. Any shell scripting program.

if\_else(){

echo "Enter your marks:"

read marks

if ! [[ $marks =~ ^[0-9]+$ ]]; then

echo "Invalid input. Please enter a valid number."

return

fi

if [ $marks -ge 70 ]; then

echo "Your grade is A."

elif [ $marks -ge 40 ]; then

echo "Your grade is B."

else

echo "You are FAIL"

fi

}

do\_while\_prime(){

echo "Enter the maximum number up to which you want to print prime numbers:"

read max\_num

if ! [[ $max\_num =~ ^[0-9]+$ ]]; then

echo "Invalid input. Please enter a valid number."

return

fi

is\_prime() {

local num=$1

if [ $num -le 1 ]; then

return 1

fi

for (( i=2; i\*i <= num; i++ )); do

if [ $((num % i)) -eq 0 ]; then

return 1

fi

done

return 0

}

num=2

while true

do

is\_prime $num

if [ $? -eq 0 ]; then

echo "$num"

fi

num=$((num + 1))

[ $num -gt $max\_num ] && break

done

}

until\_loop\_armstrong(){

echo "Enter the number to check if it is an Armstrong number:"

read num

if ! [[ $num =~ ^[0-9]+$ ]]; then

echo "Invalid input. Please enter a valid number."

return

fi

is\_armstrong() {

local num=$1

local sum=0

local n=${#num} # Number of digits

local original=$num

until [ $num -le 0 ]; do

local digit=$((num % 10))

sum=$((sum + digit\*\*n))

num=$((num / 10))

done

[ $sum -eq $original ]

}

if is\_armstrong $num; then

echo "$num is an Armstrong number."

else

echo "$num is not an Armstrong number."

fi

}

for\_loop\_factorial(){

echo "Enter the number to calculate its factorial:"

read num

if ! [[ $num =~ ^[0-9]+$ ]]; then

echo "Invalid input. Please enter a valid number."

return

fi

factorial=1

for (( i=1; i<=num; i++ ))

do

factorial=$((factorial \* i))

done

echo "The factorial of $num is $factorial"

}

while true; do

echo "Enter your choice:"

echo "1. if-else ladder example"

echo "2. for loop factorial example"

echo "3. do-while loop example"

echo "4. check if a number is an Armstrong number using until loop"

echo "5. exit"

read choice

case $choice in

1) if\_else;;

2) for\_loop\_factorial;;

3) do\_while\_prime;;

4) until\_loop\_armstrong;;

5) echo "Exiting..."; exit;;

\*) echo "Invalid input"

esac

done