_5.bam  sample_7.bam  sample_9.bam
sample_2.bam  sample_4.bam  sample_6.bam  sample_8.bam
$
```

Wildcard *

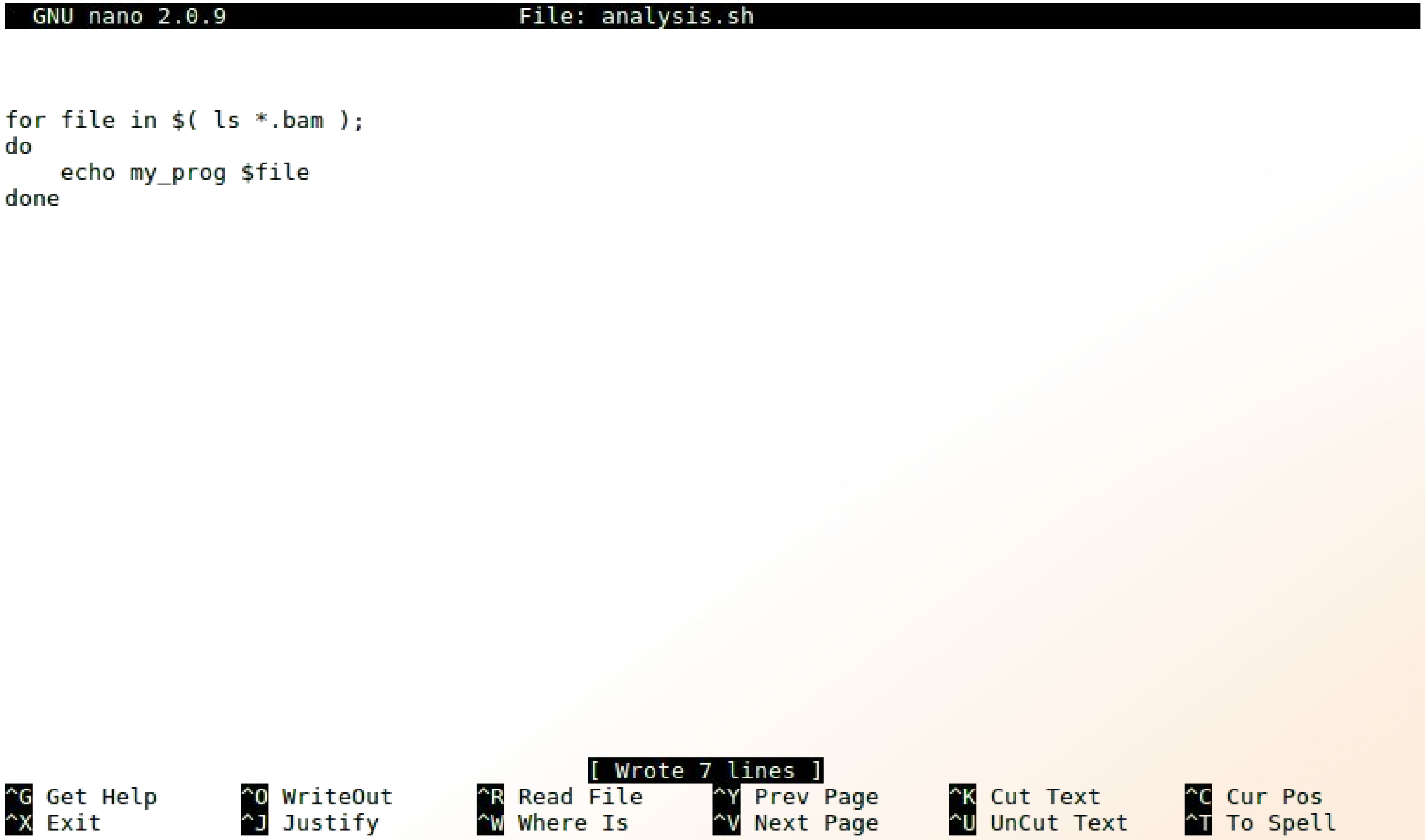
Loop over files



