**Intoduction to Groovy**

**1. Create a class in Java along with following fields. classname: Person fields: name, age, gender, address Access the fields in all known ways: like through getter, by dot operator**

**Code**

public class Person

{

String name

int age

char gender

String address

public static void main(String[] args)

{

Person p=new Person()

p.name="ashima"

p.age=22

System.out.println(p.name);

System.out.println(p.age);

Person p1=new Person(name:"deepti",age:25)

System.out.println(p1.name);

System.out.println(p1.age);

}

}

**2. Print this pattern :**

**\***

**\*\***

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

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

**code**

int i=1

while(i<=8)

{

i.times{print("\*")}

println()

i=i\*2

}

**3. GString... override the toString() of the Person class to return something like... "Sachin is a man aged 24 who lives at Delhi. He works for Intelligrape with employee id 12 and draws $$$$$$$ lots of money !!!!."**

class Person {

String name

Integer age

String gender

String add

Integer empId

void getPerson()

{

println name

println gender println add

}

String toString()

{

return "${name} is a ${gender} aged ${this.age} who lives at ${add}. He works for Intelligrape with employee id ${empId} and draws \$\$\$\$ lots of money!!!! "

}

}

Person p=new Person(name:"rahul", age:22, gender:"male", add:"DElhi", empId:12)

println p

**4. Groovy Truth: if('test') { printlnn "test evaluated to true inside if" }**

**try replacing test with various objects and observe its behaviour.**

1. **"Test"**

**Output**

test evaluated to true inside if

**b)'null'**

**Output**

test evaluated to true inside if

**c)null**

**Output**

No output

1. **100**

**Output**

test evaluated to true inside if

1. **0**

**Output**

No output

1. **empty list**

**Output**

No output

**g) list with all vaues as null List list = new ArrayList()**

List l=new ArrayList()

l=[null,null,null]

if(l)

println "true"

else

println "false"

OUTPUT: true

**5. Print multiple of 3 upto 10 terms in at least three different ways using groovy special methods**

**Code**

a) 10.times{println 3\*(it+1)}

b)for(i=1;i<=10;i++){

println(3\*i)}

c) i=1

while(i<=10)

{

println 3\*i

i++

}

**6. Write a closure which checks if a value is odd/even, Use this closure with list of values to identify which values are odd and which are even.**

**Code**

List l=[10,11,12,13]

l.each{int a1->

if(a1%2==0)

println "${a1} is even"

else

println "${a1} is odd"

}

return

**Collections - I**

1. **Initialize an empty list and give the output of the following code:**

**l[11] = "myelement"**

**println l[11]**

**println l.get(5)**

**println l**

**Code**

List l=[]

l[11]="myelement"

println l[11]

println l.get(5)

println l

**Output**

myelement

null

[null, null, null, null, null, null, null, null, null, null, null,myelement]

1. **Initialize a list and find all elements which are divisible by 5.**

**Code**

List l=[14,12, 11,10, 16, 15,12, 10, 99, 90, 14, 16, 35]

l.each({if(it%5==0)

println it

})

return

1. **Given two lists [11, 12, 13, 14] and [13, 14, 15], how would we obtain the list of items from the first that are not in the second?**

**Code**

List l1=[11,12,13,14]

List l2=[13,14,15]

println(l1-l2)

1. **Find whether two lists have a common element or not.**

**Code**

List l1=[11,12,13,14]

List l2=[13,14,15]

println(l1.intersect(l2))

1. **Remove all records from a list whose index is odd.**

List l=[12,45,2,89,10,48,54,55,90]

size1=l.size()

index=1

while(index<size1)

{

l.remove(index)

index++

size1--

}

println "List after removing all elements whose index are odd:" + l

**6. Consider the following list:**

**[1, 2, 3, "element1", 0.3, [2, 4, 6], 0..10 ]**

**Print the class name of each element. What's the output of the following statement?**

**list.get(6).get(9)**

**OUTPUT:**

class java.util.ArrayList

9

**7. Sort the given list in descending order having distinct elements:**

**[14,12, 11,10, 16, 15,12, 10, 99, 90, 14, 16, 35]**

List l=[14,12, 11,10, 16, 15,12, 10, 99, 90, 14, 16, 35]

l1= l.unique()

l2=l1.sort()

println l2.reverse()

**8. Consider a class Employee with following details**

**\* Name**

**\* Age**

**\* Salary**

**Create a list consisting of 10 Employee objects.**

**(a). Get a list of employees who earn less than 5000**

**Code**

class Employee

{

String name

int age

int salary

public static void main(String[] args)

{

Employee e1=new Employee(name:"ram",age:18,salary:20000)

Employee e2=new Employee(name:"shyam",age:20,salary:2000)

Employee e3=new Employee(name:"ankur",age:28,salary:25000)

Employee e4=new Employee(name:"shyama",age:27,salary:2900)

Employee e5=new Employee(name:"sharmila",age:22,salary:24000)

Employee e6=new Employee(name:"kishor",age:20,salary:2100)

Employee e7=new Employee(name:"vinay",age:21,salary:23000)

Employee e8=new Employee(name:"neha",age:23,salary:5000)

Employee e9=new Employee(name:"garima",age:26,salary:6000)

Employee e10=new Employee(name:"sneha",age:15,salary:4000)

List l=[e1,e2,e3,e4,e5,e6,e7,e8,e9,e10]

