1. *Number Data Types*

A) What data type is the number 2? How about 20? 200? Keep adding zeros and watch the data type change until it reaches BigInteger. Then do the same for 2.0

=> For def X= 2 it shows class java.math.Integer by calling function getclass()

**def** x = 20000000000000000000

println x

println x.getClass()

\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*

20000000000000000000

class java.math.BigInteger

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**def** x = 2.0

println x

println x.getClass()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2.0

class java.math.BigDecimal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

B) Declare a variable x of type def and assign it the sum of 1 and 1.5. What is the resulting data type?

=>

**def** x

println x=1+1.5

println x.getClass()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*

2.5

class java.math.BigDecimal

C) What do you get when you divide 5 by 2? What is the resulting data type? If you wanted to do integer division (no remainder), what method would you call?

=>

**def** x

println x=5/2

println x.getClass()

\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*

2.5

class java.math.BigDecimal

1. *Wrapper Classes*

From the associated wrapper classes, find the min and max values for the Java primitives: byte, short, int, long, float, double.

**def** b = 10

println b

println b.getClass().getName()

println Byte.***MIN\_VALUE***

println Byte.***MAX\_VALUE***

print "======================="

//short

**short** s = 1000

println b

println Short.***MIN\_VALUE***

println Short.***MAX\_VALUE***

println "======================="

//long

**long** l= 123

println l

println Long.***MIN\_VALUE***

println Long.***MAX\_VALUE***

println "========================"

//int

**int** i= 123

println i

println Integer.***MIN\_VALUE***

println Integer.***MAX\_VALUE***

println "========================"

//float

**float** f = 258.123897

println f

println Float.***MIN\_VALUE***

println Float.***MAX\_VALUE***

println "========================"

//double

**double** d = 4567

println d

println Double.***MIN\_VALUE***

println Double.***MAX\_VALUE***

println "========================"

1. *2s Complement*

Create a byte variable with its maximum value. What do you get when you add 1 to it?

**package** first\_groovy

**byte** b = 10

println b

println b.getClass().getName()

println x=Byte.***MAX\_VALUE***+1

println x

println x.getClass()

\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*

10

java.lang.Byte

128

128

class java.lang.Integer

1. *Strings and GroovyStrings*

A)How many characters are in the string "Hello, Groovy!"?

**package** first\_groovy

**def** name = "Hello"

println name.length()

\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*

5.

B) Define a string variable containing a name. Print a hello statement with your name using string concatenation, then using a Groovy string.

=>

**package** first\_groovy

**def** name = "Shivani"

println name

println "Hello " + "$name"

\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*

Shivani

Hello Shivani

C) Demonstrate that "racecar" is a palindrome by comparing it to its reverse. Do the same with "Bob", removing case sensitivity first.

=>

**def** Palindrome(String str)

{

str = str.toLowerCase();

println str == str.split("").reverse().join("");

print "$str"

}

Palindrome("RACECAR")

\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*

true

racecar

D) Define a string variable containing the sentence, "Hello, World. How are you?". Split the sentence into an array using the split method. Count the number of words. Do the same using the tokenize method.

**=>**

**package** first\_groovy

**def** name = "Hello World, How are you?"

println name.split("")

println name.length()

\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*

[H, e, l, l, o, , W, o, r, l, d, ,, , H, o, w, , a, r, e, , y, o, u, ?]

25

\*\*\*\*\*\*\*\*TOKENIZED\*\*\*\*\*\*\*\*\*\*\*

//Tokenize

n= name.tokenize()

println n

println n.size()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*

[Hello, World,, How, are, you?]

5

E) Using the same sentence, use array notation (square brackets) to print the substring "World".

=>

**package** first\_groovy

**def** name = "Hello World"

n= name.tokenize()

println n

println n[1]

\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*

[Hello,World]

World

F) Use array notation to print the last word, but reversed.

=>**def** name = "Hello World"

n= name.tokenize()

println n

println n[1].reverse()

\*\*\*\*\*OUTPUT\*\*\*\*\*

[Hello, World]

dlroW

1. *Prime Numbers*

Write a method called isPrime that takes an integer argument and returns a boolean. Determine whether the number is prime by dividing it by all numbers from 2 up to one less than the number.

That limit is too high, of course. How high do you have to check to be sure whether you've gone far enough?

**package** oops

**def** isPrime(**int** a)

{

**def** i = 1

**for** (i ; i<= a/2; i++)

{

**if** (a%2 == 0)

{

**return** **false**

}

**else**

{

**return** **true**

}

}

}

**int** number = 17

s = isPrime(number)

println s

println s.getClass()

\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*

**true**

**class** java.lang.Boolean

1. *Sorting Strings*

Create a list of strings. Sort them alphabetically. Sort them by length. Sort them by length in descending order.( *Advanced:* Sort by length, then sort equal length strings alphabetically)

**package** first\_groovy

**def** words = ["ball", "cat", "doggie","apple"]

println words

//sorting alphabetically

words = words.toSorted()

println words

//sorted by size

words = words.sort{it.size()}

println words

//sorted by size in descending order

words = words.sort{-it.size()}

println words

//sorted by size and alphabetically

words = words.toSorted().sort{it.size()}

println words

\*\*\*\*\*\*\*\*\*\*\*\*OutPut\*\*\*\*\*\*\*\*\*\*

[ball, cat, doggie, apple]

[apple, ball, cat, doggie]

[cat, ball, apple, doggie]

[doggie, apple, ball, cat]

[cat, ball, apple, doggie]

1. *Processing a list of numbers*

Create a list of numbers. Add them together. First double each number, then add them up. Compute their average.

