SHELL SCRIPTING CHEAT SHEET (Beginner to Advanced)

```
=========================
BASICS
=========================
#!/bin/bash              # Shebang (required at top)
echo "Hello"            # Print to stdout
# This is a comment

=========================
VARIABLES
=========================
NAME="John"
echo "Hi, $NAME"
readonly NAME           # Makes variable read-only
unset NAME              # Deletes variable

=========================
DATA TYPES & ARITHMETIC
=========================
NUM1=5
NUM2=3
SUM=$((NUM1 + NUM2))     # Arithmetic
echo $((5**2))           # Exponentiation

=========================
STRINGS
=========================
STR="Hello World"
echo ${#STR}             # String length
echo ${STR:0:5}          # Substring

=========================
ARRAYS
=========================
arr=(one two three)
echo ${arr[1]}           # Index access
echo ${arr[@]}           # All elements
echo ${#arr[@]}          # Length

=========================
CONDITIONALS
=========================
if [ $a -gt $b ]; then
  echo "a > b"
elif [ $a -eq $b ]; then
  echo "a = b"
else
  echo "a < b"
fi

# Operators:
# -eq -ne -gt -lt -ge -le
# == != < > -z -n -f -d
```

```
=========================
LOOPS
=========================
for i in {1..5}; do
  echo $i
done

while [ $x -le 5 ]; do
  echo $x
  ((x++))
done

until [ $x -gt 5 ]; do
  echo $x
  ((x++))
done

=========================
FUNCTIONS
=========================
my_func() {
  echo "Called with $1"
}
my_func "argument"

=========================
COMMAND LINE ARGUMENTS
=========================
echo "First arg: $1"
echo "All args: $@"
echo "Number of args: $#"
echo "Script name: $0"

=========================
INPUT / OUTPUT
=========================
read -p "Enter name: " NAME
echo "Hello $NAME"

command > file.txt       # stdout
command 2> error.txt     # stderr
command >> file.txt      # append
command > all.txt 2>&1   # both

=========================
ADVANCED: FUNCTIONS & EXIT CODES
=========================
function myFunc {
  if [ ! -f "$1" ]; then
    echo "File not found!"
    return 1
  fi
  return 0
}
```

```
myFunc "file.txt" || echo "Handle error"

=========================
ADVANCED: GETOPTS (FLAGS)
=========================
while getopts ":u:p:" opt; do
  case $opt in
    u) USER="$OPTARG" ;;
    p) PASS="$OPTARG" ;;
    *) echo "Invalid option: -$OPTARG" ;;
  esac
done

=========================
ADVANCED: TRAPS
=========================
trap "echo 'Ctrl+C pressed!'; exit" SIGINT
trap cleanup EXIT

cleanup() {
  echo "Script ended. Cleaning up..."
}

=========================
ADVANCED: PROCESSING
=========================
ps aux | grep "nginx"
find . -type f -name "*.sh"
awk '{ print $1 }' file.txt
sed 's/foo/bar/g' file.txt

=========================
BEST PRACTICES
=========================
set -e                # Exit on error
set -u                # Treat unset vars as error
set -o pipefail       # Exit if any command fails in pipe

[[ $x -gt 5 ]]        # Use [[ ]] for safe conditionals

echo "Use quotes: "$VAR"" to prevent word splitting
```