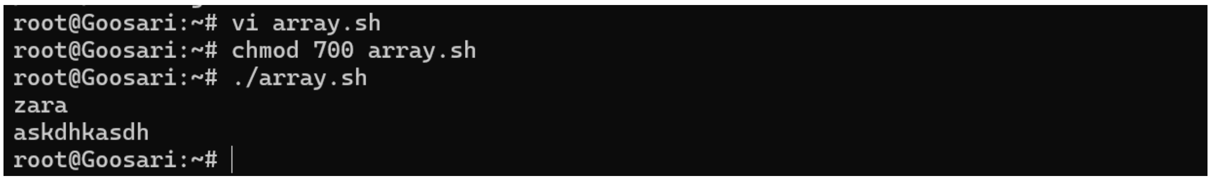
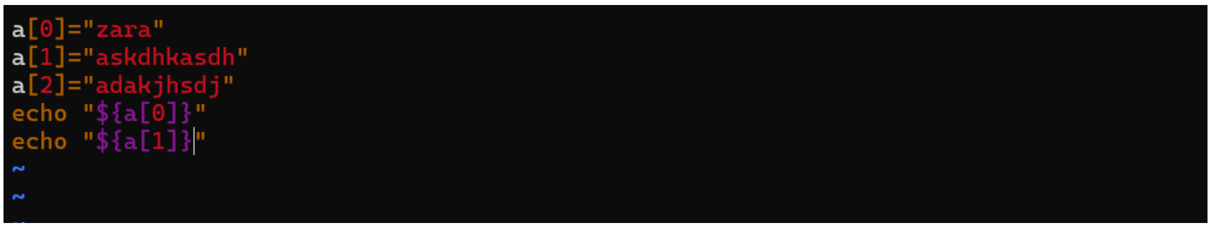
<u>19-02-2025:</u>

<u>Array:</u>



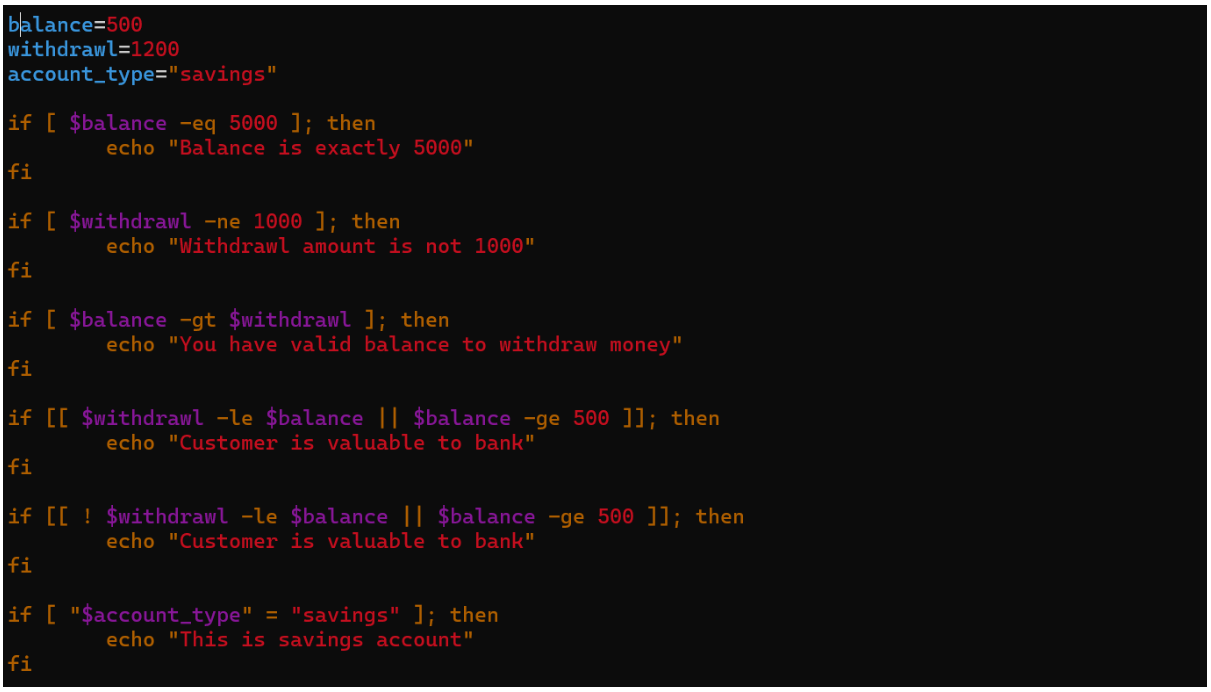
```
root@Goosari:~# vi array.sh
root@Goosari:~# chmod 700 array.sh
root@Goosari:~# ./array.sh
zara
askdhkasdh
root@Goosari:~#
```



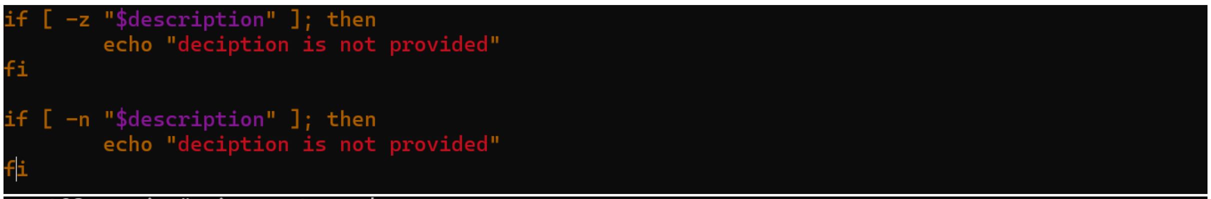
```
a[0]="zara"
a[1]="askdhkasdh"
a[2]="adakjhsdj"
echo "${a[0]}"
echo "${a[1]}"
~
~
```

Create an array with three elements and print the first two.

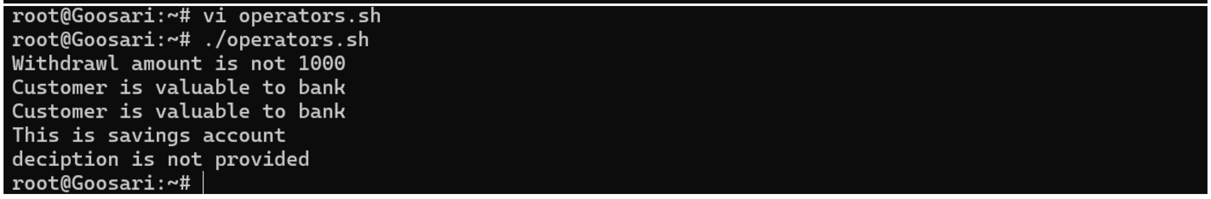
<u>Operators:</u>



```
balance=500
withdrawl=1200
account_type="savings"

if [ $balance -eq 5000 ]; then
        echo "Balance is exactly 5000"
fi

if [ $withdrawl -ne 1000 ]; then
        echo "Withdrawl amount is not 1000"
fi

if [ $balance -gt $withdrawl ]; then
        echo "You have valid balance to withdraw money"
fi

if [[ $withdrawl -le $balance || $balance -ge 500 ]]; then
        echo "Customer is valuable to bank"
fi

if [[ ! $withdrawl -le $balance || $balance -ge 500 ]]; then
        echo "Customer is valuable to bank"
fi

if [ "$account_type" = "savings" ]; then
        echo "This is savings account"
fi
```



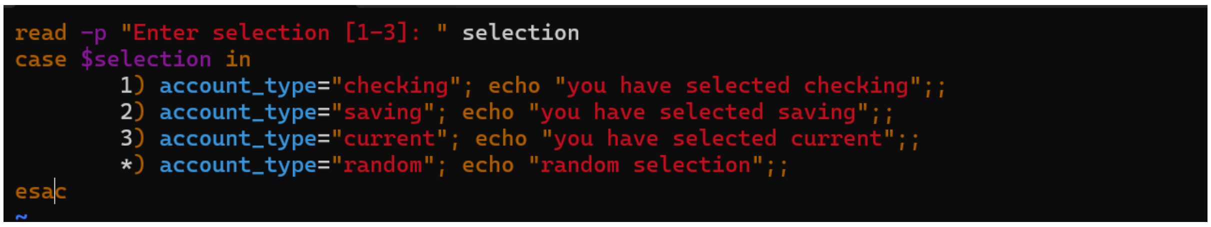
```
if [ -z "$description" ]; then
        echo "deciption is not provided"
fi

if [ -n "$description" ]; then
        echo "deciption is not provided"
fi
```