**package** first\_groovy

**def** add = [1,2,3,4,5]

println add

add1 = add.sum()

println add1

add = add.collect{it.multiply(2)}

println add

add2 = add.sum()

println add2

add = add.average()

println add

\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*

[1, 2, 3, 4, 5]

15

[2, 4, 6, 8, 10]

30

6

1. *Reading a web page*

Using the Groovy JDK, access your home page and display the source code. Print the length of each line of the home page.

1. *Closures as a filter*

Create a list of numbers. Print all elements greater than zero.

**def** add = [1,2,3,-1,-20]

println add.findIndexValues { it.value > 0 }

\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*

[0,1,2]

1. *Multi-line strings*

Make a multi-line string. Compute the number of vowels on each line.

**package** first\_groovy

String str = "In this program, our task is to count the total number of vowels and consonants present in the given string."+

"As we know that, the characters a, e, i, o, u are known as vowels in the English alphabet."

**int** count

**for**(**int** i = 0; i < str.length(); i++)

{

**if**(str.charAt(i) == 'a' || str.charAt(i) == 'e' || str.charAt(i) == 'i' || str.charAt(i) == 'o' || str.charAt(i) == 'u')

{

count++;

}

}

println count

\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*

53

1. *Padded binary output*

Print the numbers from 0 to 15 in binary (use Java's Integer.toBinaryString() method). Use a method in String from the Groovy JDK to make all the output values have four digits.

**for** (i **in** 0..15)

println String.*format*("%4s", Integer.*toBinaryString*(i)).replace(" ", "0")

\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*

0000

0001

0010

0011

0100

0101

0110

0111

1000

1001

1010

1011

1100

1101

1110

1111

1. *Encode and decode*
   1. Create two strings, one for a username and one for a password. Concatenate them together, separated by a colon. Use a method from the Groovy JDK to convert the resulting String to a byte array. Then use the encodeBase64 method on byte array to create an encoded string.
   2. Decode the string by using the decodeBase64 method, and using the result as an argument to the String constructor. Use the split method to return the original username and password.

**package** oops

String username = "Shivani\_Pathak"

String password = "12345"

String str = username +":"+password

println username

println password

//Concatenated string

println "concatenated String:"+str

//Byte array of String

**byte**[] b = str.getBytes()

println b

//encode to base64

**def** encode = b.encodeBase64().toString()

println encode

//decode same string to base64

**byte**[] decode = encode.decodeBase64()

//println decode

//byte array

**def** c = **new** String(decode)

**def** original = res.split(':')

println original

\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*

Shivani\_Pathak

12345

concatenated String:Shivani\_Pathak:12345

[83, 104, 105, 118, 97, 110, 105, 95, 80, 97, 116, 104, 97, 107, 58, 49, 50, 51, 52, 53]

U2hpdmFuaV9QYXRoYWs6MTIzNDU=

[83, 104, 105, 118, 97, 110, 105, 95, 80, 97, 116, 104, 97, 107, 58, 49, 50, 51, 52, 53]

1. *Sorting a list*

Create a class called Course, with a String attribute called name and an int attribute called days. Create a list of four course instances, where at least two have the same number of days. Sort the list by number of days. Then, sort the list by days, but when the days are equal, sort by name.

**package** oops

**class** Course {

String name

**int** days

**def** getvalues(String n ,**int** d)

{

**def** name = n

**def** days = d

**def** l = [name:n,days:d]

}

**static** **void** main(args) {

Course First=**new** Course()

Course Second=**new** Course()

Course Third=**new** Course()

Course Fourth=**new** Course()

**def** common = [First.getvalues('Roger', 2), Second.getvalues('Smith', 2), Third.getvalues('Tillie', 6), Fourth.getvalues('Ben', 9)]

**def** sort = common.sort{a, b -> b["days"] <=> a["days"] ?: a["name"] <=> b["name"]}

sort.each { println it }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[name:Ben, days:9]

[name:Tillie, days:6]

[name:Roger, days:2]

[name:Smith, days:2]

1. *Operator overloading*
   1. Create a class called Money with a double amount and a String currency (like USD or EUR). Implement a plus method that checks that the currencies are the same and, if so, returns a new Money instance with the sum of the amounts and the correct currency. Write a similar minus method.
   2. Write a MoneyTest class in Groovy that uses + and - and verifies that they work properly.

**package** oops

**class** Money

{

**double** ammount = 50

String currency = 'EUR'

Money(String currency, **double** ammount)

{

**this**.ammount = ammount

**this**.currency = currency

}

**def** plus(Money m1)

{

**if** (**this**.currency==m1.currency)

{

**return** **new** Money(**this**.currency + ","+m1.currency,**this**.ammount+m1.ammount)

}

**else** {

println "Currency is not same"

}

}

**def** minus(Money m2)

{

**return** **new** Money(**this**.currency + ","+m2.currency,**this**.ammount-m2.ammount)

}

**public** **static** **void** main(args)

{

Money m3 = **new** Money("EUR",60)

Money m4 = **new** Money("EUR",70)

Money m5 = m3+m4

println m5.ammount

println m5.currency

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

130.0

EUR,EUR