

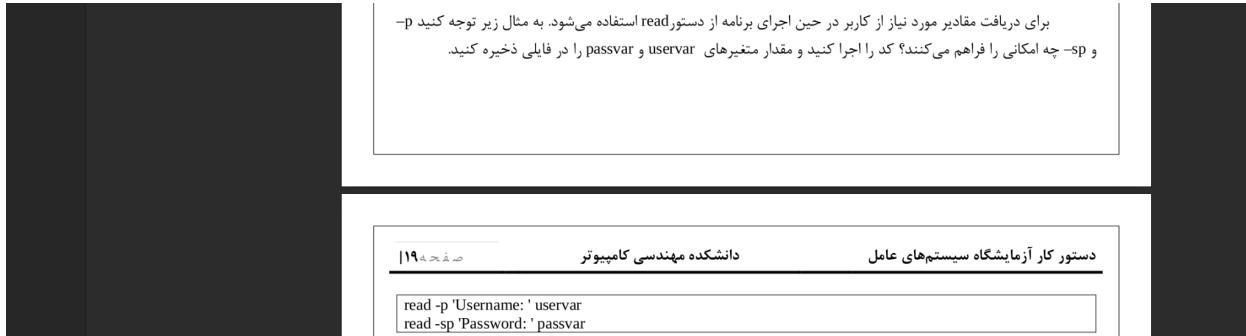
متغیرهای خاصی وجود دارد که مقادیر آنها از قبیل تعیین شده‌اند و می‌توان در کاپردهای خاص از آنها استفاده کرد. مانند:

\$0 - \$1 - \$9 - \$# - \$\$ - \$USER -

که در هنگام اجرای فایل می‌توان به فایل ورودی داد مثلا: 1 3

که در آن مقادیر ۱ و ۳ که با فاصله آمدۀ‌اند آرگومان هستند و برای استفاده از این آرگومان‌ها باید از متغیرهای \$1 و \$2 استفاده کرد. مقدار سایر متغیرهای خاص را بباید. اگر بیش از ۱۰ آرگومان ورودی باشد، چگونه باید به مقدار ۱۰-امین آرگومان دست یافت؟

در این حالت که بیشتر از ۹ آرگومان داریم، عدد را داخل {} قرار می‌دهیم. مثلاً برای دسترسی به آرگومان دهم از {10} استفاده می‌کنیم.



برای دریافت مقادیر مورد نیاز از کاربر در حین اجرای برنامه از دستور `read` استفاده می‌شود. به مثال زیر توجه کنید -p و -sp - چه امکانی را فراهم می‌کنند؟ کد را اجرا کنید و مقدار متغیرهای `uservar` و `passvar` را در فایل ذخیره کنید.

نوشتن کد و دستورات گفته شده و ذخیره‌ی فایل اسکریپت:

دستور `read` یک خط از ورودی بخواند و آنرا در متغیر تعریف شده ذخیره می‌کند.

آپشن `p` قبل از اینکه کاربر متن خود را وارد کند به آن یک `prompt` می‌دهد.

آپشن `s` از نمایش ورودی وارد شده توسط کاربر جلوگیری می‌کند که می‌تواند برای `prompt` هایی مثل رمز عبور استفاده شود.

The screenshot shows the terminal executing the script. The output is:

```
ubuntu@mehrdad:~$ ./script.sh
Username: mehdad
Password: ****
mehdad
****
```

At the bottom of the terminal, there is a menu with keyboard shortcuts:

- ^G Help
- ^O Write Out
- ^W Where Is
- ^K Cut
- ^T Execute
- ^C Location
- ^X Exit
- ^R Read File
- ^N Replace
- ^U Paste
- ^J Justify
- ^/ Go To Line

برای انجام محاسبات شیوه‌های مختلفی وجود دارد. به مثال‌های زیر توجه کنید. کد را اجرا کنید و نتیجه را گزارش کنید.

```
let a=10+8
echo $a
expr 5 \* 4
expr 5 / 4
expr 11 % 2
a=$(( expr 10 - 3 ))
echo $a
b=$(( ( a + 3 ))
echo $b
((b++))
echo $b
```

نوشتن اسکرپت:

دستور `expr` یک عبارت ریاضی تعریف می‌کند و آنرا انجام می‌دهد و نتیجه را `echo` می‌کند. بک اسلش موجود قبل از `*` به معنی مشخص کردن ضرب بودن آن است.