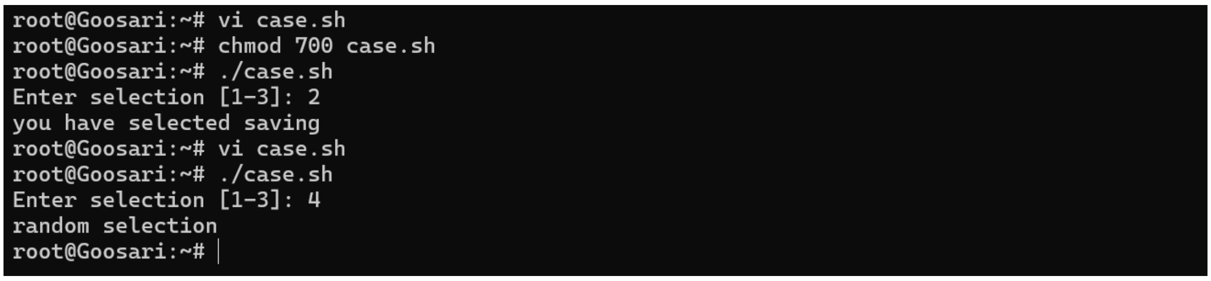
```
root@Goosari:~# vi operators.sh
root@Goosari:~# ./operators.sh
Withdrawl amount is not 1000
Customer is valuable to bank
Customer is valuable to bank
This is savings account
deciption is not provided
root@Goosari:~#
```

Operator Type	Operators	Description
Arithmetic	+ - * / % **	Addition, subtraction, multiplication, division, modulus, exponentiation
Assignment	= += -= *= /= %=	Assign values to variables with optional arithmetic operations
Comparison (Integers)	-eq -ne -gt -lt -ge -le	Equal, not equal, greater than, less than, greater or equal, less or equal
Comparison (Strings)	== != < > -z -n	Equal, not equal, less than, greater than, empty string check, non-empty check
Logical	`&&	
Bitwise	`&	^ ~ << >>`
File Test	-e -f -d -r -w -x	Exists, is a file, is a directory, readable, writable, executable
Redirection	> >> < <<	Output overwrite, append, input, here-document
Pipes	`	
Control Flow	`; ;; & &&	
Brace Expansion	{}	Expands ranges (e.g., {1..5} → 1 2 3 4 5)

Case:

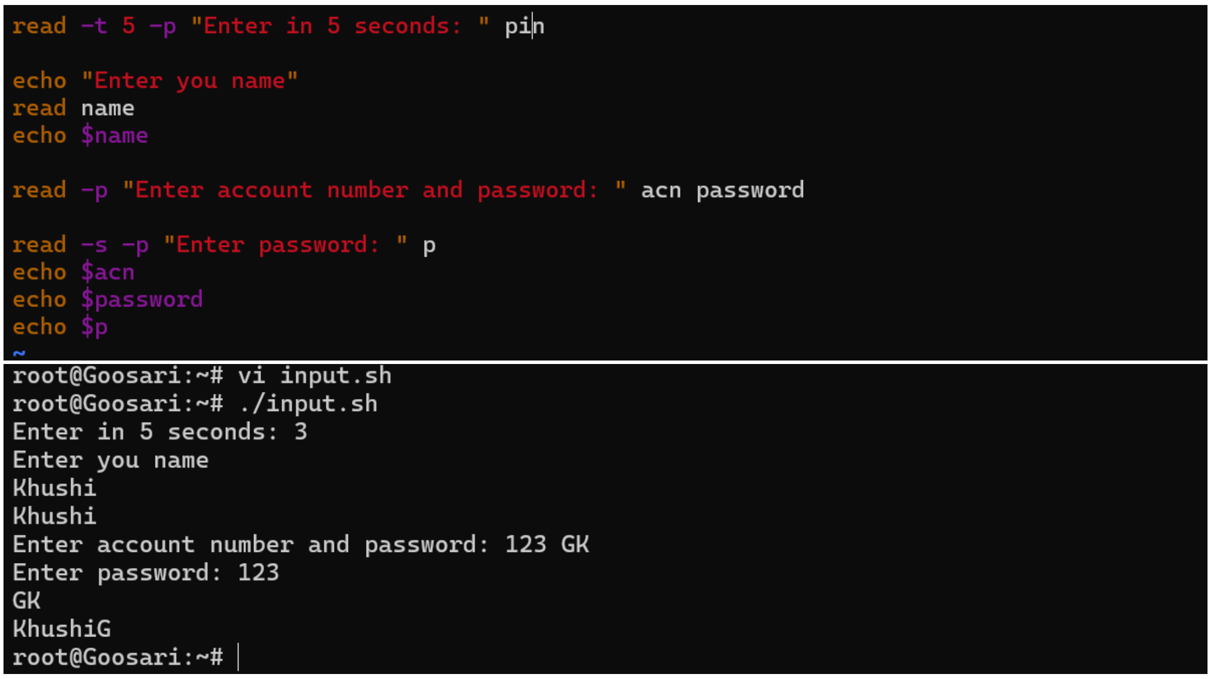


```
read -p "Enter selection [1-3]: " selection
case $selection in
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
esac
```



```
root@Goosari:~# vi case.sh
root@Goosari:~# chmod 700 case.sh
root@Goosari:~# ./case.sh
Enter selection [1-3]: 2
you have selected saving
root@Goosari:~# vi case.sh
root@Goosari:~# ./case.sh
Enter selection [1-3]: 4
random selection
root@Goosari:~#
```

Input:

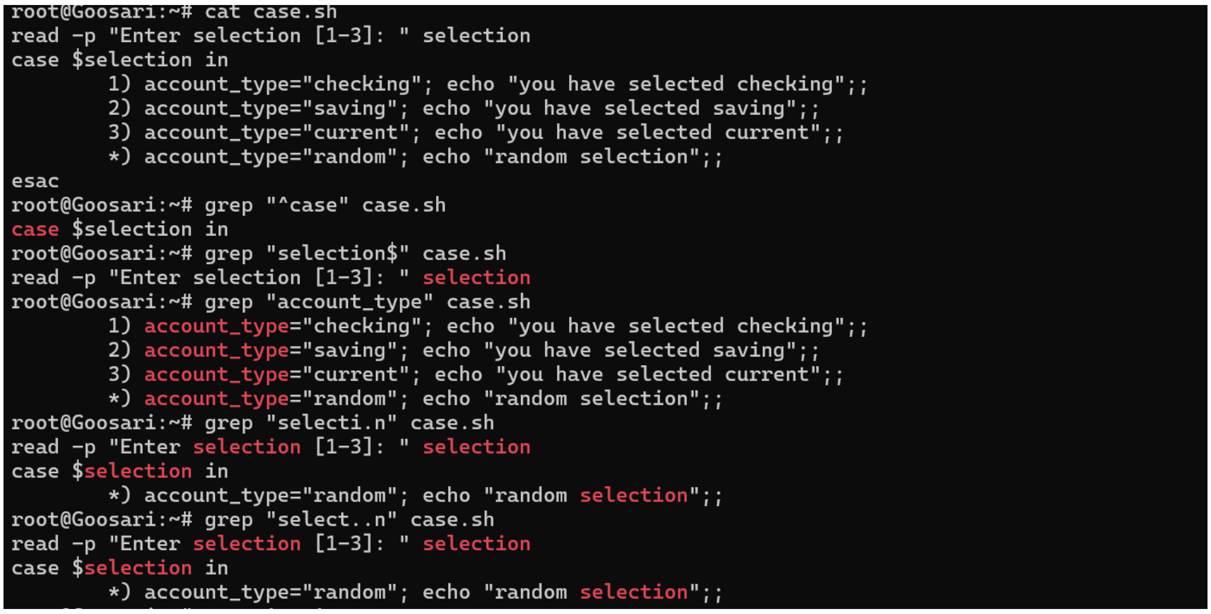


```
read -t 5 -p "Enter in 5 seconds: " pin

echo "Enter you name"
read name
echo $name

read -p "Enter account number and password: " acn password

read -s -p "Enter password: " p
echo $acn
echo $password
echo $p
~
```