```
  GNU nano 2.0.9                    File: analysis.sh

prefix="sample"

for file in $( ls *.bam );
do
    echo my_prog $file
done
```

```
                                          [ Wrote 7 lines ]
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

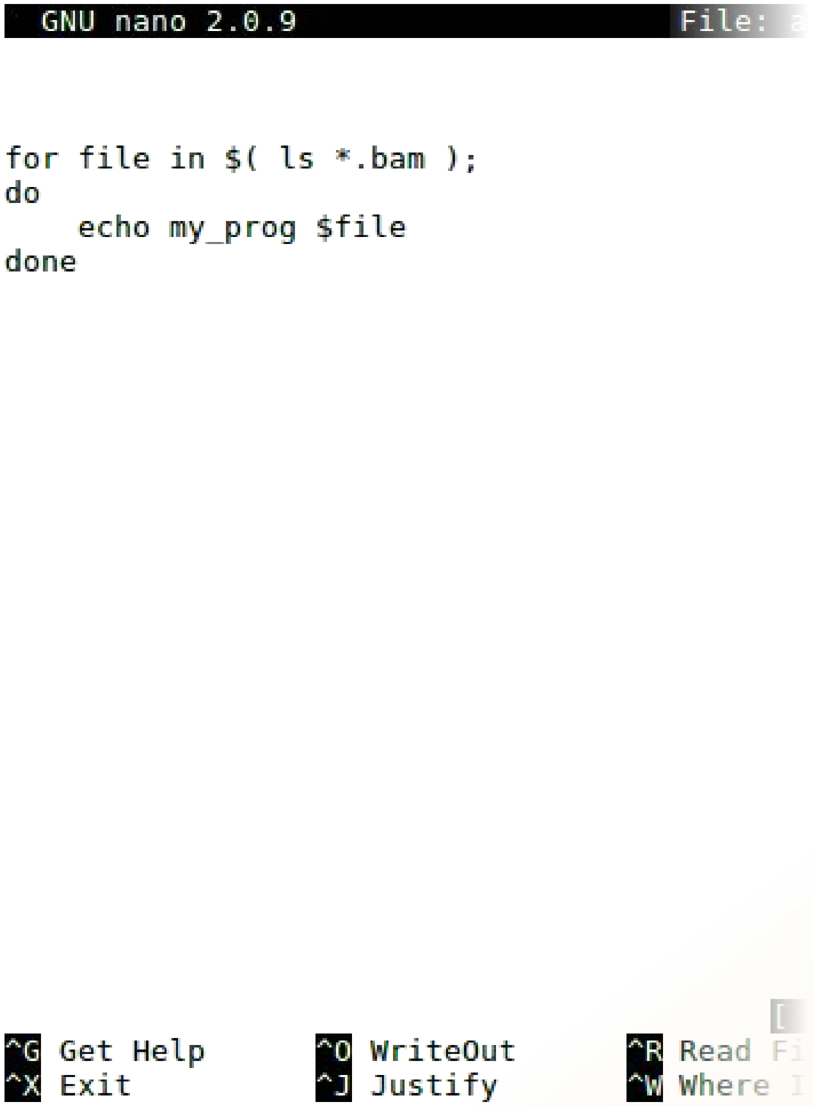
Loop over files



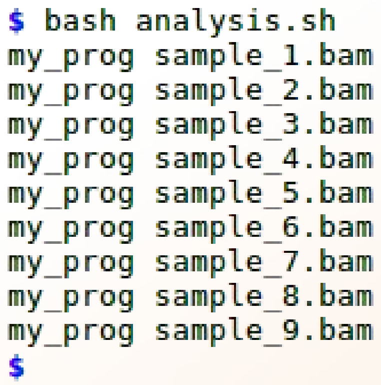
```
  GNU nano 2.0.9                    File: analysis.sh


for file in $( ls *.bam );
do
    echo my_prog $file
done
```

```
                                          [ Wrote 7 lines ]
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

Loop over files

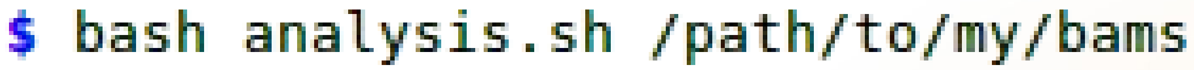


