Bash Script Helper

```
#!/bin/bash [fibonacci]
fibbonacci(){
a=0
b=1
c=0
echo "" $b
for (( i=0; i<$1; i++ ))
do
	c=$(expr $a + $b)
	a=$b
	b=$c
	echo "" $c
done
}
fibbonacci 5
```

```
number_1=20.5 [arithmatic]
number_2=5.2
echo "scale=2;$number_1+$number_2" | bc
echo "scale=2;20.5/5.2" | bc
echo "scale=2;20.5+5.2" | bc
echo "scale=2;20.5*5.2" | bc
echo "scale=2;20.5-5.2" | bc
echo "scale=2;20.5%5" | bc
num=27
echo "scale=2;sqrt($num)" | bc -l
echo "scale=2;3^3" | bc -l
```

```
#!/bin/bash [array]
os=('ubuntu' 'windows' 'macos')
os[3]='linux' #add element
os[1]='linux' #update element
unset os[1] #remove element
echo "${os[@]}"
echo "${os[1]}"
echo "${!os[@]}" #elements no
echo "${#os[@]}" #length array
string=HMMahmudulHasan
echo "${string[@]}"
```

```
echo -e "enter value : \c" [switch case]
read value
case $value in
	[a-z] )
		echo "user entered $value a to z" ;;
	[A-Z] )
		echo "user entered $value A to Z" ;;
	[0-9] )
		echo "user entered $value 0 to 9" ;;
	? )
		echo "user entered $value special charecter" ;;
	* )
		echo "user entered $value unknown value" ;;
Esac
```

```
echo -e "enter file name : \c" [cat in script]
read file_name

if [ -f $file_name ]
then
	if [ -w $file_name ]
	then
		echo "write something & press CLTR+D."
		cat >> $file_name
	else
		echo "file don't have write permission."
	fi
else
	echo "$file_name not exixt."
fi
```

```
! /bin/bash [srach file]

search file by name
echo -e "enter file name : \c"
read file_name

if [ -e $file_name ]
#-d directory/-b music,picture,video/-c word,textfile/-s
empty or not
then
	echo "$file_name found"
else
	echo "$file_name not found"
Fi
```

```
# if else
word=abc
word1=abc

if [ $word != $word1 ]; then
	echo "condition working."
else
	echo "condition is not working."
fi
# if-elif
word=abc
if [[ $word == "abc" ]]
then
	echo "1.condition working."
elif [[ $word == "bcd" ]]
then
	echo "2.condition is working."
else
	echo "condition is not working"
Fi
```

```
#!/bin/bash [or]
age=25
# if [ "$age" -eq 18 ] || [ "$age" -eq 30 ]
# if [ "$age" -eq 18 -o "$age" -eq 30 ]
if [[ "$age" -eq 18 || "$age" -eq 30 ]]
then
	echo "valide age"
else
	echo "not valide age"
Fi
```

```
for item in * [shell command]
do
	# if [ -d $item ] #directory list
	if [ -f $item ] #file list
	then
		echo $item
	fi
Done
```

```
!/bin/bash [shell command]
#exAMPLE:1
for command in ls pwd date
do
	echo "------------$command----------"
	$command
Done
```

```
#!/bin/bash
echo "HM Mahmudul Hasan"
x=2
echo $BASH
echo $BASH_VERSION
echo $x
echo $PWD
name=hridoy
echo the name is $name
```

```
#!/bin/bash [while loop]
n=1
# while [ $n -le 10 ]
while (( $n <= 10 ))
do
	echo "$n"
```

```
        # n=$(( n+1 ))
        # (( n++ ))
        (( ++n ))
        # sleep 1 #pausing output
        # gnome-terminal & # open extra terminal
Done
```

```
!/bin/bash
#file read using while loop[input redirection]
while read p
do
        echo $p
done < script.sh
#file read using while loop[read content & uptput it]
cat script.sh | while read p
do
        echo $p
done < script.sh
# read file from some folder.
#terminal
# cat /etc/host.conf
#read file using IFS=internal field seperator [word spliting]
while IFS= read -r line
do
        echo $p
done < /etc/host.conf
```

```
print_desktop(){ [function]
        ls -al
        echo "-------------------------------------"
        ls
}
print_desktop
#example 2
print_name(){
#first parameter will store at $1
#second parameter will store at $2
        sum=$(expr $1 + $2 + $3)
        echo "the sum is :" $sum
}
print_name "3" "4" "5"
```

```
printname(){
        #name="mark" #global variable
        local name="mark" #local variable
        echo "the name is : " $name
}
name="tom"
echo "before funtion call, name is :" $name
printname
echo "after funtion call, name is :" $name
```

```
#!/bin/bash [number triangle]
echo "enter the n value"
read n
for((i=1;i<=n;i++))
do
for((j=1;j<=i;j++))
do
echo -n $j " "
done
echo
Done
```

```
#!/bin/bash [number triangle]
read -p "Enter Number:" number
for((row=1;row<=number;row++))
do
    for((spaces=row;spaces<=number;spaces++))
    do
        echo -ne " "
    done
    for((j=1;j<=row;j++))
    do
        echo -ne "$j"
    done
    for((l=(row-1);l>=1;l--))
    do
        echo -ne "$l"
    done
    echo
done
```