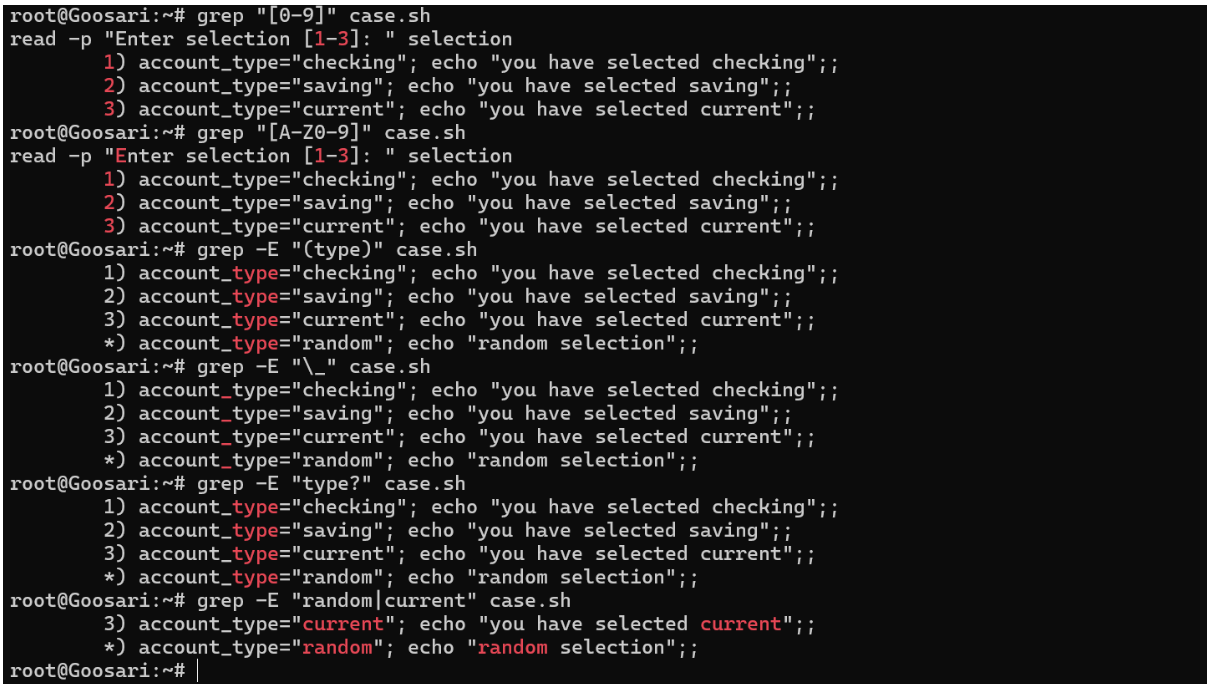
```
root@Goosari:~# vi input.sh
root@Goosari:~# ./input.sh
Enter in 5 seconds: 3
Enter you name
Khushi
Khushi
Enter account number and password: 123 GK
Enter password: 123
GK
KhushiG
root@Goosari:~#
```

- read: Reads user input
- -t 5: Sets a timeout of 5 sec. It exits if input not provided
- -p: display a prompt
- -s: silent mode (useful for passwords)

Grep: searches for and filters lines in a file or input that match a specified pattern.



```
root@Goosari:~# cat case.sh
read -p "Enter selection [1-3]: " selection
case $selection in
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
esac
root@Goosari:~# grep "^case" case.sh
case $selection in
root@Goosari:~# grep "selection$" case.sh
read -p "Enter selection [1-3]: " selection
root@Goosari:~# grep "account_type" case.sh
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
root@Goosari:~# grep "selecti.n" case.sh
read -p "Enter selection [1-3]: " selection
case $selection in
        *) account_type="random"; echo "random selection";;
root@Goosari:~# grep "select..n" case.sh
read -p "Enter selection [1-3]: " selection
case $selection in
        *) account_type="random"; echo "random selection";;
```



```
root@Goosari:~# grep "[0-9]" case.sh
read -p "Enter selection [1-3]: " selection
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
root@Goosari:~# grep "[A-Z0-9]" case.sh
read -p "Enter selection [1-3]: " selection
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
root@Goosari:~# grep -E "(type)" case.sh
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
root@Goosari:~# grep -E "\_" case.sh
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
root@Goosari:~# grep -E "type?" case.sh
        1) account_type="checking"; echo "you have selected checking";;
        2) account_type="saving"; echo "you have selected saving";;
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
root@Goosari:~# grep -E "random|current" case.sh
        3) account_type="current"; echo "you have selected current";;
        *) account_type="random"; echo "random selection";;
root@Goosari:~# |
```

^ : matches the start of a line

$: matches the end of a line.

-E : enables extended regex (ERE) for advanced patterns. (OR | operator)

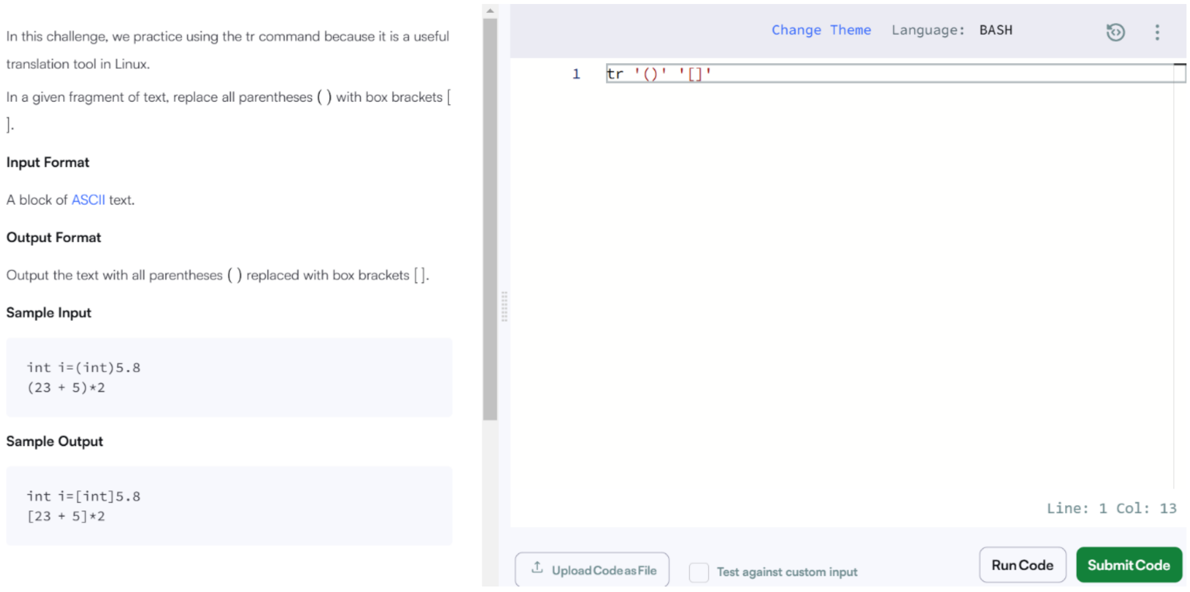
() : groups expressions.

[] : class of chars

\ : escapes special characters.

? : makes the preceding character or group optional

PRACTICE PROBLEMS: (Hackerrank)



In this challenge, we practice using the tr command because it is a useful translation tool in Linux.

In a given fragment of text, replace all parentheses ( ) with box brackets [ ].

Input Format

A block of ASCII text.

Output Format

Output the text with all parentheses ( ) replaced with box brackets [ ].

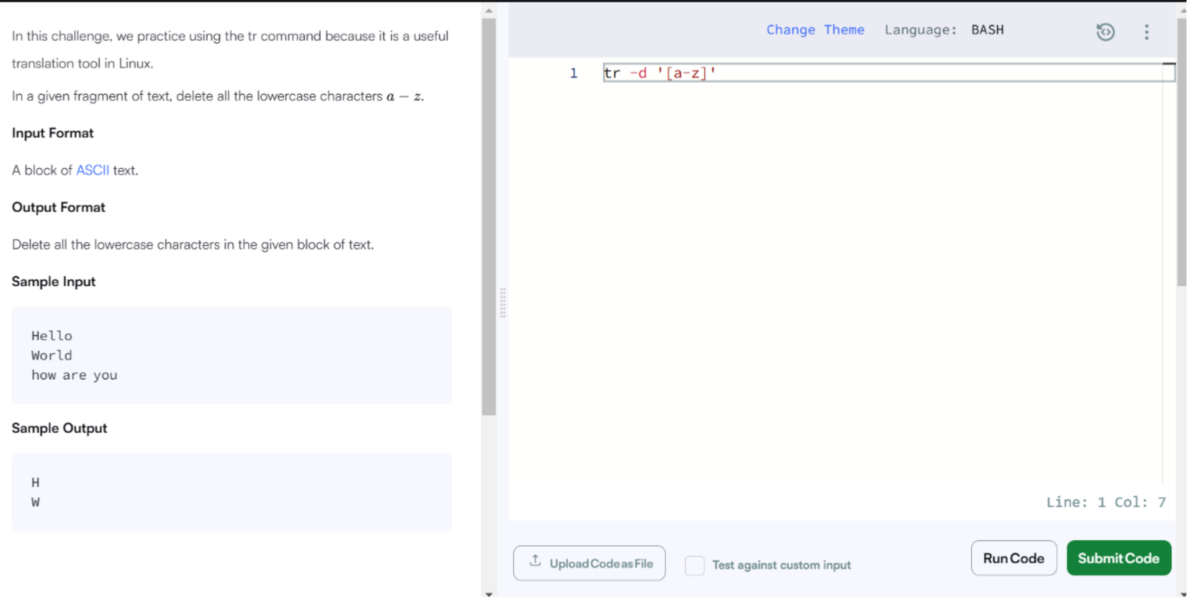
Sample Input

```
int i=(int)5.8
(23 + 5)*2
```

Sample Output

```
int i=[int]5.8
[23 + 5]*2
```

```bash
tr '()' '[]'
```



In this challenge, we practice using the tr command because it is a useful translation tool in Linux.