```
 GNU nano 2.0.9                    File: a


for file in $( ls *.bam );
do
    echo my_prog $file
done




















                                      [
^G Get Help      ^O WriteOut      ^R Read Fi
^X Exit          ^J Justify       ^W Where I
```



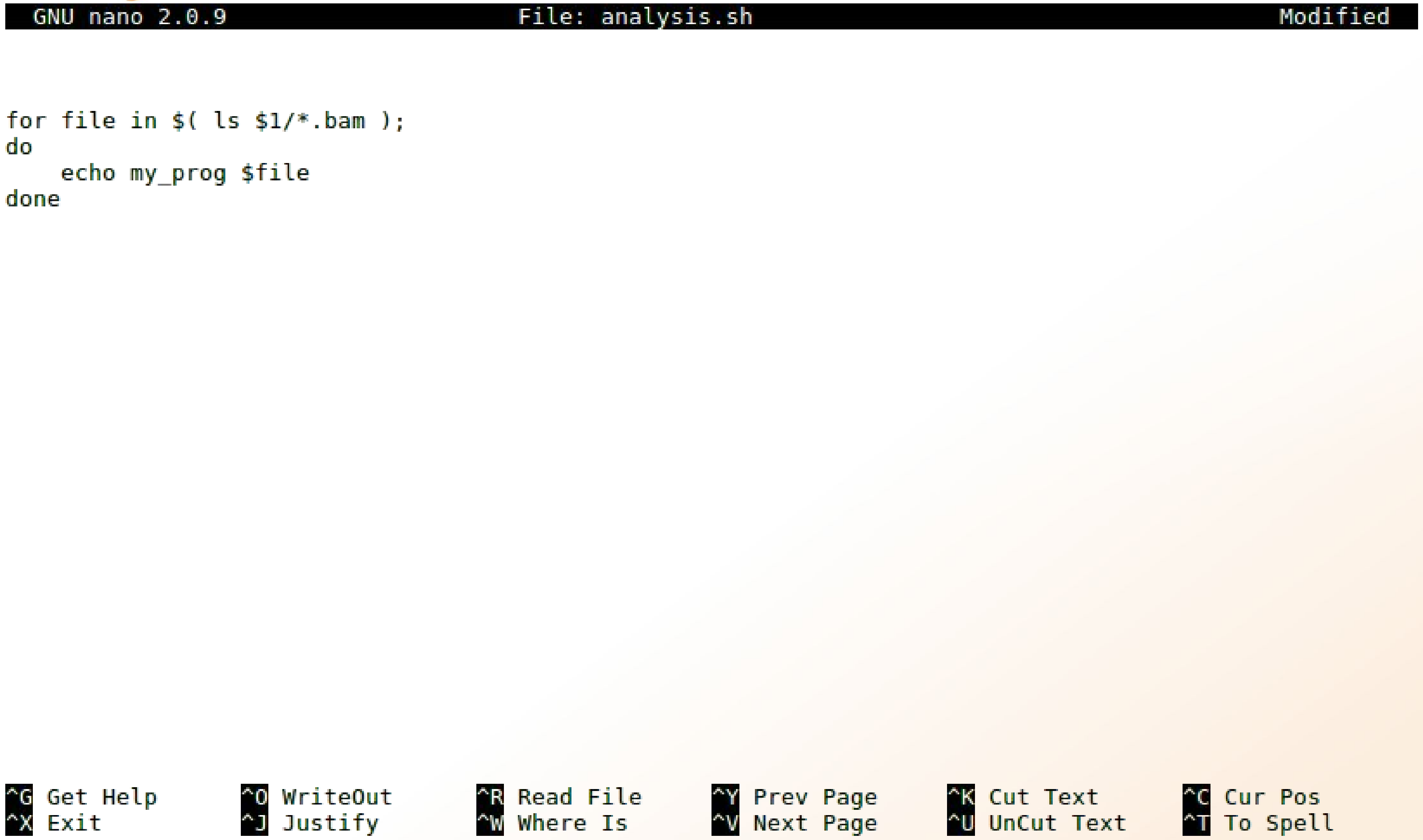
```
$ bash analysis.sh
my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam
$
```

Loop over files



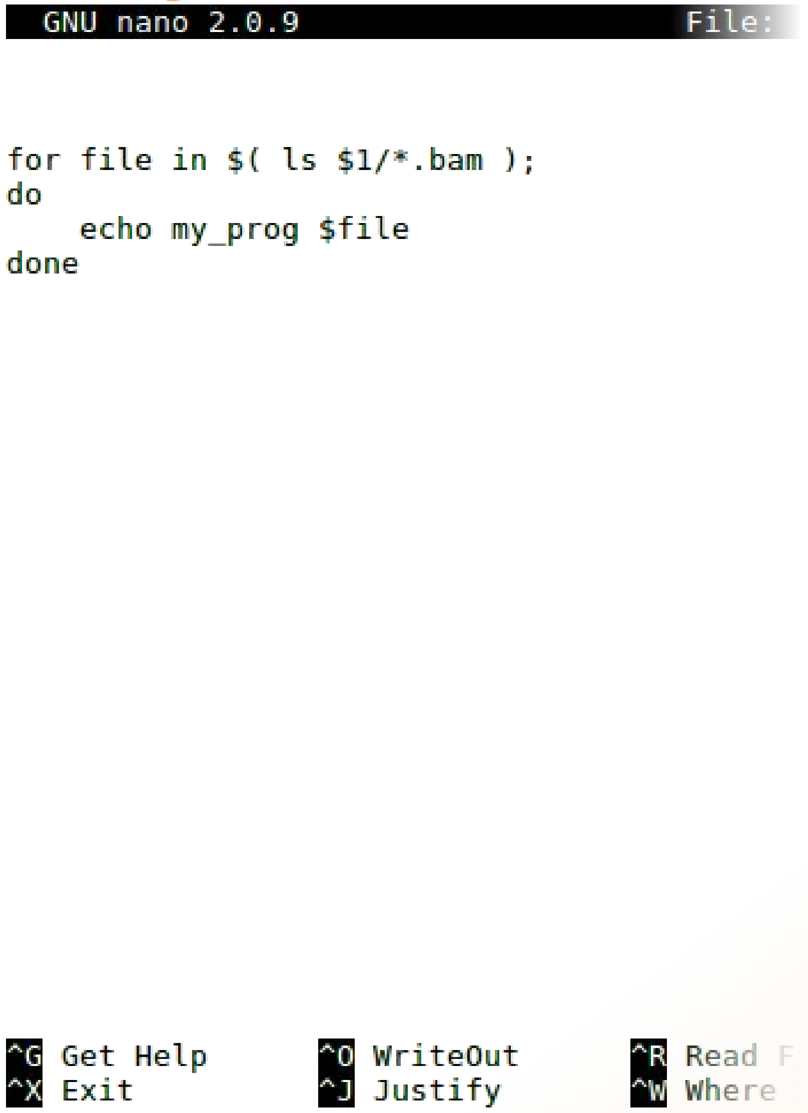
```
$ bash analysis.sh /path/to/my/bams
```

Loop over files



