

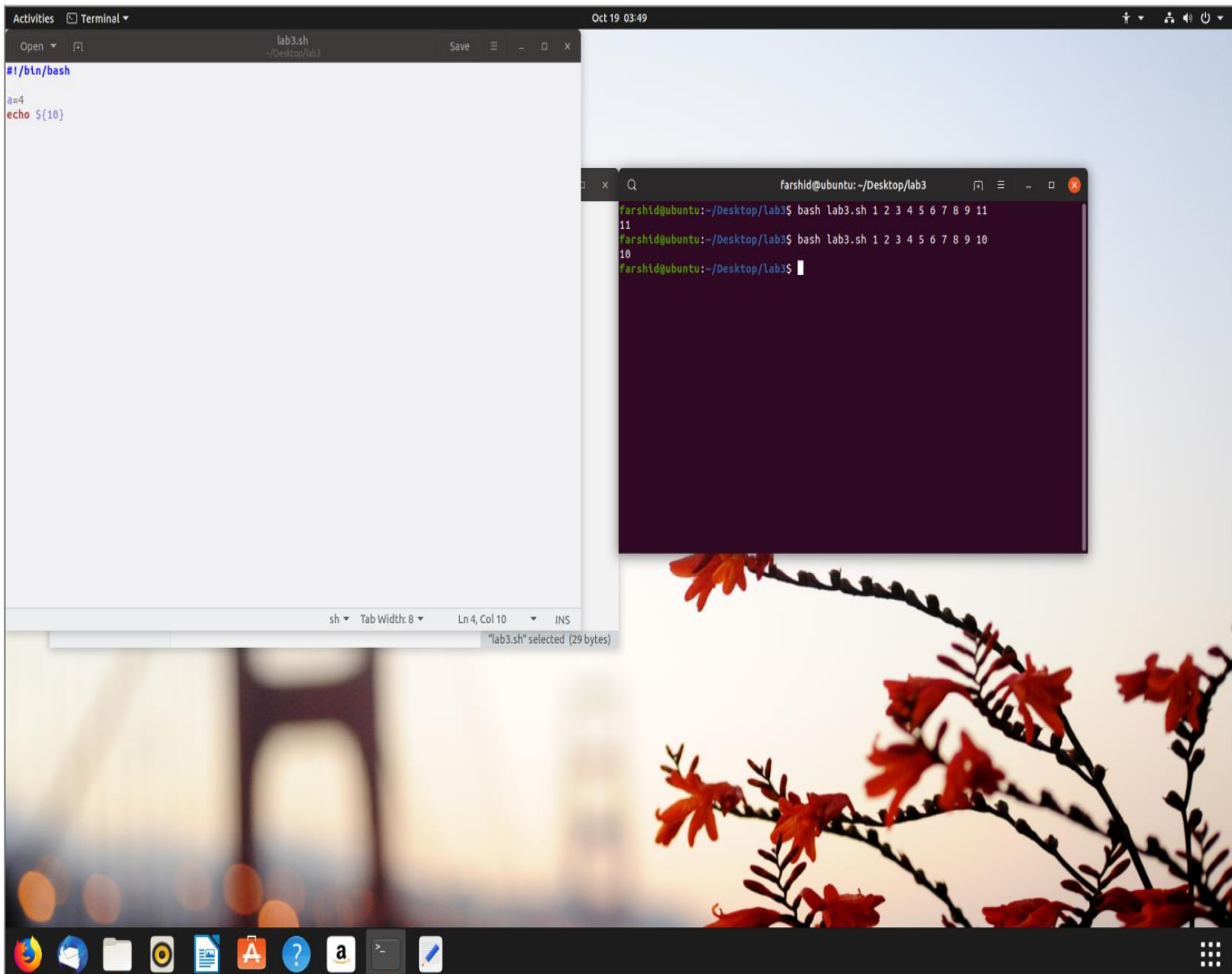
آزمایش 3

فرشید نوشی 9831068

اگر بیش از 10 آرگومان ورودی باشد، چگونه باید به مقدار 10-امین آرگومان دست یافت؟

جواب:

با استفاده از `echo ${10}` ما میتوانیم به مقدار 10مین آرگومان ورودی دست پیدا کنیم.



```
Oct 19 03:49
lab3.sh
~/Desktop/lab3
Save
# /bin/bash
a=4
echo ${10}

sh Tab Width: 8 Ln 4, Col 10 INS
"lab3.sh" selected (29 bytes)

farshid@ubuntu: ~/Desktop/lab3
farshid@ubuntu:~/Desktop/lab3$ bash lab3.sh 1 2 3 4 5 6 7 8 9 11
11
farshid@ubuntu:~/Desktop/lab3$ bash lab3.sh 1 2 3 4 5 6 7 8 9 10
10
farshid@ubuntu:~/Desktop/lab3$
```

با توجه به تصویر بالا مشخص است با `${10}` مشکل حل 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 فایلی ذخیره کنید.