بنابراین اول مقدار متغیر `a` که ۱۸ است پرینت می‌شود. سپس ۲۰ که حاصل ضرب ۵ در ۴ است. سپس رند تقسیم ۴ / ۵ پرینت می‌شود و بعد از آن باقی مانده تقسیم ۱۱ به ۲ پرینت می‌شود.

سپس یک عبارت ریاضی تعریف می‌کنیم و ۱۰ را منهای ۳ می‌کنیم و حاصل را درون `a` می‌ریزیم و در نهایت آنرا چاپ می‌کنیم که ۷ است.

مقدار نهایی `a` را با ۳ جمع می‌کنیم و در `b` ذخیره و چاپ می‌کنیم.

مقدار `b` را یک واحد افزایش می‌دهیم و ۱۱ را چاپ می‌کنیم.

The screenshot shows a terminal window with the title 'script.sh'. The terminal prompt is 'ubuntu@mehrdad: ~'. The nano editor is open with the following content:

```
GNU nano 6.2
#!/bin/bash
let a=10+8
echo $a
expr 5 \* 4
expr 5 / 4
expr 11 % 2
a=$(( expr 10 - 3 ))
echo $a
b=$(( ( a + 3 ))
echo $b
((b++))
echo $b
```

At the bottom of the terminal window, there are several command-line options: `[ Read 12 lines ]`, `^G Help`, `^O Write Out`, `^W Where Is`, `^K Cut`, `^T Execute`, and `^C Location`.

اجرای اسکریپت:

```
ubuntu@mehrdad:~$ ./script.sh
18
20
1
1
7
10
11
```

#### خروجی‌های مورد انتظار آزمایش:

انتظار می‌رود دانشجویان پس از یادگیری مطالب فوق بتوانند به سوالات زیر پاسخ دهند و نتیجه را به مدرس ارائه دهند.

۱. دستنوشتی (اسکریپتی) بنویسید که دو عددی که به صورت آرگومان به آن داده شده را

الف- با هم جمع کند و نتیجه را اعلام کند

دو آرگومان که هنگام اجرای برنامه به آن می‌دهیم را از \$1 و \$2 دسترسی پیدا می‌کنیم و آنها را درون متغیری دیگر ذخیره می‌کنیم.

مجموع این دو متغیر را با استفاده از دستور expr بدست می‌آوریم و در متغیر sum ذخیره می‌کنیم.

