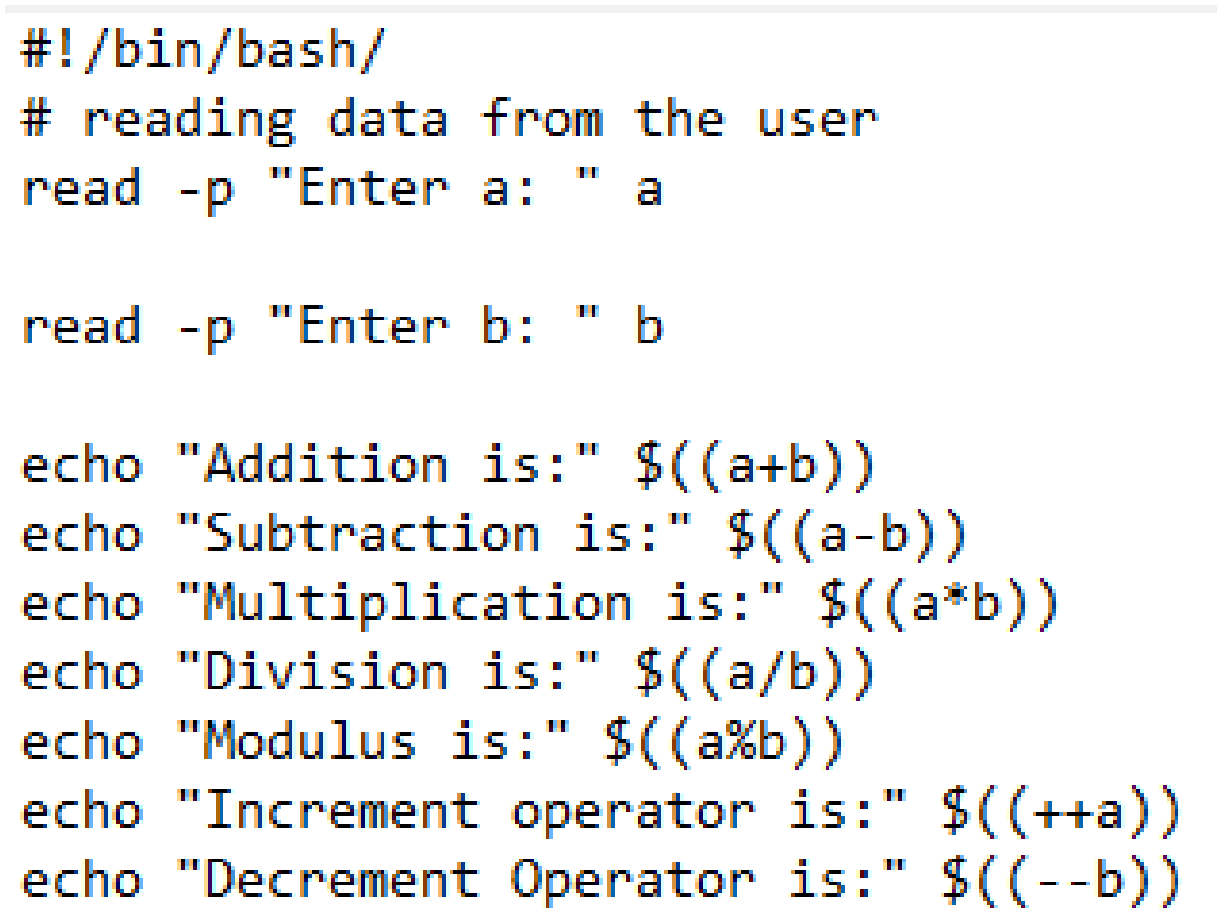
- **There are 5 basic operators in bash/shell scripting:**
  - Arithmetic Operators
  - Relational Operators
  - Boolean Operators
  - Bitwise Operators
  - File Test Operators