جواب:

در دستور اول که -p دارد p اول کلمه ی prompt میباشد و باعث میشود ما بتوانیم قبل از گرفتن ورودی از کاربر پیامی را در صفحه برایش نمایش دهیم.

Commands

```
{
  read -p 'Type something and hit space: ' '-d ';
  echo "";
  echo "You typed ${REPLY}"
}
```

Output

```
Type something and hit space: something(space)
You typed something
```

در دستور دوم برای -sp ما از دو option داریم استفاده میکنیم یکی prompt کردن و دیگری برای `secret` هستش که `echo` کردن ورودی کاربر را در ترمینال خاموش میکند و متنی را که کاربر در حال چاپ کردن هست را در ترمینال نشان نمیدهد همچنین با prompt نیز قبل از گرفتن ورودی پیغامی را برای کاربر نمایش میدهد.

Commands

```
{
  read -s -p 'Type something I promise to keep it a secret: '
  echo "";
  echo "Your secret is safe with me" ; unset REPLY ;
  echo "${REPLY}"
}
```

Output

```
Type something I promise to keep it a secret:
Your secret is safe with me
```

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

الف- با هم جمع کند و نتیجه را اعلام کند

ب- عدد بزرگتر را نمایش دهد.

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

```
#!/bin/bash
for input in $1 $2
do
    if ! [[ "$input" =~ ^[0-9]+$ ]]; then
        exec >&2; echo "error: Not a number, please enter two valid numbers"; exit 1
    fi
done
let sum=$1+$2
echo "sum is $sum"

if [ "$1" -gt "$2" ]
then
    echo "$1 is greater than the second number"
else
    echo "$2 is greater than or equal the first number"
fi
```

Handwritten Persian text: همیشه در دل عدد بودن ورودی ها (Always being a number in the heart of inputs)

Terminal output:

```
farshid@ubuntu:~/Desktop/lab3$ bash lab32.sh 23 d
error: Not a number, please enter two valid numbers
farshid@ubuntu:~/Desktop/lab3$ bash lab32.sh 23 12
sum is 35
23 is greater than the second number
farshid@ubuntu:~/Desktop/lab3$ bash lab32.sh 12 13
sum is 25
13 is greater than or equal the first number
farshid@ubuntu:~/Desktop/lab3$ bash lab32.sh 12 12
sum is 24
12 is greater than or equal the first number
farshid@ubuntu:~/Desktop/lab3$
```

قسمت اول کد چک میکند که اعداد ورودی رقم غیر 0-9 نداشته باشند و اگر داشتند برنامه متوقف میشود با پیغام مناسب در قسمت دوم حاصل جمع حساب میشود و در قسمت سوم با شرط گذاشتن عدد بزرگ تر چاپ میشود.

۲. ماشین حسابی با استفاده از case طراحی کنید.

```
#!/bin/bash

echo "firstnumber, secondnumber, operator(+./)"
read -ra array

arraylength=${#array[@]}

for ((i=0; i<${arraylength} - 1; i++))
do
    input=${array[$i]}
    if ! [[ "$input" =~ ^[0-9]+$ ]]; then
        echo "error: Not a number, please enter two valid numbers"; exit 1
    fi
done

case ${array[2]} in
    +)
        let sum=${array[0]}+${array[1]}
        echo "sum is $sum"
        ;;
    -)
        let sum=${array[0]}-${array[1]}
        echo "difference is $sum"
        ;;
    .)
        let sum=${array[0]}\*${array[1]}
        echo "product is $sum"
        ;;
    /)
        let sum=${array[0]}/${array[1]}
        echo "ratio is $sum"
        ;;
    *)
        echo "invalid operation"
        ;;
esac
```

The terminal output shows the script being executed with various inputs. It correctly calculates sum, difference, product, and ratio for valid inputs. For invalid inputs (non-numbers or invalid operators), it displays an error message and exits with status 1.

در قسمت اول کد به مانند بالا با یک حلقه روی اعداد ارایه عدد بودن آن ها را چک میکنیم و در ادامه با استفاده از دستور case روی عملیات 4 حالت جمع و تفریق و ضرب و تقسیم را انجام داده ایم و در نهایت اگر حالتی غیر از این 4 مورد بود پیغام خطا میدهیم.

