Bash Snippets — Learning Collection with Expected Output

About This Collection

This is a comprehensive set of 64 ready-to-run Bash script snippets designed to help you master shell scripting in Ubuntu/Debian environments. Each snippet includes the complete working code plus commented expected output, making it easy to understand what the script does before you run it.

What you'll find here:

- Variable declaration, manipulation, and scope management
- String operations (length, concatenation, case conversion, substring extraction)
- Arithmetic operations and numeric comparisons
- Control structures (if/else, for loops, while loops, case statements)
- Arrays and iteration techniques
- File and directory operations (creation, deletion, copying, searching)
- Text processing (grep, sed, line counting, pattern matching)
- Functions with parameters and return values
- Command-line argument handling
- Date/time operations and system information retrieval

Who is this for:

- Linux beginners learning Bash scripting
- System administrators automating routine tasks
- Developers writing deployment or build scripts
- Anyone preparing for DevOps/SRE interviews

Environment: All scripts are tested and verified to run on Ubuntu 24.04, Debian Raspberry PI and MacOS Sequoia as a regular user (no root required). They use `/tmp` for temporary files with `$$` (process ID) for uniqueness and clean up after themselves.

How to use: Each snippet is a complete, self-contained script. Copy the code block, save it to a `.sh` file, make it executable with `chmod +x script.sh`, and run it with `./script.sh`.

| *All code blocks below are runnable bash scripts*

1. Print "Hello, World!"

```
#!/bin/bash
echo "Hello, World!"
# Expected output:
# Hello, World!
```

2. Declare and Print Variables

```
#!/bin/bash
name="Alice"
age=30
echo "Name: $name, Age: $age"
# Expected output:
# Name: Alice, Age: 30
```

3. Read User Input

```
#!/bin/bash
echo -n "Enter your name: "
read username
echo "Hello, $username!"
# Expected output (if user enters "Bob"):
# Enter your name: Bob
# Hello, Bob!
```

4. Basic Arithmetic Operations

```
#!/bin/bash
a=10
b=5
echo "Sum: $((a + b))"
echo "Difference: $((a - b))"
```

```
echo "Product: $((a * b))"
echo "Quotient: $((a / b))"
# Expected output:
# Sum: 15
# Difference: 5
# Product: 50
# Quotient: 2
```

5. Check if Number is Even or Odd

```
#!/bin/bash
num=7
if [ $((num % 2)) -eq 0 ]; then
    echo "$num is even"
else
    echo "$num is odd"
fi
# Expected output:
# 7 is odd
```

6. Find Maximum of Two Numbers

```
#!/bin/bash
a=15
b=20
if [ $a -gt $b ]; then
    echo "Max: $a"
else
    echo "Max: $b"
fi
# Expected output:
# Max: 20
```

7. Check if File Exists

```
#!/bin/bash
filename="/etc/passwd"
```

```
if [ -f "$filename" ]; then
    echo "File $filename exists"
else
    echo "File $filename does not exist"
fi
# Expected output:
# File /etc/passwd exists
```

8. Loop from 1 to 5

```
#!/bin/bash
for i in {1..5}; do
    echo "Number: $i"
done
# Expected output:
# Number: 1
# Number: 2
# Number: 3
# Number: 4
# Number: 5
```

9. While Loop Example

```
#!/bin/bash
counter=1
while [ $counter -le 3 ]; do
    echo "Count: $counter"
    ((counter++))
done
# Expected output:
# Count: 1
# Count: 2
# Count: 3
```

10. Function to Add Two Numbers

```
#!/bin/bash
```

```
add_numbers() {
    local sum=$(($1 + $2))
    echo $sum
}
result=$(add_numbers 10 25)
echo "Sum: $result"
# Expected output:
# Sum: 35
```

11. String Length

```
#!/bin/bash
text="Hello"
echo "Length: ${#text}"
# Expected output:
# Length: 5
```

12. String Concatenation

```
#!/bin/bash
first="Hello"
second="World"
combined="$first $second"
echo "$combined"
# Expected output:
# Hello World
```

