CSE 4502

Operating System Lab Assignment

Lab Task on Shell Scripting

Md. Rafid Haque

ID: 1700 420 72

1.

- First I created a function which will display the --help or -h message.

```
Help()
{
   # Display Help
   echo "-./crdir.sh creates files in the Documents directory."
   echo "- Syntax : ./crdir.sh <dir> <subdir_1> <subdir_2> ... <subdir_N>"
   echo "-- Parameter 1 <dir> : Name of the parent directory."
   echo "-- Parameter 2 <subdir_1> : Name of the 1st sub-directory."
   echo "--."
   echo "-- Parameter N <subdir_N> : Name of the Nth sub-directory."
}
```

- In arguments variable i took the number of arguments in, then changed it to suit my loop.
- List of arguments taken in list_arguments.

```
arguments=$#
arguments=$(( arguments - 1))

i=0
list_arguments=( "$@" )
```

- This particular while loop will take care of the --help message to show.

```
while getopts ":h" option; do
   case $option in
      h) # display Help
         Help
         exit;;
   esac
done
```

- Then in the first if statement we took care if someone types the command without any arguments given.
- Otherwise it will just make directories and sub-directories in order.

```
if [[ $arguments == -1 ]]
then
  echo "Syntax Error!!"
  echo "Syntax : ./crdir.sh <dir> <subdir_1> <subdir_2> ... <subdir_N>"
  echo "For more information type: ./crdir.sh -h"
else
  until [ $i -gt $arguments ]
  do
    mkdir "${list_arguments[i]}"
    cd "${list_arguments[i]}"
    i=$(( i +1 ))
  done
fi
```

2.

To find the factorial of a number

- Take input number.
- Use loop to multiply in each iteration.
- Echo the result as output.

```
echo "Enter a number: "
read number

temp=1
i=1

until [ $i -gt $number ]
do
  temp=$(( temp * i ))
  i=$(( i +1 ))
done

echo Factorial of $number is: $temp
```

- The solution was to wrap the $@ into a quotation.

  Before:

```
for var in $@
do
  echo $var
Done
```

  After:

```
for var in "$@"
do
  echo $var
done
```

The reason why it didn’t work before is, the $@ takes string into array considering inbetween spaces as separator, not the quotation marks.
Wrapping it into quotation mark thus does the trick.

4.

The task was to enisle the integers from non-integers.

- First we created a 2 array, one for each type.
- Then, for each variable in given command, we used the hint's condition to separate each kind to it's designated arrays.
- Finally we print the lists.

```
lis1[0]=0
i=0

lis2[0]=0
j=0

for var in "$@"
do
  if [[ $var =~ ^[+-]?[0-9]+$ ]]
  then
     lis1[$i]=$var
     i=$(( i + 1 ))
  else
     lis2[$j]=$var
     j=$(( j + 1 ))
  fi
done

echo List of Integers: [${lis1[*]}]
echo List of Non-Integers: [${lis2[*]}]
```