۳. برنامه ای بنویسید که به طور متوالی از کاربر عدد دریافت کند و عددی چاپ کند که ترتیب ارقامش معکوس باشد. مثلا 567 را به صورت 765 چاپ کند. سپس جمع ارقام آن را چاپ کند.

The screenshot shows a Linux desktop environment. On the left, a text editor window titled 'lab34.sh' contains the following shell script:

```
#!/bin/bash

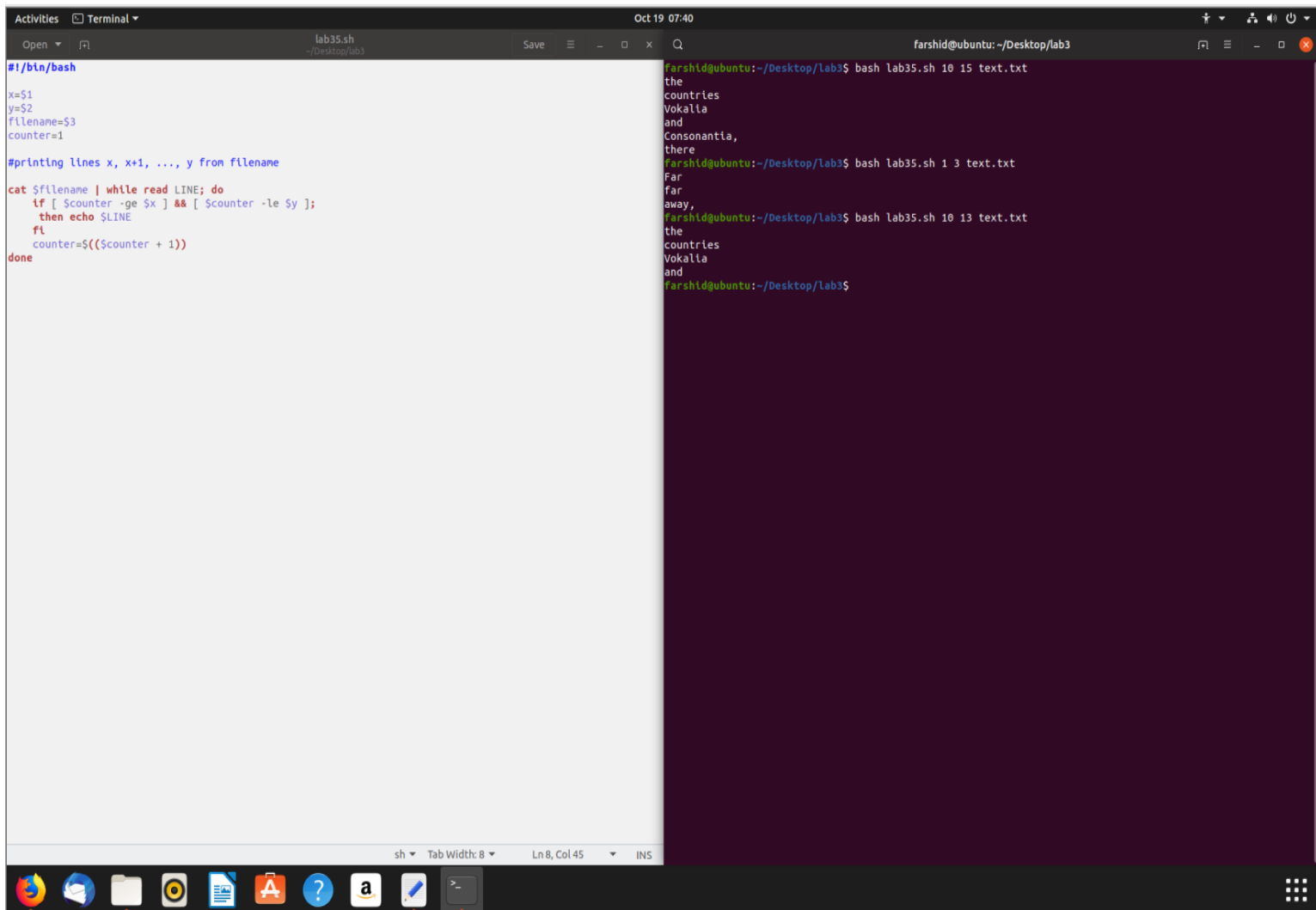
while [ 1 = 1 ]
do
    read -p 'please enter a number: ' number
    len=$(expr length "$number")
    for ((i= $len - 1; i>=0; i--))
    do
        reverse="$reverse${number:$i:1}"
    done
    echo "the reverse is: $reverse"
    reverse=""
done
```