13. Substring Extraction

```
#!/bin/bash
text="HelloWorld"
sub="${text:0:5}"
echo "$sub"
# Expected output:
# Hello
```

14. Convert String to Uppercase

```
#!/bin/bash
```

```
text="hello world"
upper="${text^^}"
echo "$upper"
# Expected output:
# HELLO WORLD
```

15. Convert String to Lowercase

```
#!/bin/bash
text="HELLO WORLD"
lower="${text,,}"
echo "$lower"
# Expected output:
# hello world
```

16. Count Files in Current Directory

```
#!/bin/bash
count=$(ls -1 | wc -l)
echo "Files: $count"
# Expected output (example):
# Files: 12
```

17. Print Current Date and Time

```
#!/bin/bash
echo "Current date: $(date '+%Y-%m-%d %H:%M:%S')"
# Expected output (example):
# Current date: 2026-02-15 14:30:45
```

18. Create an Array

```
#!/bin/bash
fruits=("apple" "banana" "cherry")
echo "First fruit: ${fruits[0]}"
echo "All fruits: ${fruits[@]}"
# Expected output:
# First fruit: apple
# All fruits: apple banana cherry
```

19. Array Length

```
#!/bin/bash
numbers=(10 20 30 40 50)
echo "Array length: ${#numbers[@]}"
# Expected output:
# Array length: 5
```

20. Loop Through Array

```
#!/bin/bash
colors=("red" "green" "blue")
for color in "${colors[@]}"; do
    echo "Color: $color"
done
# Expected output:
# Color: red
# Color: green
# Color: blue
```

21. Check if Directory Exists

```
#!/bin/bash
dirname="/tmp"
if [ -d "$dirname" ]; then
    echo "Directory $dirname exists"
else
    echo "Directory $dirname does not exist"
fi
# Expected output:
# Directory /tmp exists
```

22. Create a Directory

```
#!/bin/bash
mkdir -p /tmp/test_dir_$$
echo "Created directory: /tmp/test_dir_$$"
rmdir /tmp/test_dir_$$
```

```
# Expected output (example):
# Created directory: /tmp/test_dir_12345
```

23. Remove a File

```
#!/bin/bash
touch /tmp/testfile_$$
echo "Created file"
rm /tmp/testfile_$$
echo "Removed file"
# Expected output:
# Created file
# Removed file
```

24. Copy a File

```
#!/bin/bash
echo "test" > /tmp/source_$$
cp /tmp/source_$$ /tmp/dest_$$
echo "File copied"
cat /tmp/dest_$$
rm /tmp/source_$$ /tmp/dest_$$
# Expected output:
# File copied
# test
```

25. Count Lines in a File

```
#!/bin/bash
echo -e "line1\nline2\nline3" > /tmp/count_$$
echo -e "line1\nline2\nline3" > /tmp/count_$$
echo "Lines: $lines"
rm /tmp/count_$$
# Expected output:
# Lines: 3
```

26. Search for a Pattern in File

```
#!/bin/bash
```

```
echo -e "apple\nbanana\napricot" > /tmp/fruits_$$
grep "^a" /tmp/fruits_$$
rm /tmp/fruits_$$
# Expected output:
# apple
# apricot
```

27. Replace Text in File

```
#!/bin/bash
echo "Hello World" > /tmp/replace_$$
sed -i 's/World/Universe/g' /tmp/replace_$$
cat /tmp/replace_$$
rm /tmp/replace_$$
# Expected output:
# Hello Universe
```

28. Get Current Username

```
#!/bin/bash
echo "Current user: $USER"
# Expected output (example):
# Current user: alice
```

29. Get Current Working Directory

```
#!/bin/bash
echo "Current directory: $(pwd)"
# Expected output (example):
# Current directory: /home/alice
```

30. Print Environment Variable

```
#!/bin/bash
echo "Home directory: $HOME"
# Expected output (example):
# Home directory: /home/alice
```

31. Set and Export Variable

```
#!/bin/bash
export MY_VAR="test_value"
bash -c 'echo "Variable in subshell: $MY_VAR"'
# Expected output:
# Variable in subshell: test_value
```