In a given fragment of text, delete all the lowercase characters $a - z$.

Input Format

A block of ASCII text.

Output Format

Delete all the lowercase characters in the given block of text.

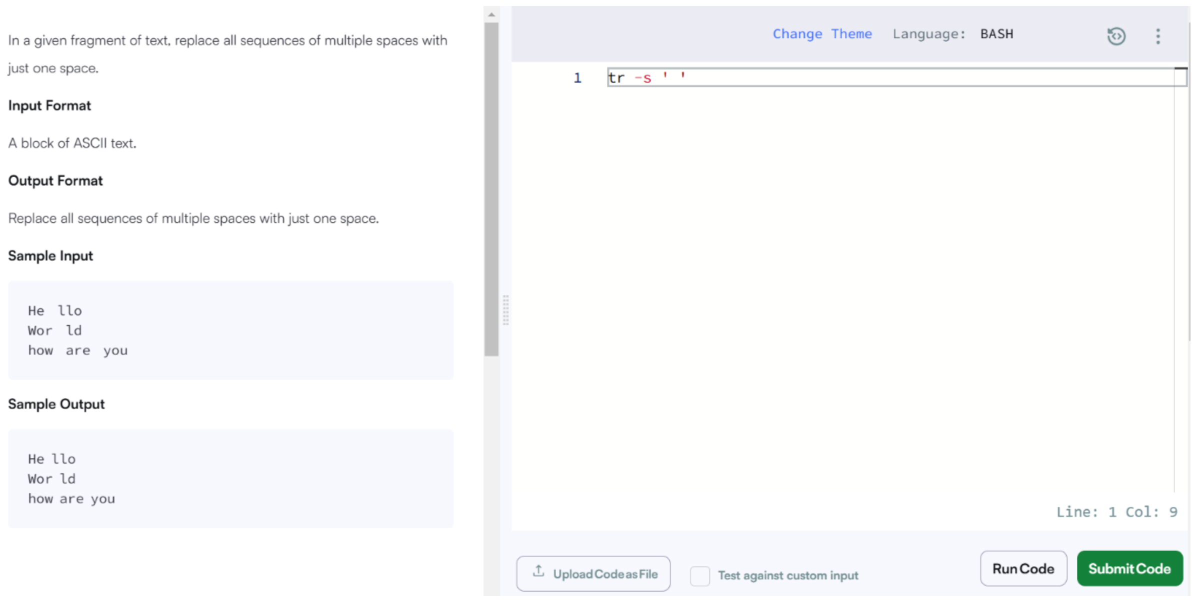
Sample Input

```
Hello
World
how are you
```

Sample Output

```
H
W
```

```bash
tr -d '[a-z]'
```


In a given fragment of text, replace all sequences of multiple spaces with just one space.

Input Format

A block of ASCII text.

Output Format

Replace all sequences of multiple spaces with just one space.

Sample Input

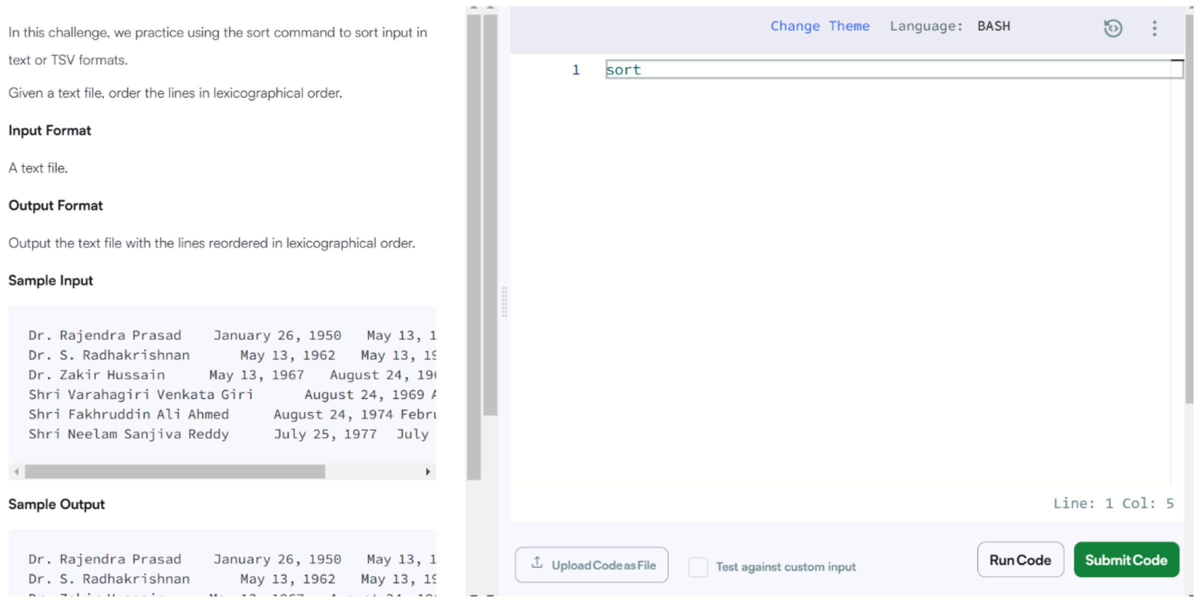
```
He  llo
Wor  ld
how  are  you
```

Sample Output

```
He llo
Wor ld
how are you
```

Language: BASH

```bash
tr -s ' '
```


In this challenge, we practice using the sort command to sort input in text or TSV formats.

Given a text file, order the lines in lexicographical order.

Input Format

A text file.

Output Format

Output the text file with the lines reordered in lexicographical order.

Sample Input

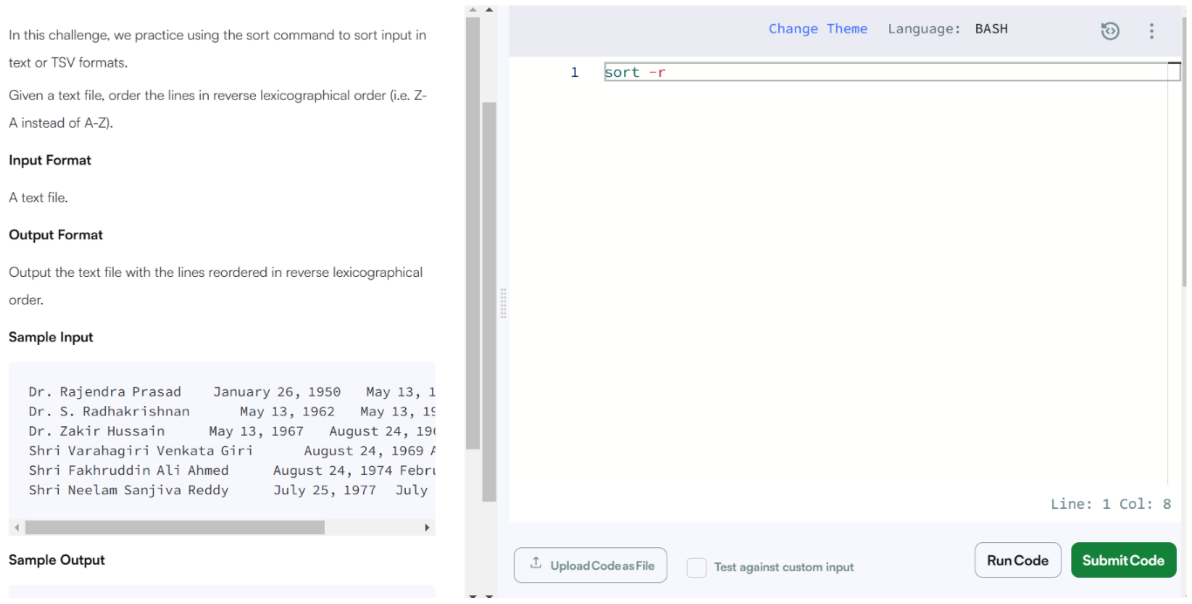
```
Dr. Rajendra Prasad    January 26, 1950    May 13, 1
Dr. S. Radhakrishnan       May 13, 1962    May 13, 19
Dr. Zakir Hussain      May 13, 1967    August 24, 19
Shri Varahagiri Venkata Giri       August 24, 1969 A
Shri Fakhruddin Ali Ahmed      August 24, 1974 Febru
Shri Neelam Sanjiva Reddy      July 25, 1977   July
```

Sample Output

```
Dr. Rajendra Prasad    January 26, 1950    May 13, 1
Dr. S. Radhakrishnan       May 13, 1962    May 13, 19
```

Language: BASH

```bash
sort
```


In this challenge, we practice using the sort command to sort input in text or TSV formats.