On the right, a terminal window titled 'farshid@ubuntu: ~/Desktop/lab3' shows the script being executed. The output is as follows:

```
farshid@ubuntu:~/Desktop/lab3$ bash lab34.sh
please enter a number: 12
the reverse is: 21
please enter a number: 1234
the reverse is: 4321
please enter a number: 670
the reverse is: 076
please enter a number: 4678901
the reverse is: 1098764
please enter a number: ^C
farshid@ubuntu:~/Desktop/lab3$
```

ابتدا برای اینکه بتوانیم مدام ورودی بگیریم یک While با شرط همیشه درست میسازیم و در داخل آن ابتدا طول آن رشته را حساب میکنیم و سپس با یک حلقه از آخر آن به اول آن حرکت کرده و کاراکتر هایش را در یک رشته ی دیگر میریزیم. (\$number:\$i:1) یک کاراکتر از جایگاه iam را برمیگرداند. و این رشته در نهایت میشود رشته ی برعکس شده و آن را چاپ میکنیم و در نهایت هم اطلاعاتش را برای در بعدی حلقه پاک میکنیم.

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



```
#!/bin/bash
x=$1
y=$2
filename=$3
counter=1

#printing lines x, x+1, ..., y from filename

cat $filename | while read LINE; do
    if [ $counter -ge $x ] && [ $counter -le $y ];
    then echo $LINE
    fi
    counter=$((counter + 1))
done

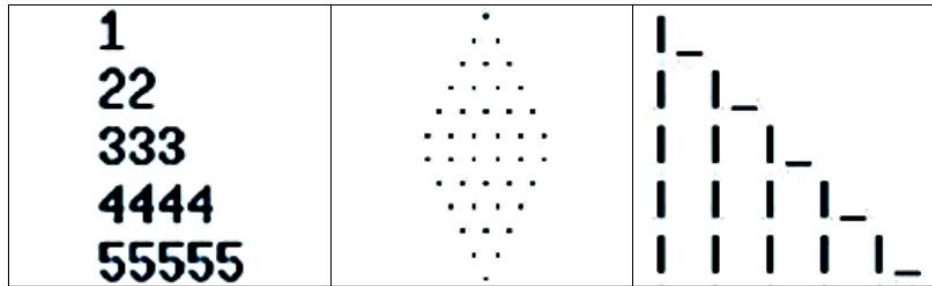
farshid@ubuntu: ~/Desktop/lab3$ bash lab35.sh 10 15 text.txt
the
countries
Vokallia
and
Consonantia,
there
farshid@ubuntu:~/Desktop/lab3$ bash lab35.sh 1 3 text.txt
Far
far
away,
farshid@ubuntu:~/Desktop/lab3$ bash lab35.sh 10 13 text.txt
the
countries
Vokallia
and
farshid@ubuntu:~/Desktop/lab3$
```

با استفاده از دستور cat اینکار را انجام میدهیم به اینصورت که تمامی خط های فایل را یک به یک میخوانیم و در متغیر line میریزیم و با یک شمارنده چک میکنیم که اگر شماره ی خط ما بین مقادیر مطلوبمان بود آن خط را چاپ کند.

The screenshot shows a text editor window titled "Text Editor" with a menu bar containing "Activities", "Text Editor", and a date "Oct 19". The window has a toolbar with "Open", "Save", and window control icons. Two tabs are open: "lab35.sh" and "text.txt". The "text.txt" tab is active and displays a poem in a monospaced font. The poem is a variation of the poem "Paradise Lost" by John Milton. The status bar at the bottom indicates "Plain Text", "Tab Width: 8", "Ln 64, Col 7", and "INS".

```
Far
far
away,
behind
the
word
mountains,
far
from
the
countries
Vokalia
and
Consonantia,
there
live
the
blind
texts.
Separated
they
live
in
Bookmarksgrove
right
at
the
coast
of
the
Semantics,
a large
language
ocean.
A small
river
named
Duden
flows
by
their
place
and
supplies
it
with
the
necessary
regellalia.
It
is
a
paradisematic
country,
in
which
```


