**Bash User Input Cheatsheet**

| Positional variables | |
| --- | --- |
| Variables | Description |
| $0 | Name of script |
| $1 - $9 | First 9 arguments |
| ${10} - ${99} | Next 90 arguments |
| $# | Number of arguments |
| $\* | All arguments passed as a single item |
| $@ | All arguments passed as multiple items |

|  |  |
| --- | --- |
| **Reading input from the user** | |
| read -p “Enter your name: “ NAME | Prompt the user to enter for input  Input is assigned to $NAME |
| read -p “Enter first and last name: “ FIRST LAST | Two words will be assigned to $FIRST and $LAST |
| if read -t 5 -p “Enter name: “ NAME ;then  echo “Your name is $NAME”  else  echo “You took to long to enter your name”  fi | User has to enter input in less than 5 seconds  Return code 1 if user does not respond |
| read -s -p “Enter your password: “ PASS | Do not echo input on the screen |

|  |  |
| --- | --- |
| **Getopts example** | |
| Example | Description |
| while getopts :dha: opt ;do  case $opt in  d) set -x ;;  h) echo “Syntax: $0 -a <arg>;;  a) ARGUMENT**=**"$OPTARG" ;;  \?) echo “Unknown option” ;;  esac  done  shift $(($OPTIND -1)) | Leading colon suppresses error output.  Accepts -d -h and -a <arg> options.  The <arg> after -a is required.  If -d is passed set shell option -x  If -h is passed echo syntax  If -a <arg> passed assign <arg> to $ARGUMENT  If not -d ,-h, or -a <arg> then display error and loop  OPTIND -1 resets positional variable back to $1 after all options have been passed. This allows the script to pass additional arguments using $1, $2 positions. |