```
  GNU nano 2.0.9                File: analysis.sh                        Modified


for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```

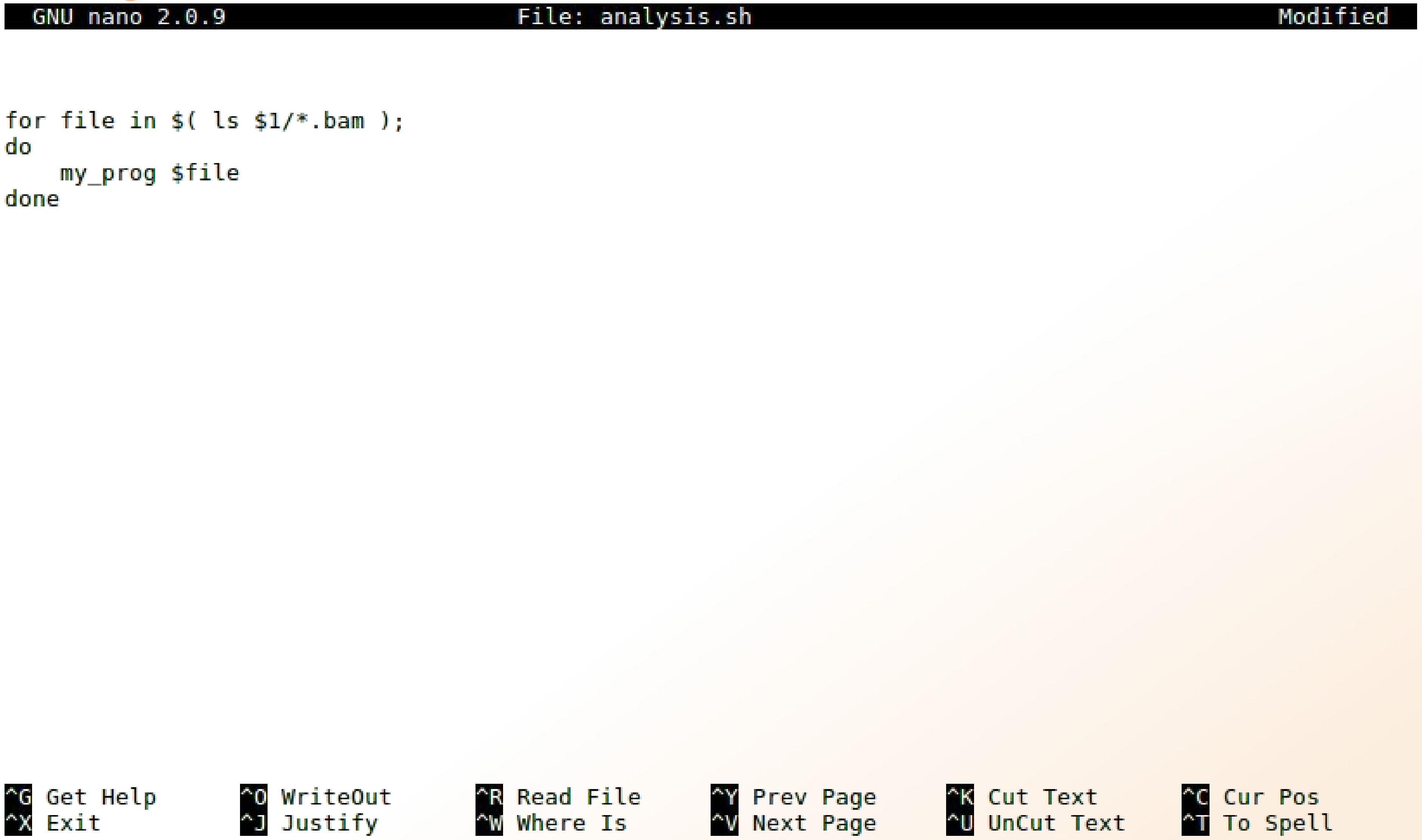
Loop over files



```
for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```