Given a text file, order the lines in reverse lexicographical order (i.e. Z-A instead of A-Z).

Input Format

A text file.

Output Format

Output the text file with the lines reordered in reverse lexicographical order.

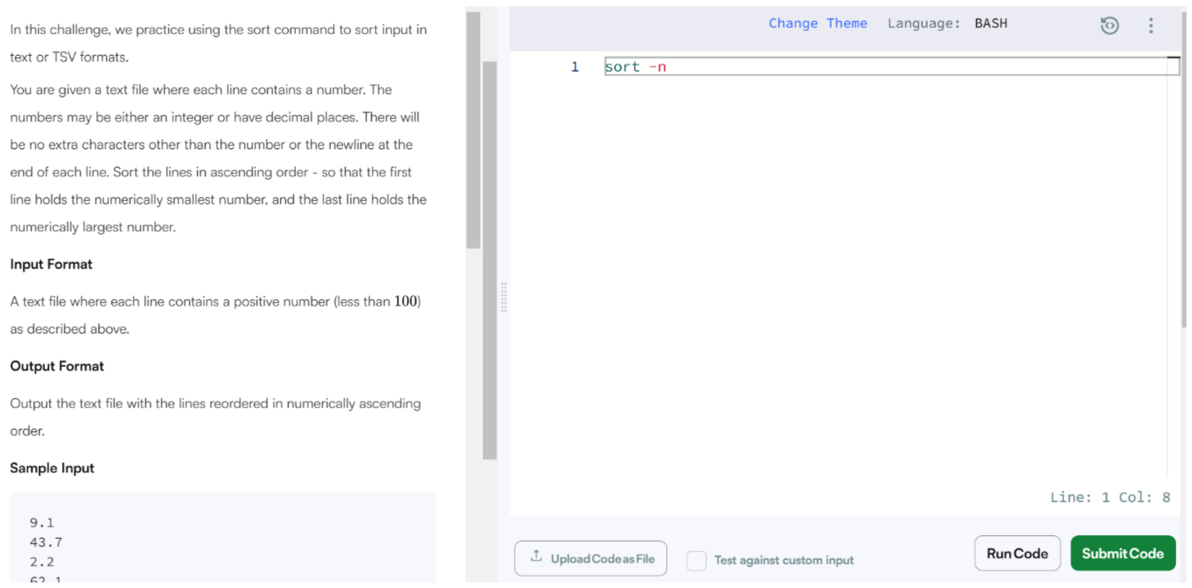
Sample Input

```
Dr. Rajendra Prasad    January 26, 1950    May 13, 1
Dr. S. Radhakrishnan       May 13, 1962    May 13, 19
Dr. Zakir Hussain      May 13, 1967    August 24, 19
Shri Varahagiri Venkata Giri       August 24, 1969 A
Shri Fakhruddin Ali Ahmed      August 24, 1974 Febru
Shri Neelam Sanjiva Reddy      July 25, 1977   July
```

Sample Output

Language: BASH

```bash
sort -r
```



In this challenge, we practice using the sort command to sort input in text or TSV formats.

You are given a text file where each line contains a number. The numbers may be either an integer or have decimal places. There will be no extra characters other than the number or the newline at the end of each line. Sort the lines in ascending order - so that the first line holds the numerically smallest number, and the last line holds the numerically largest number.

Input Format

A text file where each line contains a positive number (less than 100) as described above.

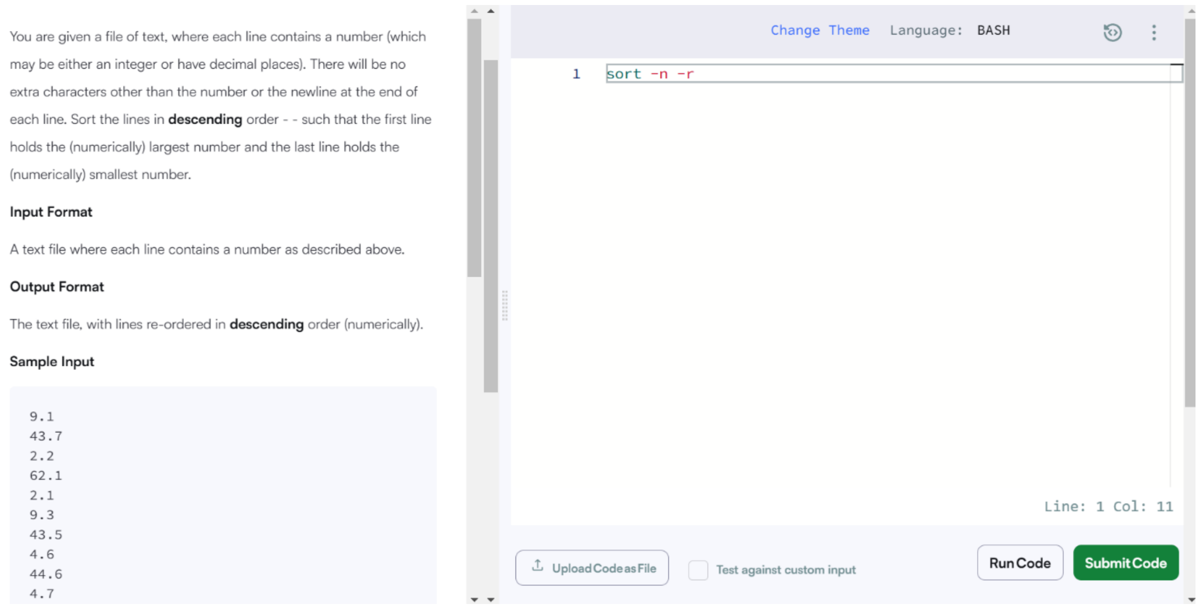
Output Format

Output the text file with the lines reordered in numerically ascending order.

Sample Input

```
9.1
43.7
2.2
62.1
```

```bash
sort -n
```



You are given a file of text, where each line contains a number (which may be either an integer or have decimal places). There will be no extra characters other than the number or the newline at the end of each line. Sort the lines in **descending** order - - such that the first line holds the (numerically) largest number and the last line holds the (numerically) smallest number.

Input Format

A text file where each line contains a number as described above.

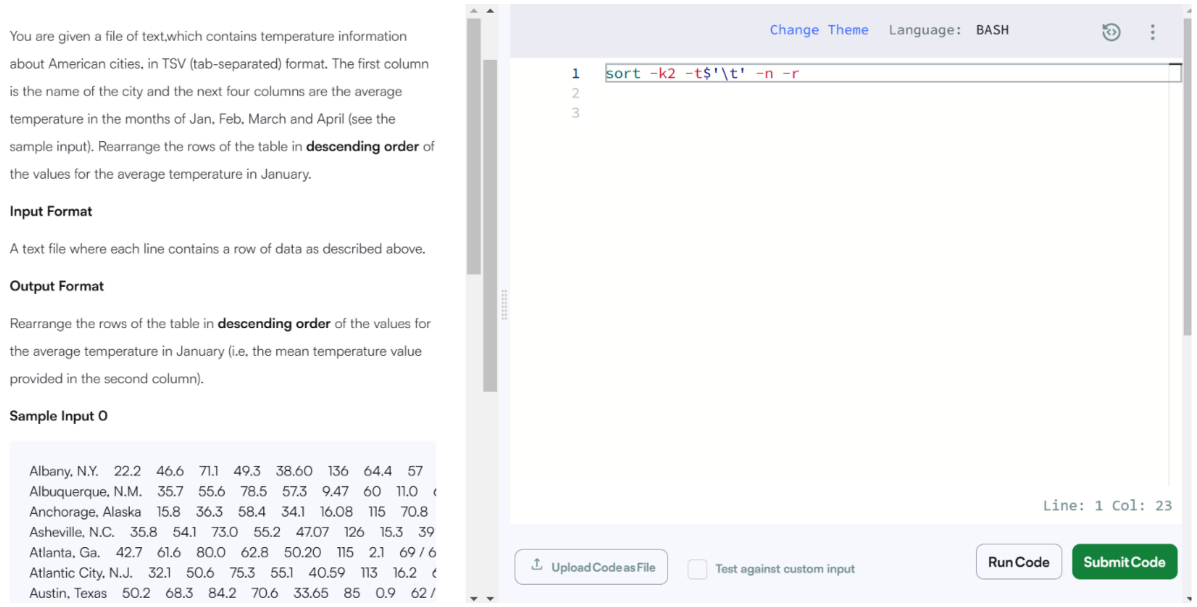
Output Format

The text file, with lines re-ordered in **descending** order (numerically).