List l1=l.findAll{

Employee e->e.salary<5000

}

println(l1\*.name)

}

}

OUTPUT: [shyam, shyama, kishor, sneha]

**(b). Get the name of the youngest employee and oldest employee**

**Code**

class Employee

{

String name

int age

int salary

public static void main(String[] args)

{

Employee e1=new Employee(name:"ram",age:18,salary:20000)

Employee e2=new Employee(name:"shyam",age:20,salary:2000)

Employee e3=new Employee(name:"ankur",age:28,salary:25000)

Employee e4=new Employee(name:"shyama",age:27,salary:2900)

Employee e5=new Employee(name:"sharmila",age:22,salary:24000)

Employee e6=new Employee(name:"kishor",age:20,salary:2100)

Employee e7=new Employee(name:"vinay",age:21,salary:23000)

Employee e8=new Employee(name:"neha",age:23,salary:5000)

Employee e9=new Employee(name:"garima",age:26,salary:6000)

Employee e10=new Employee(name:"sneha",age:15,salary:4000)

List l=[e1,e2,e3,e4,e5,e6,e7,e8,e9,e10]

print("Name of the youngest employee is:")

println(l.min({it.age}).name)

print("Name of the oldest employee is:")

println(l.max({it.age}).name)

}

}

OUTPUT:

Name of the youngest employee is:sneha

Name of the oldest employee is:ankur

**(c). Get the employee with maximum salary**

**Code**

class Employee

{

String name

int age

int salary

public static void main(String[] args)

{

Employee e1=new Employee(name:"ram",age:18,salary:20000)

Employee e2=new Employee(name:"shyam",age:20,salary:2000)

Employee e3=new Employee(name:"ankur",age:28,salary:25000)

Employee e4=new Employee(name:"shyama",age:27,salary:2900)

Employee e5=new Employee(name:"sharmila",age:22,salary:24000)

Employee e6=new Employee(name:"kishor",age:20,salary:2100)

Employee e7=new Employee(name:"vinay",age:21,salary:23000)

Employee e8=new Employee(name:"neha",age:23,salary:5000)

Employee e9=new Employee(name:"garima",age:26,salary:6000)

Employee e10=new Employee(name:"sneha",age:15,salary:4000)

List l=[e1,e2,e3,e4,e5,e6,e7,e8,e9,e10]

print("Name of the employee having maximum salary is:")

println(l.max({it.salary}).name)

}

}

OUTPUT

Name of the employee having maximum salary is:ankur

**(d). Get the list of names of all the employees**

**Code**

class Employee

{

String name

int age

int salary

public static void main(String[] args)

{

Employee e1=new Employee(name:"ram",age:18,salary:20000)

Employee e2=new Employee(name:"shyam",age:20,salary:2000)

Employee e3=new Employee(name:"ankur",age:28,salary:25000)

Employee e4=new Employee(name:"shyama",age:27,salary:2900)

Employee e5=new Employee(name:"sharmila",age:22,salary:24000)

Employee e6=new Employee(name:"kishor",age:20,salary:2100)

Employee e7=new Employee(name:"vinay",age:21,salary:23000)

Employee e8=new Employee(name:"neha",age:23,salary:5000)

Employee e9=new Employee(name:"garima",age:26,salary:6000)

Employee e10=new Employee(name:"sneha",age:15,salary:4000)

List l=[e1,e2,e3,e4,e5,e6,e7,e8,e9,e10]

println(l\*.name)

}

}

OUTPUT:

[ram, shyam, ankur, shyama, sharmila, kishor, vinay, neha, garima, sneha]

**9. Consider the following piece of code:**

**String s = "this string needs to be split"**

**println s.tokenize(" ")**

**println s.tokenize()**

**Output**

[this, string, needs, to, be, split]

[this, string, needs, to, be, split]

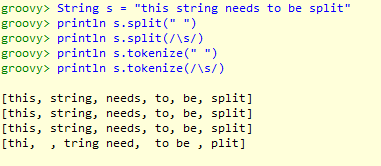
**Compare this with the following code:**

**String s = "this string needs to be split"**

**println s.split(" ")**

**println s.split(/\s/) (Try Same Parameter with tokenize)**

**Output**

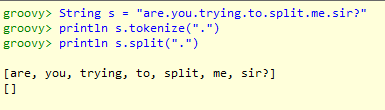


**Also try the following exercise:**

**String s = "**[**are.you.trying.to.split.me**](http://are.you.trying.to.split.me/)**.sir?"**

**s.tokenize(".")**

**s.split(".")**

****

**10. Get first and last element of List.**

**Code**

List l=[12,45,2,89,10,48,54]

println(l.head())

println(l.last())

**11. Print the table of a given number : 2 and 12**

**Code**

11.times{println "2 \* $it =" + 2\*(it)}

11.times{println "12 \* $it =" + 12\*(it)}

**12. We have a sorted list of alphabets a-z, print all alphabets appearing after j**

List l=('a'..'z')

println ('k'..'z')

**14. Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print "FizzBuzz".**

List l=(1..100)

println "Numbers from 1 to 100:\n"

l.each{int a1->

if((a1%5==0) && (a1%3==0))

println "FizzBuzz"

else if(a1 % 5 ==0)

println "Buzz"

else if(a1%3==0)

println "Fizz"

else

println a1

}

return