```
$ bash analysis.sh /path/to/my/bams
my_prog /path/to/my/bams/sample_1.bam
my_prog /path/to/my/bams/sample_2.bam
my_prog /path/to/my/bams/sample_3.bam
my_prog /path/to/my/bams/sample_4.bam
my_prog /path/to/my/bams/sample_5.bam
my_prog /path/to/my/bams/sample_6.bam
my_prog /path/to/my/bams/sample_7.bam
my_prog /path/to/my/bams/sample_8.bam
my_prog /path/to/my/bams/sample_9.bam
$
```

Loop over files



```
  GNU nano 2.0.9                 File: analysis.sh                          Modified


for file in $( ls $1/*.bam );
do
    my_prog $file
done
```

```
^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is      ^V Next Page     ^U UnCut Text    ^T To Spell
```

If

- Control statement

```
if condition; then
  action
fi
```

If

- Control statement

```
if true; then
  echo ”This is true”
fi
```

result:

```
 This is true
```

If

- Control statement

```
if false; then
  echo ”This is true”
fi
```

result:

If

- Control statement

```
if [[ 5 < 9 ]]; then
  echo ”This is true”
fi
```

result:

```
This is true
```

If

- Control statement

```
if [[ 5 > 9 ]]; then
  echo "This is true"
fi
```

result:

If

- Control statement

```
if [[ 5 == 9 ]]; then
  echo ”This is true”
fi
```

result:

If

- Control statement

```
if [[ ”Hello” == ”Hello” ]]; then
  echo ”This is true”
fi
```

result:

```
This is true
```

If

- Control statement

```
if [[ ”Hello” == ”Hi” ]]; then
  echo ”This is true”
fi
```

result:

If

- Control statement

```
if [[ ″Hello″ == ″Hel″* ]]; then
  echo ″This is true″
fi
```

result:

```
This is true
```

If

- For all samples except dog

```
for file in $( ls $1/*.bam );
do
    echo my_prog $file
done
```

If

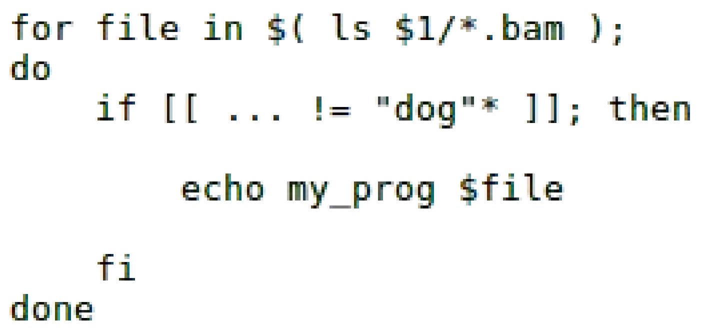
- For all samples except dog

```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

SciLifeLab UPPSALA

If

- For all samples except dog

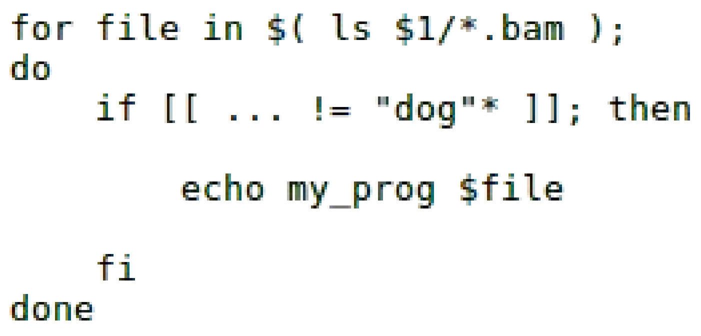


```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

If

- For all samples except dog



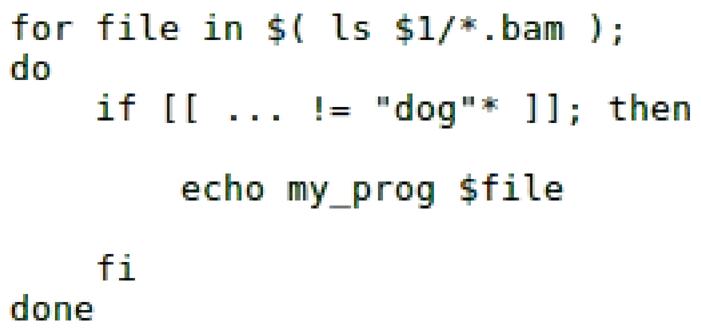
```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

```
basename $file
```