Sample Input

```
9.1
43.7
2.2
62.1
2.1
9.3
43.5
4.6
44.6
4.7
```

```bash
sort -n -r
```



You are given a file of text,which contains temperature information about American cities, in TSV (tab-separated) format. The first column is the name of the city and the next four columns are the average temperature in the months of Jan, Feb, March and April (see the sample input). Rearrange the rows of the table in **descending order** of the values for the average temperature in January.

Input Format

A text file where each line contains a row of data as described above.

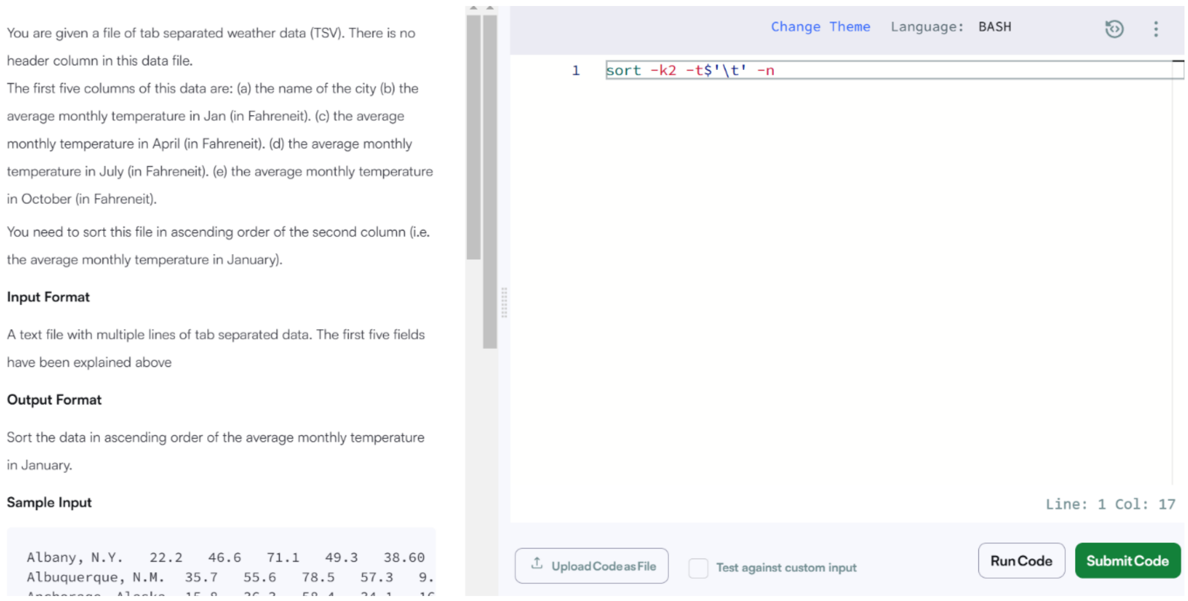
Output Format

Rearrange the rows of the table in **descending order** of the values for the average temperature in January (i.e. the mean temperature value provided in the second column).

Sample Input 0

```
Albany, N.Y.    22.2    46.6    71.1    49.3    38.60    136    64.4    57
Albuquerque, N.M.    35.7    55.6    78.5    57.3    9.47    60    11.0
Anchorage, Alaska    15.8    36.3    58.4    34.1    16.08    115    70.8
Asheville, N.C.    35.8    54.1    73.0    55.2    47.07    126    15.3    39
Atlanta, Ga.    42.7    61.6    80.0    62.8    50.20    115    2.1    69 / 6
Atlantic City, N.J.    32.1    50.6    75.3    55.1    40.59    113    16.2
Austin, Texas    50.2    68.3    84.2    70.6    33.65    85    0.9    62 /
```

```bash
sort -k2 -t$'\t' -n -r
```



You are given a file of tab separated weather data (TSV). There is no header column in this data file.

The first five columns of this data are: (a) the name of the city (b) the average monthly temperature in Jan (in Fahreneit). (c) the average monthly temperature in April (in Fahreneit). (d) the average monthly temperature in July (in Fahreneit). (e) the average monthly temperature in October (in Fahreneit).

You need to sort this file in ascending order of the second column (i.e. the average monthly temperature in January).

Input Format

A text file with multiple lines of tab separated data. The first five fields have been explained above

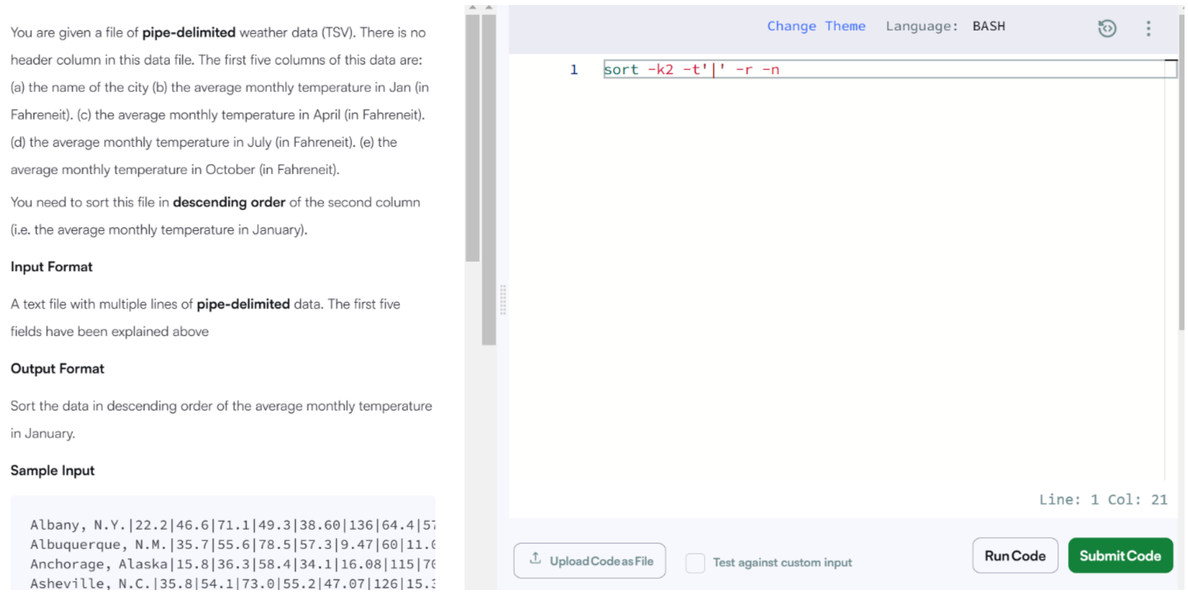
Output Format

Sort the data in ascending order of the average monthly temperature in January.

Sample Input

```
Albany, N.Y.    22.2    46.6    71.1    49.3    38.60
Albuquerque, N.M.   35.7    55.6    78.5    57.3    9.
```

```bash
sort -k2 -t$'\t' -n
```



You are given a file of **pipe-delimited** weather data (TSV). There is no header column in this data file. The first five columns of this data are: (a) the name of the city (b) the average monthly temperature in Jan (in Fahreneit). (c) the average monthly temperature in April (in Fahreneit). (d) the average monthly temperature in July (in Fahreneit). (e) the average monthly temperature in October (in Fahreneit).

You need to sort this file in **descending order** of the second column (i.e. the average monthly temperature in January).

Input Format

A text file with multiple lines of **pipe-delimited** data. The first five fields have been explained above

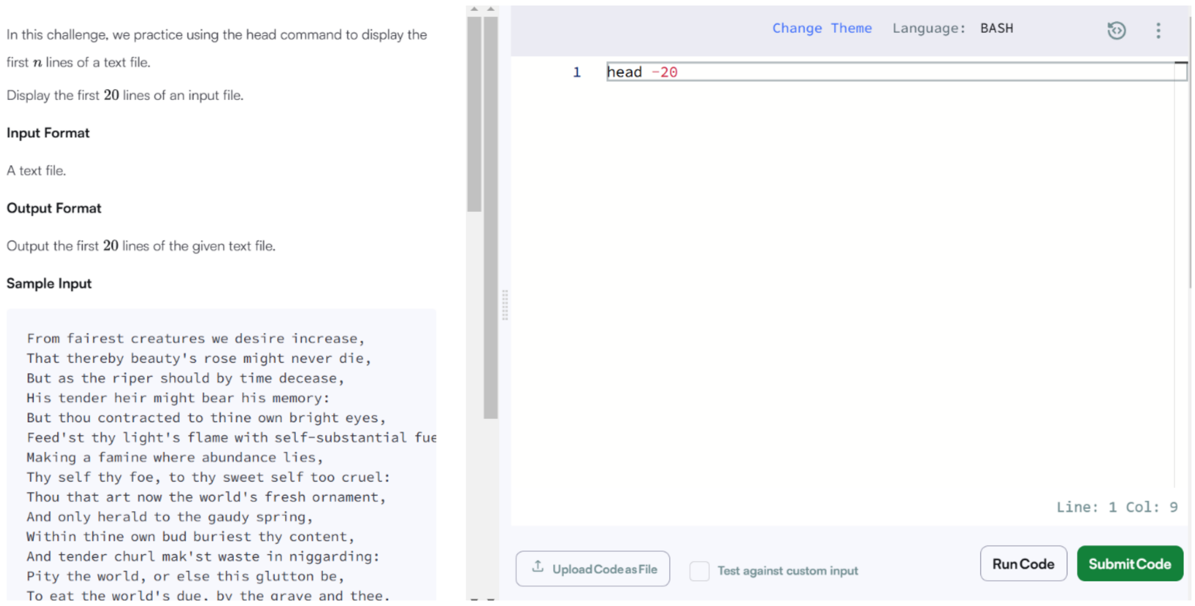
Output Format

Sort the data in descending order of the average monthly temperature in January.