مقدار خروجی را echo می‌کنیم.

```
GNU nano 6.2
#!/bin/bash

num1=$1
num2=$2

sum=$(expr $num1 + $num2)

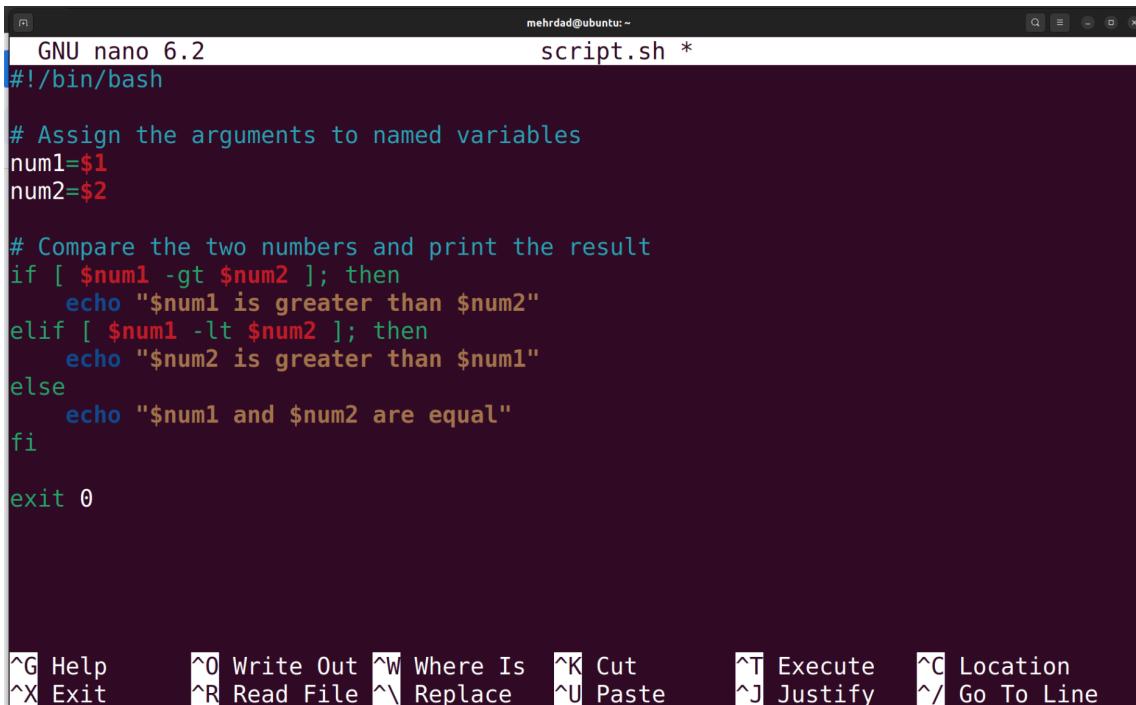
echo "The sum of $num1 and $num2 is: $sum"
```

#### خروجی اسکریپت:

```
(base) mehrdad@ubuntu:~$ ./script.sh 1 2
The sum of 1 and 2 is: 3
```

ب- عدد بزرگتر را نمایش دهد.

دو آرگومان ورودی می‌گیرد و با استفاده از آپشن های `lt` که به معنی کوچکتر بودن و `gt` بزرگتر بودن مقایسه می‌کند.



```
mehrdad@ubuntu:~$ nano script.sh
GNU nano 6.2                                     script.sh *
#!/bin/bash

# Assign the arguments to named variables
num1=$1
num2=$2

# Compare the two numbers and print the result
if [ $num1 -gt $num2 ]; then
    echo "$num1 is greater than $num2"
elif [ $num1 -lt $num2 ]; then
    echo "$num2 is greater than $num1"
else
    echo "$num1 and $num2 are equal"
fi

exit 0

^G Help      ^O Write Out ^W Where Is   ^K Cut        ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace    ^U Paste     ^J Justify   ^/ Go To Line
```

تست اسکریپت:

```
(base) mehrdad@ubuntu:~$ ./script.sh 100 123
123 is greater than 100
(base) mehrdad@ubuntu:~$
```

ج- اگر کاربر در وارد کردن ورودی ها اشتباه کرده بود راهنمای مناسبی چاپ کند.