If

- For all samples except dog



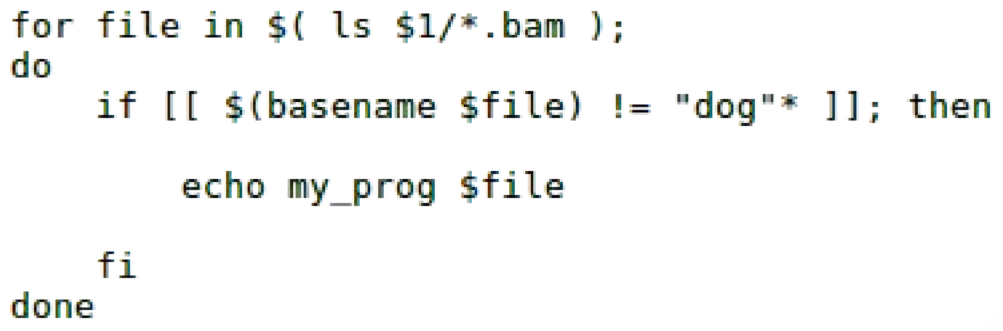
```
for file in $( ls $1/*.bam );
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

```
basename $file

dog_1.bam
```

If

- For all samples except dog



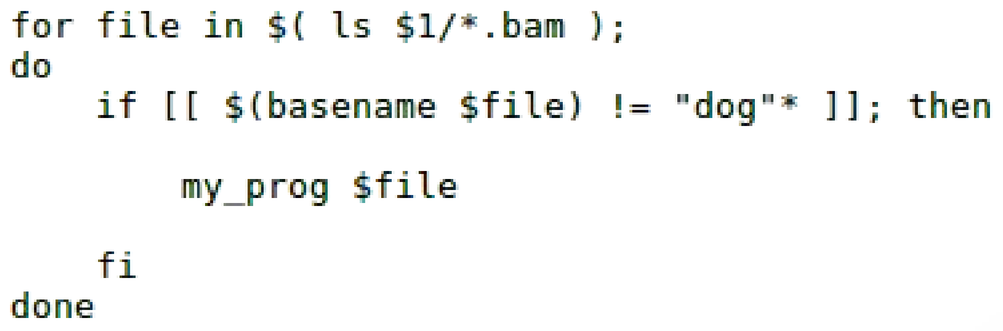
```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

```
basename $file

dog_1.bam
```

If

- For all samples except dog



```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

```
basename $file

dog_1.bam
```

Different languages

- Programming is programming
  - Perl, Python, Bash, and more

Different languages

- Programming is programming
  - Perl, Python, **Bash**, and more

```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

Different languages

- Programming is programming
  - **Perl**, Python, Bash, and more

```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

```
use strict;
use warnings;
use File::Basename;

foreach my $file (glob("$ARGV[0]/*.bam")) {

    if(basename($file) !~ "^dog.+"){

        system("my_prog", $file);
    }
}
```

SciLifeLab UPPSALA

Different languages

- Programming is programming
  - Perl, **Python**, Bash, and more

```
for file in $( ls $1/*.bam );
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

```
import glob
import sys
import subprocess
import os

for file in glob.glob( sys.argv[1] + "/*.bam" ):

    if not os.path.basename(file).startswith("dog"):

        subprocess.call( ["my_prog" , file] )
```

Different languages

- Programming is programming
  - Perl, Python, Bash, and more

- Start with one, git gud, (learn another)

Different languages

- Programming is programming
  - Perl, Python, Bash, and more
- Start with one, git gud, (learn another)

Laboratory time! (yet again)