۵. برنامه‌ای بنویسید که از کاربر یک عدد بین ۱ و ۳ دریافت کند و شکل مربوط به آن عدد را رسم کند.



```

# /bin/bash

function type1(){
  for((i=1;i<=5;i++)) do
    val=""
    for((j=1;j<=i;j++)) do
      val=${val}i
    done
    echo $val
  done
}

function type2(){
  val=""
  val2=""
  for((i=0;i<=5;i++)) do
    res=""
    for((j=0;j<=5-i;j++)) do
      printf " "
    done
    for((j=5-$i;j<=5+$i;j=$j+2)) do
      printf "i"
    done
    echo ""
  done
  for((i=4;i>=0;i--)) do
    res=""
    for((j=0;j<=5-i;j++)) do
      printf " "
    done
    for((j=5-$i;j<=5+$i;j=$j+2)) do
      printf "i"
    done
    echo ""
  done
}

function type3(){
  val=""
  val2=""
  for((i=0;i<=5;i++)) do
    res=""
    for((j=0;j<=5-i;j++)) do
      printf "${val2}"
    done
    echo ${val}
  done
}

if [ $1 -eq 1 ]; then
  type1
elif [ $1 -eq 2 ]; then
  type2
elif [ $1 -eq 3 ]; then
  type3
fi

```

با سه تابع این کار را انجام می‌دهیم، هر تابع دو حلقه ی تو در تو دارد که کارهای چاپ کردن را انجام می‌دهند در تابع اول ما در خط ۱ام باید رقم ۱ را چاپ کنیم. در تابع سوم در خط ۱ ما باید ۱ بار علامت " | " را چاپ می‌کردیم و در آخر هر خط هم علامت " | _ " را چاپ می‌کنیم.

در تابع دوم الگو کمی پیچیده بود، در خط i ام که i از 5 کوچک تر بود ما تا جایگاه اولین نقطه space میگذاریم تا به اولین نقطه برسیم (تعداد $5-i$ عدد space) بعد به اندازه $i+1$ بار " " چاپ میکنیم و سپس به خط بعد میرویم با اینکار هرم بالایی شکل چاپ میشود و برای درست کردن حرم پایینی حلقه ی اول که برای هرم بالایی بود را برعکس دوباره حرکت میکنیم تا هرم برعکس هم ساخته شود. (فقط این بار i از 4 تا صفر می آید).

سوال امتیازی:

ماشین حسابی برای اعداد حقیقی بنویسید

```
Activities Terminal Oct 19 09:51 lab3 farshid@ubuntu: ~/Desktop/lab3
Open
#!/bin/bash

echo "firstnumber, secondnumber, operator(+..)/"
read -ra array

arraylength=${#array[@]}

case ${array[2]} in
+)
    sum=$(echo "scale=4; ${array[0]} + ${array[1]}" | bc)
    printf "sum is "
    printf %.10f\n $sum
    ;;
-)
    sum=$(echo "scale=4; ${array[0]} - ${array[1]}" | bc)
    printf "difference is "
    printf %.10f\n $sum
    ;;
.)
    sum=$(echo "scale=4; ${array[0]} * ${array[1]}" | bc)
    printf "product is "
    printf %.10f\n $sum
    ;;
/)
    sum=$(echo "scale=4; ${array[0]} / ${array[1]}" | bc)
    printf "ratio is "
    printf %.10f\n $sum
    ;;
*)
    echo "invalid operation"
    ;;
esac

farshid@ubuntu:~/Desktop/lab3$ bash lab37.sh
firstnumber, secondnumber, operator(+..)/
1.12 3.14 +
sum is 4.2600000000
farshid@ubuntu:~/Desktop/lab3$ bash lab37.sh
firstnumber, secondnumber, operator(+..)/
1.12 3.14 -
difference is -2.0200000000
farshid@ubuntu:~/Desktop/lab3$ bash lab37.sh
firstnumber, secondnumber, operator(+..)/
1.12 3.14 .
product is 3.5168000000
farshid@ubuntu:~/Desktop/lab3$ bash lab37.sh
firstnumber, secondnumber, operator(+..)/
1.12 3.14 /
ratio is 0.3560000000
farshid@ubuntu:~/Desktop/lab3$ bash lab37.sh
firstnumber, secondnumber, operator(+..)/
1.12 3.14 %
invalid operation
farshid@ubuntu:~/Desktop/lab3$
```

چون در bash نمیتوانستیم عملیات های اعشاری را انجام دهیم با کمک bc اینکار را انجام دادیم $scale=4$ دقت اعشار را مشخص میکند و در ادامه عملیات ماشین حساب را با کمک bc برای هر یک از 4 عمل انجام میدهم. در اینجا به یک روش دیگر نیز طول آرایه را حساب کردیم البته با آن کاری نداشتیم، ماشین حساب

به کمک دستور case متوجه نوع عملیات میشود برای چاپ کردن جواب نیز از printf با 10 رقم دقت استفاده کردیم.