## تعداد آرگومان های داده شده را دارد. بنابراین چک میکنیم اگر تعداد آرگومان ها مساوی ۲ نبود یک پیامی `echo` شود.

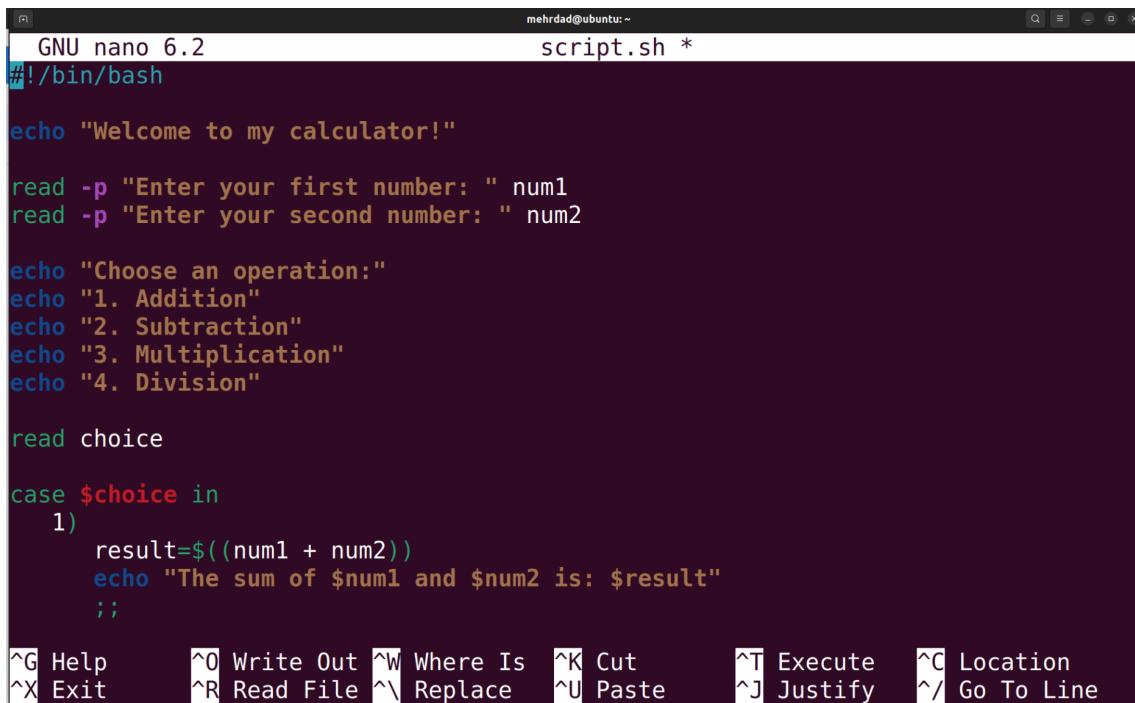
```
(base) mehrdad@ubuntu:~$ cat > script.sh
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "Error: expected two arguments"
    exit 1
fi
```

تست اسکریپت:

```
(base) mehrdad@ubuntu:~$ ./script.sh 2 2 3
Error: expected two arguments
```

۲. ماشین حسابی با استفاده از `case` طراحی کنید.

