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.

//

def num=2

Println num.getClass().getName()

2,20,200,.. are Integers

From 20000000000000000000 it is BIgInteger

//

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

Ans: 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=5/2

println x

println x.getClass().getName()

**def** num=5

println num.div(2)

println num.getClass().getName()

**def** z=0

println z.*divideUnsigned*(5, 2)

println z.getClass().getName()

//

Output:

2.5

java.math.BigDecimal

2.5

java.lang.Integer

2

java.lang.Integer

1. *Wrapper Classes*

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

//byte

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

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

println "<-------------------------->"

//short

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

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

println "<-------------------------->"

//int

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

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

println "<-------------------------->"

//long

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

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

println "<-------------------------->"

//float

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

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

println "<-------------------------->"

//double

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

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

1. *2s Complement*

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

//

**byte** x=127

println x+1

println x.getClass().getName()

//

Output:

128

java.lang.Byte

1. *Strings and GroovyStrings*

a. How many characters are in the string "Hello, Groovy!"?

//

**def** str="Hello,Groovy!"

println str.length()

//

Output: 13

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

//

String name="Vinay"

println "Hello "+name

println "Hello ${name}"

//

Output:

Hello Vinay

Hello Vinay

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

//

String str1="racecar"

println str1.reverse()

String str2="Bob"

str2=str2.toLowerCase()

println str2.reverse()

//

Output:

racecar

bob

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.

//

String sen="Hello, World. How are you?"

List li=[]

List si=[]

li=sen.split()

println li

println li.size()

si=sen.tokenize()

println si

println si.size()

//

Output:

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

5

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

5

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

//

String sen="Hello, World. How are you?"

println sen.substring(7, 12)

//

Output:

World

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

//

String sen="Hello, World. How are you?"

String cut=""

cut=sen.substring(22, 25)

println cut.reverse()

//

Output:

uoy

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.

//

**def** isPrime(num) {

**def** i

**def** flag=0

**for**(i=2; i<=num; i++) {

**if**(num%i==0) {

flag=1

**break**

}

}

**if**(flag==0 ) {

**return** **true**

}

**else** {

**return** **false**

}

}

//

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

Ans: num/2 is enough to check whether it is a prime number or not

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

*//*

List places=['Bhutan','Nepal','India','Brazil','Spain','Argentina','Conrad']

println places.sort()

List sort=places.sort{it.size()}

println sort

println sort.reverse()

//

Output:

[Argentina, Bhutan, Brazil, Conrad, India, Nepal, Spain]

[India, Nepal, Spain, Bhutan, Brazil, Conrad, Argentina]

[Argentina, Conrad, Brazil, Bhutan, Spain, Nepal, India]

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.

//

**def** nums=[123,56,45,78,354,10,27]

**def** sum=0

**def** sumd=0

println nums

**for**(i=0; i<7; i++) {

sum=sum+nums[i]

}

println "Adding all numbers in list: "+sum

**for**(i=0; i<7; i++) {

sumd=sumd+(nums[i]\*nums[i])

}

println "Doubling and adding: "+sumd

**def** avg2=sum/7

**def** avg3=sumd/7

println "Average of numbers: "+avg2

println "Average of double numbers: "+avg3

//

Output:

[123, 56, 45, 78, 354, 10, 27]

Adding all numbers in list: 693

Doubling and adding: 152519

Average of numbers: 99

Average of double numbers: 21788.4285714286

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.

//

**def** urlData = "https://www.linkedin.com/".toURL().getText()

println urlData.eachLine { line -> println line }

println urlData.eachLine { line -> println "Data on the page : " + line + "size of data: " + line.size() }

//

1. *Closures as a filter*

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

//

**def** myList=[2,0,-4,3,47,-8,3,5,-6,8,4,7,4,3]

println myList.findAll {

item -> item > 0

}

//

Output:

[2, 3, 47, 3, 5, 8, 4, 7, 4, 3]

1. *Multi-line strings*

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

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.

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.

//

String uname = "vinay"

String pass = "vinay1"

List end=[]

String str = "${uname} : ${pass}"

byte[] byt = str.getBytes()

def enc = byt.encodeBase64().toString()

byte[] dec = enc.decodeBase64()

def res = new String(dec)

end = res.split(':')

println end//

Output:

[vinay , vinay1]

* 1. 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.

//

**byte**[] dec = enc.decodeBase64()

**def** res = **new** String(dec)

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

println end

//

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.

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.