32. Calculate Factorial

```
#!/bin/bash
factorial() {
    if [ $1 -le 1 ]; then
        echo 1
    else
        local prev=$(factorial $(($1 - 1)))
        echo $(($1 * prev))
    fi
}
echo "Factorial of 5: $(factorial 5)"
# Expected output:
# Factorial of 5: 120
```

33. Check if String is Empty

```
#!/bin/bash
str=""
if [ -z "$str" ]; then
    echo "String is empty"
else
    echo "String is not empty"
fi
# Expected output:
# String is empty
```

34. Check if String is Not Empty

```
#!/bin/bash
str="Hello"
```

```
if [ -n "$str" ]; then
    echo "String is not empty"
else
    echo "String is empty"
fi
# Expected output:
# String is not empty
```

35. Compare Two Strings

```
#!/bin/bash
str1="hello"
str2="hello"
if [ "$str1" = "$str2" ]; then
    echo "Strings are equal"
else
    echo "Strings are not equal"
fi
# Expected output:
# Strings are equal
```

36. Check if Number is Positive

```
#!/bin/bash
num=10
if [ $num -gt 0 ]; then
    echo "$num is positive"
else
    echo "$num is not positive"
fi
# Expected output:
# 10 is positive
```

37. Generate Random Number

```
#!/bin/bash
random=$((RANDOM % 100))
```

```
echo "Random number (0-99): $random"
# Expected output (example):
# Random number (0-99): 42
```

38. Sleep for 2 Seconds

```
#!/bin/bash
echo "Waiting..."
sleep 2
echo "Done waiting"
# Expected output:
# Waiting...
# Done waiting
```

39. Get Script Name

```
#!/bin/bash
echo "Script name: $0"
# Expected output (example):
# Script name: ./script.sh
```

40. Count Command Line Arguments

```
#!/bin/bash
echo "Number of arguments: $#"
echo "Arguments: $@"
# Expected output (when run as: ./script.sh a b c):
# Number of arguments: 3
# Arguments: a b c
```

41. Process Each Argument

```
#!/bin/bash
for arg in "$@"; do
    echo "Argument: $arg"
done
# Expected output (when run as: ./script.sh one two):
# Argument: one
# Argument: two
```

42. Check if Script is Run as Root

```
#!/bin/bash
if [ "$EUID" -eq 0 ]; then
    echo "Running as root"
else
    echo "Not running as root"
fi
# Expected output (normal user):
# Not running as root
```

43. Get Last Exit Status

```
#!/bin/bash
ls /nonexistent 2>/dev/null
if [ $? -ne 0 ]; then
    echo "Previous command failed"
fi
# Expected output:
# Previous command failed
```

44. Redirect Output to File

```
#!/bin/bash
echo "This goes to file" > /tmp/output_$$
cat /tmp/output_$$
rm /tmp/output_$$
# Expected output:
# This goes to file
```

45. Append to File

```
#!/bin/bash
echo "Line 1" > /tmp/append_$$
echo "Line 2" >> /tmp/append_$$
cat /tmp/append_$$
rm /tmp/append_$$
# Expected output:
```

```
# Line 1
# Line 2
```

46. Read File Line by Line

```
#!/bin/bash
echo -e "apple\nbanana\ncherry" > /tmp/fruits_$$
while IFS= read -r line; do
    echo "Fruit: $line"
    echo "Fruit: $line"
rm /tmp/fruits_$$
# Expected output:
# Fruit: apple
# Fruit: banana
# Fruit: cherry
```

47. Case Statement Example

```
#!/bin/bash
fruit="apple"
case $fruit in
    apple)  echo "Red fruit" ;;
    banana) echo "Yellow fruit" ;;
    *)      echo "Unknown fruit" ;;
esac
# Expected output:
# Red fruit
```

48. Find Files by Extension

```
#!/bin/bash
mkdir -p /tmp/findtest_$$
touch /tmp/findtest_$$/file1.txt /tmp/findtest_$$/
file2.sh
find /tmp/findtest_$$ -name "*.txt"
rm -rf /tmp/findtest_$$
# Expected output:
```

```
# /tmp/findtest_12345/file1.txt
```

49. Count Words in String

```
#!/bin/bash
text="Hello World from Bash"
count=$(echo "$text" | wc -w)
echo "Word count: $count"
# Expected output:
# Word count: 4
```