دو عدد ورودی را با استفاده از دستور `read` و یک پaramپت مشخص، از کاربر می‌گیریم.  
از کاربر می‌خواهیم که یک عمل ریاضی را انتخاب کند سپس مقدار انتخاب شده را در متغیر `choice` میریزیم. سپس با استفاده از مقدار عددی `choice` را چک می‌کنیم و مطابق با آن نتیجه را در یک متغیر دیگر میریزیم و آنرا `echo` می‌کنیم:



```
mehrdad@ubuntu:~$ nano script.sh
#!/bin/bash

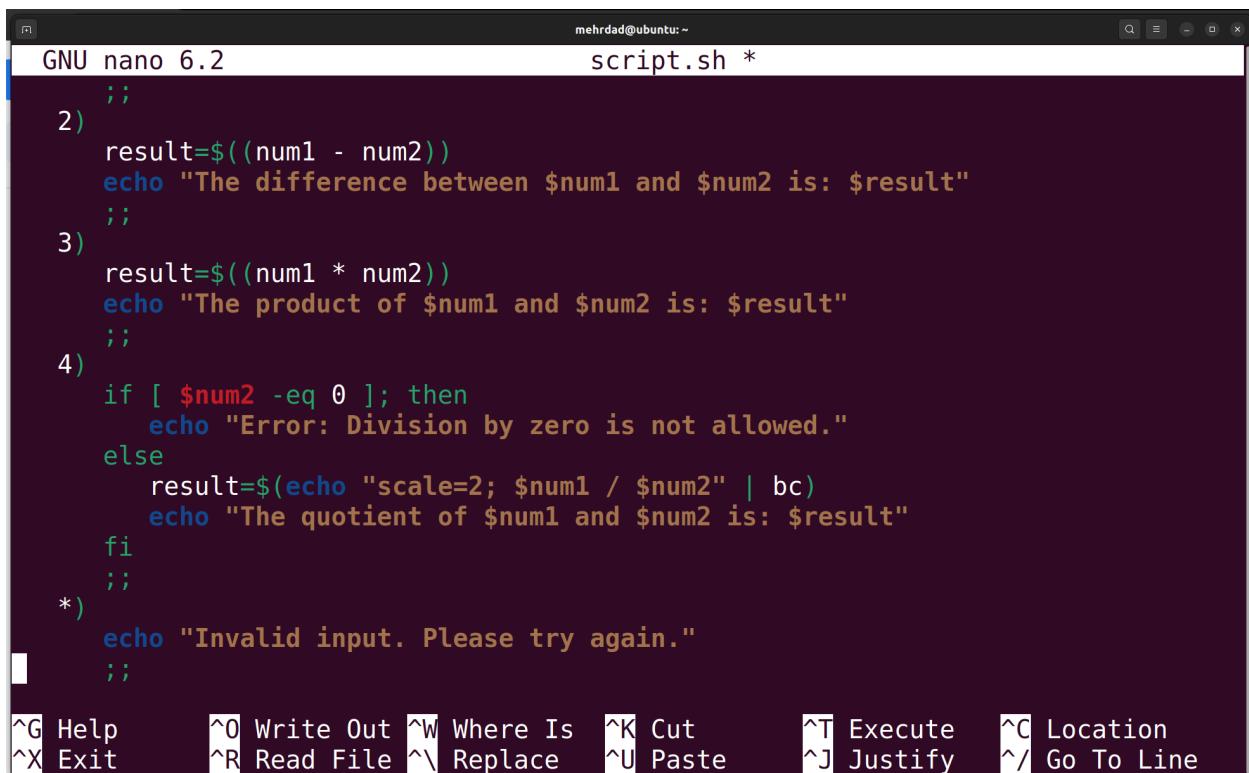
echo "Welcome to my calculator!"

read -p "Enter your first number: " num1
read -p "Enter your second number: " num2

echo "Choose an operation:"
echo "1. Addition"
echo "2. Subtraction"
echo "3. Multiplication"
echo "4. Division"

read choice

case $choice in
  1)
    result=$((num1 + num2))
    echo "The sum of $num1 and $num2 is: $result"
    ;;
  ^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
  ^X Exit      ^R Read File ^\ Replace   ^U Paste   ^J Justify  ^/ Go To Line
```



```
;;
  2)
    result=$((num1 - num2))
    echo "The difference between $num1 and $num2 is: $result"
    ;;
  3)
    result=$((num1 * num2))
    echo "The product of $num1 and $num2 is: $result"
    ;;
  4)
    if [ $num2 -eq 0 ]; then
      echo "Error: Division by zero is not allowed."
    else
      result=$(echo "scale=2; $num1 / $num2" | bc)
      echo "The quotient of $num1 and $num2 is: $result"
    fi
    ;;
  *)
    echo "Invalid input. Please try again."
    ;;
esac

^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste   ^J Justify  ^/ Go To Line
```

تست اسکریپت:

```
(base) mehrdad@ubuntu:~/Documents$ ./script.sh
Welcome to my calculator!
Enter your first number: 1
Enter your second number: 2
Choose an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
1
The sum of 1 and 2 is: 3
```

۳. برنامه‌ای بنویسید که به طور متوالی از کاربر عدد دریافت کند و عددی چاپ کند که ترتیب ارقامش معکوس باشد. مثلاً ۵۶۷ را به صورت ۷۶۵ چاپ کند. سپس جمع ارقام آن را چاپ کند.