Sample Input

```
Albany, N.Y.|22.2|46.6|71.1|49.3|38.60|136|64.4|57
Albuquerque, N.M.|35.7|55.6|78.5|57.3|9.47|60|11.6
Anchorage, Alaska|15.8|36.3|58.4|34.1|16.08|115|70
Asheville, N.C.|35.8|54.1|73.0|55.2|47.07|126|15.3
```

```bash
sort -k2 -t'|' -r -n
```



In this challenge, we practice using the head command to display the first n lines of a text file.

Display the first 20 lines of an input file.

Input Format

A text file.

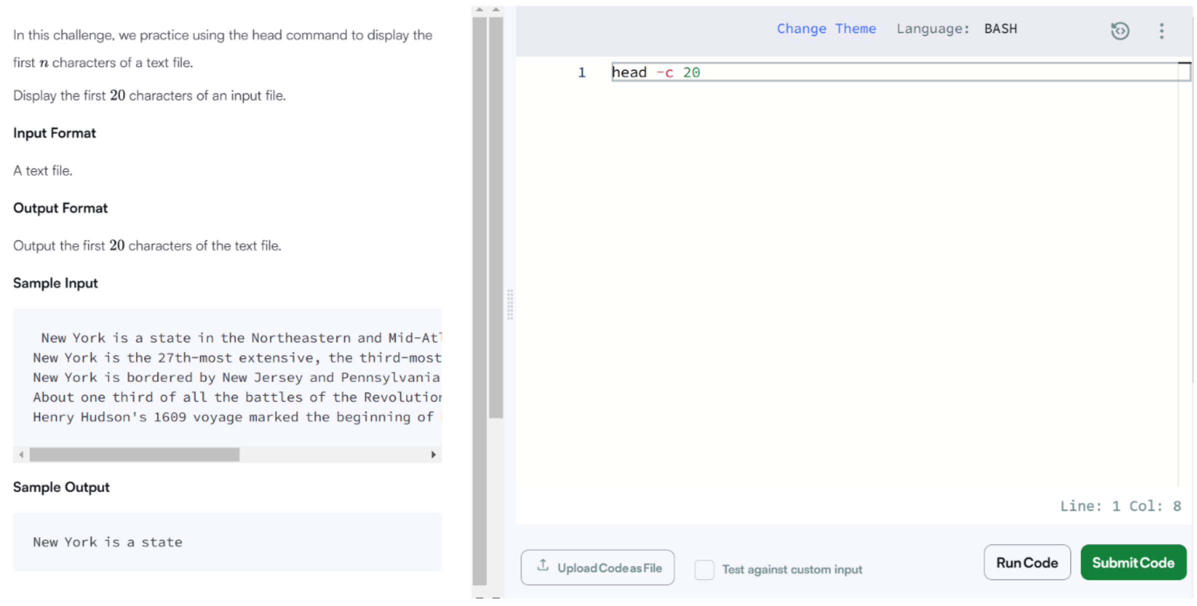
Output Format

Output the first 20 lines of the given text file.

Sample Input

```
From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fue
Making a famine where abundance lies,
Thy self thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
Pity the world, or else this glutton be,
To eat the world's due, by the grave and thee.
```

```bash
head -20
```


In this challenge, we practice using the head command to display the first n characters of a text file.

Display the first 20 characters of an input file.

Input Format

A text file.

Output Format

Output the first 20 characters of the text file.

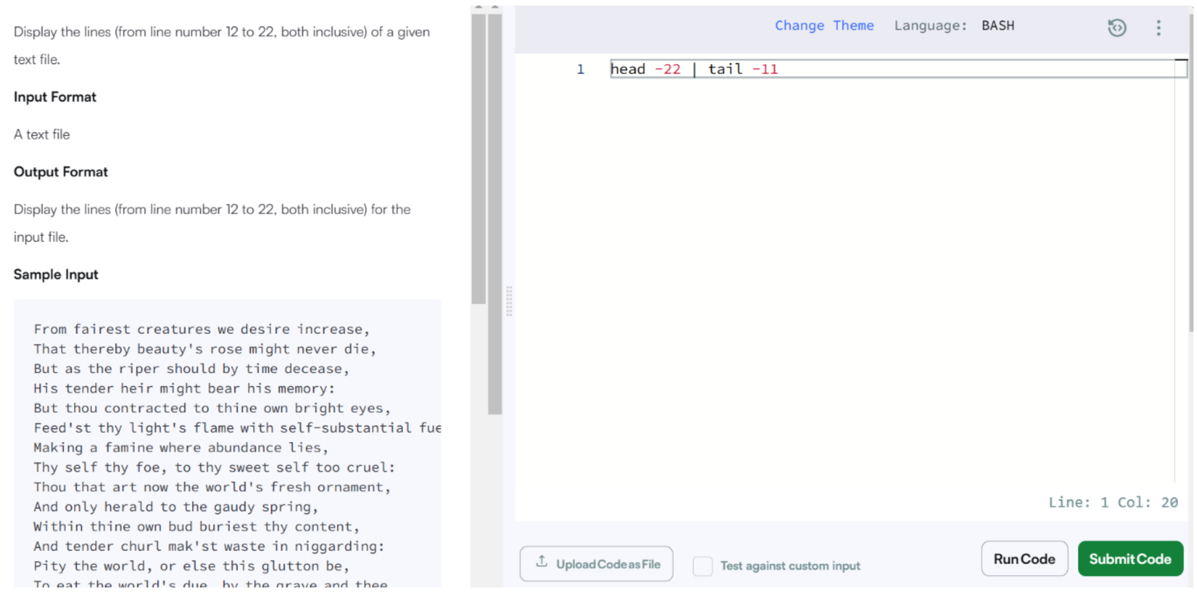
Sample Input

```
 New York is a state in the Northeastern and Mid-At
New York is the 27th-most extensive, the third-most
New York is bordered by New Jersey and Pennsylvania
About one third of all the battles of the Revolutio
Henry Hudson's 1609 voyage marked the beginning of
```

Sample Output

```
New York is a state
```

```bash
head -c 20
```


Display the lines (from line number 12 to 22, both inclusive) of a given text file.

Input Format

A text file

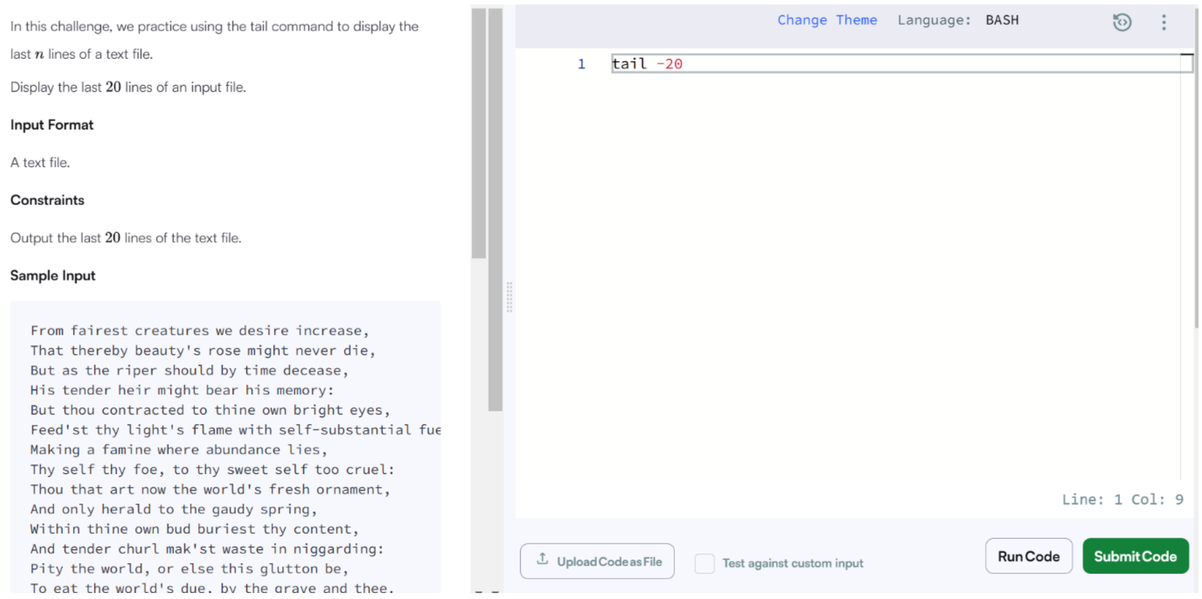
Output Format

Display the lines (from line number 12 to 22, both inclusive) for the input file.