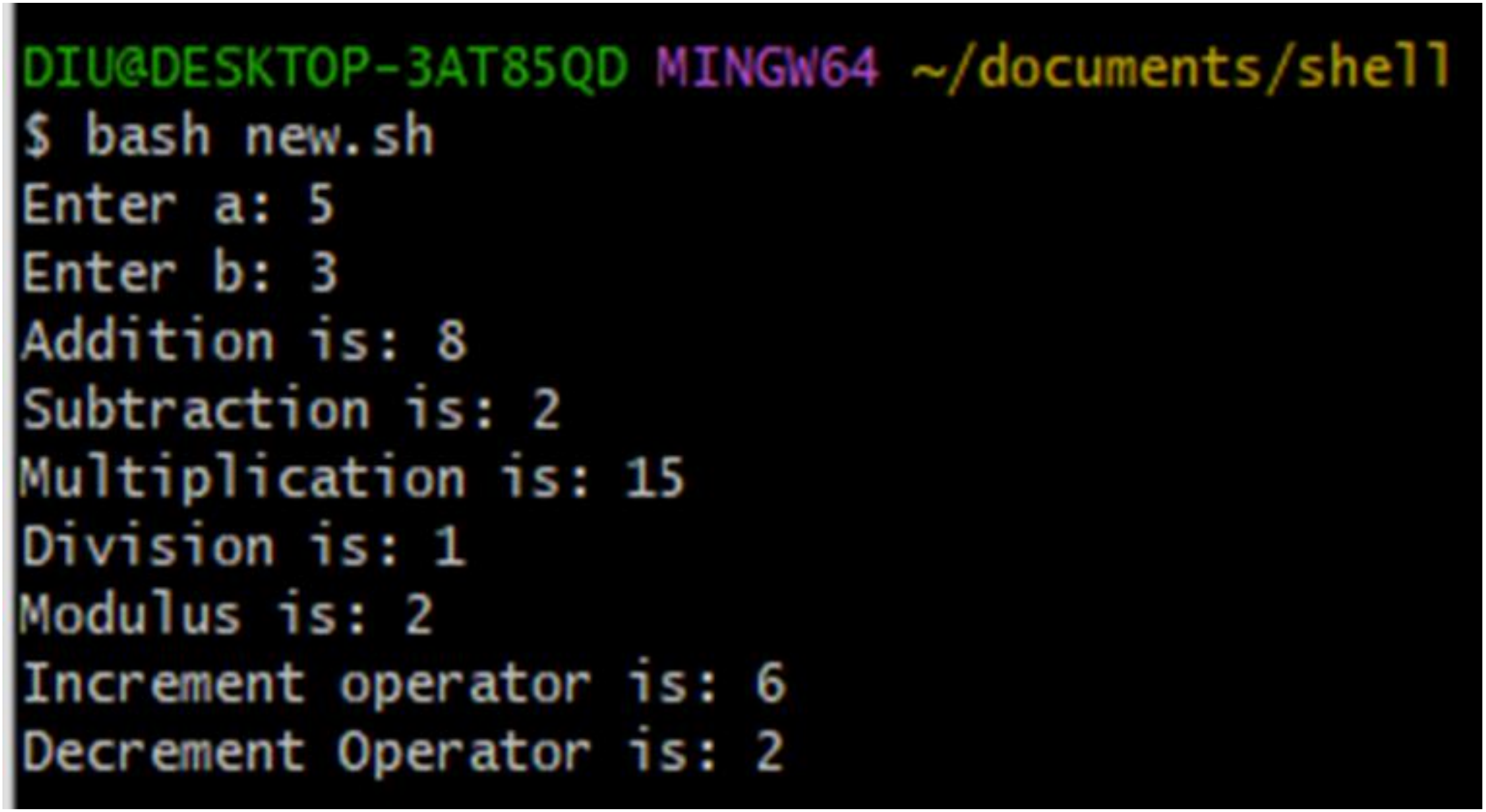
1. **Arithmetic Operators**: These operators are used to perform normal arithmetic/mathematical operations. There are 7 arithmetic operators:

- **Addition (+)**: Binary operation used to add two operands.
- **Subtraction (-)**: Binary operation used to subtract two operands.
- **Multiplication (*)**: Binary operation used to multiply two operands.
- **Division (/)**: Binary operation used to divide two operands.
- **Modulus (%)**: Binary operation used to find remainder of two operands.
- **Increment Operator (++)**: Unary operator used to increase the value of operand by one.
- **Decrement Operator (- -)**: Unary operator used to decrease the value of a operand by one

- **the syntax for arithmetic expressions can be done Using (( ))**
- This is the preferred method for arithmetic operations:
  - $((expression))



```
#!/bin/bash/
# reading data from the user
read -p "Enter a: " a

read -p "Enter b: " b

echo "Addition is:" $((a+b))
echo "Subtraction is:" $((a-b))
echo "Multiplication is:" $((a*b))
echo "Division is:" $((a/b))
echo "Modulus is:" $((a%b))
echo "Increment operator is:" $((++a))
echo "Decrement Operator is:" $((--b))
```



```
DIU@DESKTOP-3AT85QD MINGW64 ~/documents/shell
$ bash new.sh
Enter a: 5
Enter b: 3
Addition is: 8
Subtraction is: 2
Multiplication is: 15
Division is: 1
Modulus is: 2
Increment operator is: 6
Decrement Operator is: 2
```

2. Relational Operators: Relational operators are those operators which define the relation between two operands. They give either true or false depending upon the relation. They are of 6 types:

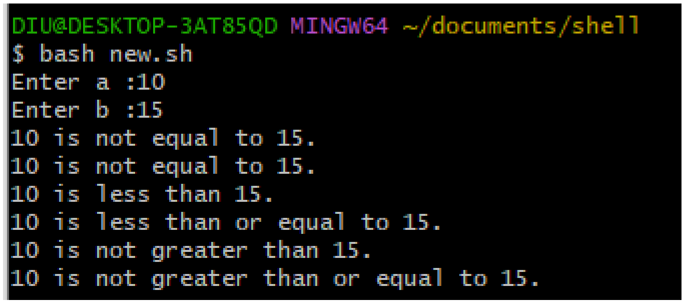
- **‘==’ Operator**: Double equal to operator compares the two operands. Its returns true is they are equal otherwise returns false.
- **‘!=’ Operator**: Not Equal to operator return true if the two operands are not equal otherwise it returns false.
- **‘<‘ Operator**: Less than operator returns true if first operand is less than second operand otherwise returns false.
- **‘<=’ Operator**: Less than or equal to operator returns true if first operand is less than or equal to second operand otherwise returns false
- **‘>’ Operator**: Greater than operator return true if the first operand is greater than the second operand otherwise return false.
- **‘>=’ Operator**: Greater than or equal to operator returns true if first operand is greater than or equal to second operand otherwise returns false

- **The Syntax for if-else**

```
if [ condition ]
 then
    # Commands to execute if condition is true
else
    # Commands if none of the above conditions are true
fi  (In Bash scripting, fi is used to signify the end of an if
    statement.)
```

```
#!/bin/bash

#reading data from the user
read -p "Enter a :" a
read -p "Enter b :" b

if(( $a==$b ))
then
    echo "$a is equal to $b."
else
    echo "$a is not equal to $b."
fi

if(( $a!=$b ))
then
    echo "$a is not equal to $b."
else
    echo "$a is equal to $b."
fi
```

```
if(( $a<$b ))
then
    echo "$a is less than $b."
else
    echo "$a is not less than $b."
fi

if(( $a<=$b ))
then
    echo "$a is less than or equal to $b."
else
    echo "$a is not less than or equal to $b."
fi

if(( $a>$b ))
then
    echo "$a is greater than $b."
else
    echo "$a is not greater than $b."
fi

if(( $a>=$b ))
then
    echo "$a is greater than or equal to $b."
else
    echo "$a is not greater than or equal to $b."
fi
```



```
DIU@DESKTOP-3AT85QD MINGW64 ~/documents/shell
$ bash new.sh
Enter a :10
Enter b :15
10 is not equal to 15.
10 is not equal to 15.
10 is less than 15.
10 is less than or equal to 15.
10 is not greater than 15.
10 is not greater than or equal to 15.
```