سه متغیر input, reverse, sum را تعریف می‌کنیم.

درون یک حلقه بی نهایت هر دفعه ورودی را از کاربر می‌گیریم چنانچه q بود از حلقه خارج می‌شود در غیر این صورت با استفاده از دستور rev و پایپ کردن مقدار ورودی داده شده به این دستور، عدد وارد شده را بر عکس می‌کنیم و چاپ می‌کنیم.

```
GNU nano 6.2          scripts.sh
#!/bin/bash

# Initialize variables
input=""
reverse=""
sum=0

# Loop until user enters 'q' to quit
while true; do
    # Prompt user for input
    read -p "Enter a number (or 'q' to quit): " input

    # Check if input is 'q'
    if [[ "$input" == "q" ]]; then
        break
    fi

    # Reverse the input using 'rev' command
    reverse=$(echo $input | rev)

    # Print the reversed input
    echo "Reversed input: $reverse"
```

[ Read 32 lines ]

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Local

بعد از معکوس کردن ارقام، مجموع ارقام را با استفاده از یک حلقه ای که به تعداد کاراکتر های موجود در ورودی پیمایش می‌کند، محاسبه می‌کنیم و به sum اضافه می‌کنیم.

در نهایت که از حلقه خارج شدیم مقدار نهایی sum را پرینت می کنیم.

```
GNU nano 6.2                                     script5.sh
read -p "Enter a number (or 'q' to quit): " input

# Check if input is 'q'
if [[ "$input" == "q" ]]; then
    break
fi

# Reverse the input using 'rev' command
reverse=$(echo $input | rev)

# Print the reversed input
echo "Reversed input: $reverse"

# Calculate the sum of the digits
for (( i=0; i<#${input}; i++ )); do
    digit=${input:i:1}
    sum=$((sum + digit))
done
done

# Print the final sum
echo "Sum of all entered digits: $sum"
```

تست اسکریپت:

```
(base) mehrdad@ubuntu:~/Documents/Operating-systems-lab$ ./script5.sh
Enter a number (or 'q' to quit): 65464
Reversed input: 46456
Enter a number (or 'q' to quit): 321321
Reversed input: 123123
Enter a number (or 'q' to quit): q
Sum of all entered digits: 37
```

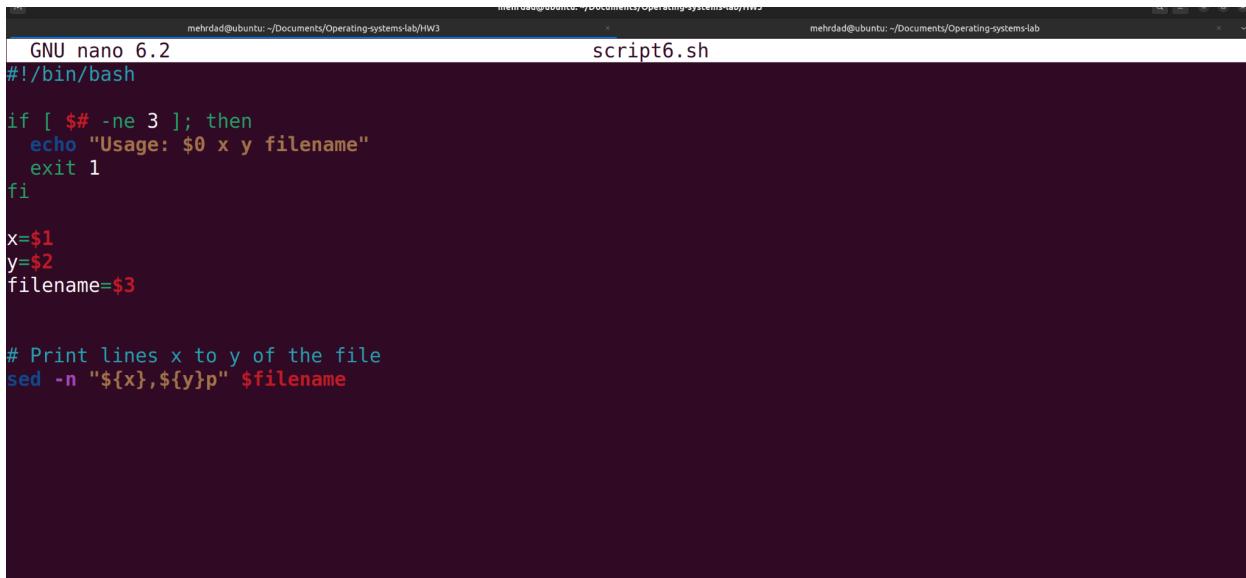
۴. برنامه‌ای بنویسید که در هنگام اجرا دو عدد x,y و اسم یک فایل را دریافت کند و در خروجی خط x تا y ام فایل مذکور را نمایش دهد.