50. Get File Size

```
#!/bin/bash
echo "test content" > /tmp/size_$$
size=$(stat -f%z /tmp/size_$$ 2>/dev/null || stat -
c%s /tmp/size_$$)
echo "File size: $size bytes"
rm /tmp/size_$$
# Expected output:
# File size: 13 bytes
```

51. Check if Variable is Set

```
#!/bin/bash
VAR="set"
if [ -v VAR ]; then
    echo "VAR is set"
else
    echo "VAR is not set"
fi
# Expected output:
# VAR is set
```

52. Default Value for Variable

```
#!/bin/bash
NAME=${NAME:-"Guest"}
echo "Hello, $NAME"
```

```
# Expected output:
# Hello, Guest
```

53. Trim Whitespace from String

```
#!/bin/bash
text="  hello world  "
trimmed=$(echo "$text" | xargs)
echo "Trimmed: '$trimmed'"
# Expected output:
# Trimmed: 'hello world'
```

54. Check if Number is Prime

```
#!/bin/bash
is_prime() {
    local n=$1
    if [ $n -lt 2 ]; then echo "no"; return; fi
    if [ $n -eq 2 ]; then echo "yes"; return; fi
    if [ $((n % 2)) -eq 0 ]; then echo "no"; return; fi
    local i=3
    while [ $((i * i)) -le $n ]; do
        if [ $((n % i)) -eq 0 ]; then echo "no";
return; fi
        i=$((i + 2))
    done
    echo "yes"
}
echo "Is 17 prime? $(is_prime 17)"
# Expected output:
# Is 17 prime? yes
```

55. Sum Numbers from 1 to N

```
#!/bin/bash
n=10
sum=0
```

```
sum=0
    sum=$((sum + i))
done
echo "Sum from 1 to $n: $sum"
# Expected output:
# Sum from 1 to 10: 55
```

56. Reverse a String

```
#!/bin/bash
text="hello"
reversed=$(echo "$text" | rev)
echo "Reversed: $reversed"
# Expected output:
# Reversed: olleh
```

57. Get First N Characters

```
#!/bin/bash
text="Hello World"
first="${text:0:5}"
echo "First 5 chars: $first"
# Expected output:
# First 5 chars: Hello
```

58. Get Last N Characters

```
#!/bin/bash
text="Hello World"
last="${text: -5}"
echo "Last 5 chars: $last"
# Expected output:
# Last 5 chars: World
```

59. Count Occurrences of Character

```
#!/bin/bash
text="hello world"
char="l"
```

```
count=$(echo "$text" | grep -o "$char" | wc -l)
echo "Occurrences of '$char': $count"
# Expected output:
# Occurrences of 'l': 3
```

60. Split String by Delimiter

```
#!/bin/bash
#!/bin/bash
for part in "${parts[@]}"; do
    echo "Part: $part"
done
# Expected output:
# Part: apple
# Part: banana
# Part: cherry
```

61. Check if Command Exists

```
#!/bin/bash
#!/bin/bash
    echo "Command 'ls' exists"
else
    echo "Command 'ls' not found"
fi
# Expected output:
# Command 'ls' exists
```

62. Get Current Unix Timestamp

```
#!/bin/bash
timestamp=$(date +%s)
echo "Current timestamp: $timestamp"
# Expected output (example):
# Current timestamp: 1708012800
```

63. Calculate Script Execution Time

```
#!/bin/bash
```

```
start=$(date +%s)
sleep 1
end=$(date +%s)
duration=$((end - start))
echo "Execution time: ${duration}s"
# Expected output:
# Execution time: 1s
```

64. Print Multiplication Table

```
#!/bin/bash
num=5
echo "Multiplication table for $num:"
for i in {1..10}; do
    result=$((num * i))
    echo "$num x $i = $result"
done
# Expected output:
# Multiplication table for 5:
# 5 x 1 = 5
# 5 x 2 = 10
# 5 x 3 = 15
# 5 x 4 = 20
# 5 x 5 = 25
# 5 x 6 = 30
# 5 x 7 = 35
# 5 x 8 = 40
# 5 x 9 = 45
# 5 x 10 = 50
```