Sample Input

```
From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fue
Making a famine where abundance lies,
Thy self thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
Pity the world, or else this glutton be,
To eat the world's due, by the grave and thee.
```

```bash
head -22 | tail -11
```


In this challenge, we practice using the tail command to display the last n lines of a text file.

Display the last 20 lines of an input file.

Input Format

A text file.

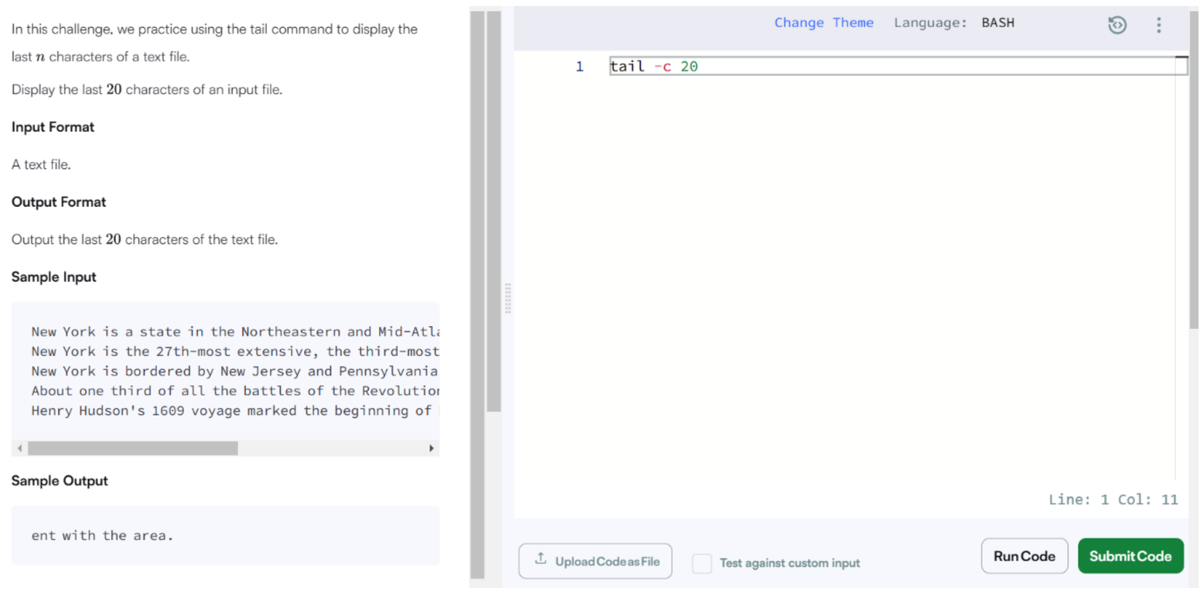
Constraints

Output the last 20 lines of the text file.

Sample Input

```
From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fue
Making a famine where abundance lies,
Thy self thy foe, to thy sweet self too cruel:
Thou that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
Pity the world, or else this glutton be,
To eat the world's due, by the grave and thee,
```

```bash
tail -20
```



In this challenge, we practice using the tail command to display the last n characters of a text file.

Display the last 20 characters of an input file.

Input Format

A text file.

Output Format

Output the last 20 characters of the text file.

Sample Input

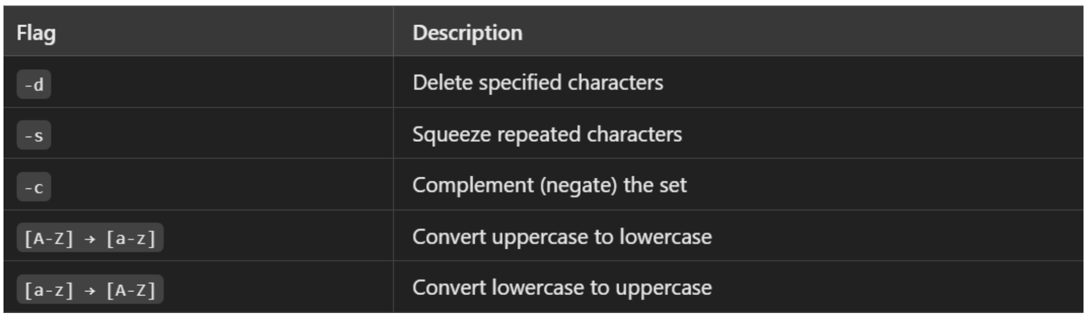
```
New York is a state in the Northeastern and Mid-Atl
New York is the 27th-most extensive, the third-most
New York is bordered by New Jersey and Pennsylvania
About one third of all the battles of the Revolutior
Henry Hudson's 1609 voyage marked the beginning of
```

Sample Output

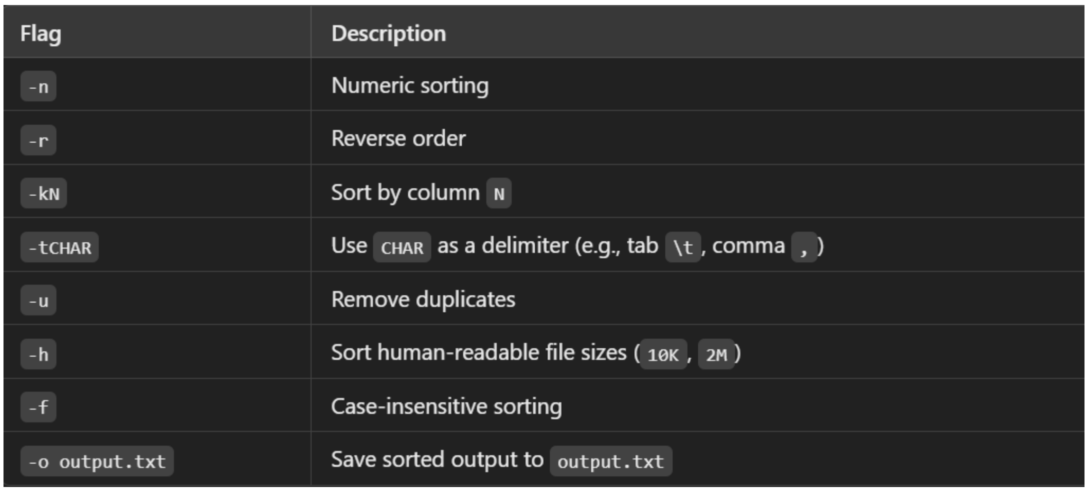
```
ent with the area.
```

```bash
tail -c 20
```

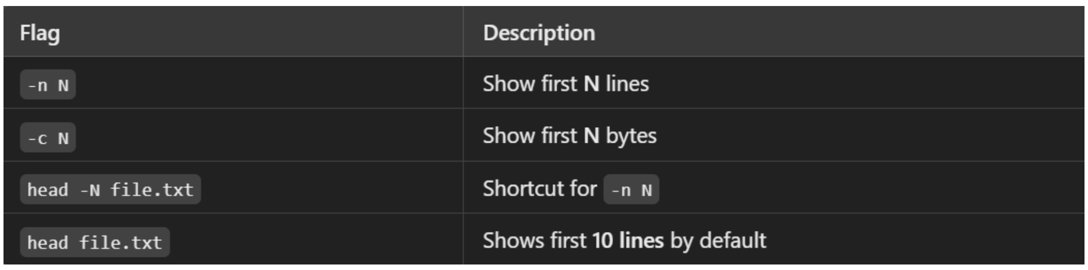
Line: 1 Col: 11
Upload Code as File
Test against custom input
Run Code
Submit Code



Flag	Description
`-d`	Delete specified characters
`-s`	Squeeze repeated characters
`-c`	Complement (negate) the set
`[A-Z] → [a-z]`	Convert uppercase to lowercase
`[a-z] → [A-Z]`	Convert lowercase to uppercase



Flag	Description
`-n`	Numeric sorting
`-r`	Reverse order
`-kN`	Sort by column `N`
`-tCHAR`	Use `CHAR` as a delimiter (e.g., tab `\t`, comma `,`)
`-u`	Remove duplicates
`-h`	Sort human-readable file sizes (`10K`, `2M`)
`-f`	Case-insensitive sorting
`-o output.txt`	Save sorted output to `output.txt`



Flag	Description
`-n N`	Show first **N** lines
`-c N`	Show first **N** bytes
`head -N file.txt`	Shortcut for `-n N`
`head file.txt`	Shows first **10 lines** by default