با دستور زیر یک فایل ۲۱ خطی ایجاد می کنیم:  
در این دستور یک حلقه for میزیم تا ۲۱ بار پیمایش شود و هر دفعه شماره خط را چاپ کند سپس مقدار آن را به فایل مشخص شده کند و append کند و redirect

```
(base) mehrdad@ubuntu:~/Documents/Operating-systems-lab$ for i in {1..21}; do echo "This is line $i." >> myfile.txt; done
(base) mehrdad@ubuntu:~/Documents/Operating-systems-lab$
```

در این اسکریپت چک می کنیم که حتما ۳ آرگومان ورودی داده شده باشد.

سپس با استفاده از دستور **sed** و آپشن **n** برای اینکه تا وقتی مشخص نکردیم چیزی پرینت نکند و سپس شماره خط آغازین و پایانی با **p** و در نهایت فایل مورد نظر، خطوط را پرینت می کنیم.



```
GNU nano 6.2
#!/bin/bash

if [ $# -ne 3 ]; then
    echo "Usage: $0 x y filename"
    exit 1
fi

x=$1
y=$2
filename=$3

# Print lines x to y of the file
sed -n "${x} , ${y} p" $filename
```

تست اسکریپت:

```
(base) mehrdad@ubuntu: ~/Documents/Operating-systems-lab/HW3$ ./script6.sh 4 9 myfile.txt
This is line 4.
This is line 5.
This is line 6.
This is line 7.
This is line 8.
This is line 9.
```

#### سوال امتیازی:

ماشین حسابی برای اعداد حقیقی بنویسید.

از دستور **bc** برای نوشتن ماشین حساب اعداد حقیقی استفاده شد که این دستور یک آپشن | دارد که دقت اعشار را سنت می کند.

```

GNU nano 6.2                                     script7.sh
#!/bin/bash
read -p "Enter the first number: " num1
read -p "Enter the operation (+,-,*,/): " op
read -p "Enter the second number: " num2

case "$op" in
    "+")
        result=$(echo "$num1 + $num2" | bc -l)
        ;;
    "-")
        result=$(echo "$num1 - $num2" | bc -l)
        ;;
    "*")
        result=$(echo "$num1 * $num2" | bc -l)
        ;;
    "/")
        result=$(echo "$num1 / $num2" | bc -l)
        ;;
esac

echo "Result: $result"

```

تست اسکریپت:

```

(base) mehrdad@ubuntu:~/Documents/Operating-systems-lab/HW3$ ./script7.sh
Enter the first number: 123123.123123
Enter the operation (+,-,*,/): /
Enter the second number: 321321.654
Result: .38317717337219980823

```