**Quiz 2**

**kiranraj jonnalagadda**

1. Ruby program when that accepts a date in a month and prints it out the appropriate suffix. For example, for 1 as input, print 1st, 2 as input 2nd, 3 as input 3rd etc

answer:

#!/usr/bin/ruby

require 'ordinalize'

puts **"please enter a month number to print suffix of the month"**

num=gets.to\_i

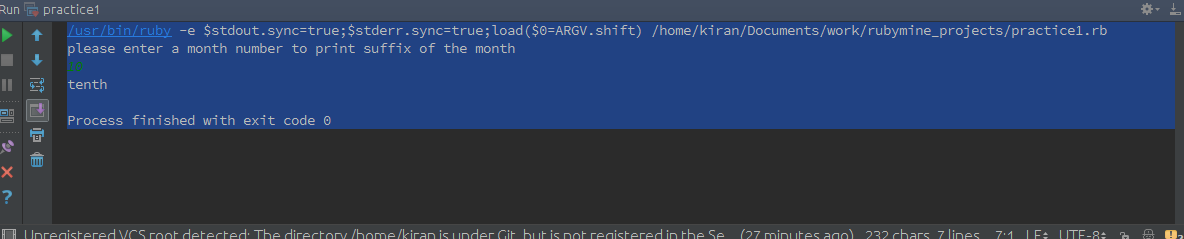
**if** ((num <31) && (num>0))

puts num.ordinalize

**else** puts **"please enter correct month number"**

**end**

output:



2. Write your own ruby program using a case statement

answer:

2)write your own program using a ruby program using a case statement

answer:

#!/usr/bin/ruby

puts **"guess a birds or animals names present in our database"**

bird =gets.chomp

**case** bird

**when "peacock"**

puts **"**#{bird} **is in our database , congrats you won"**

**when "crow"**

puts **"**#{bird} **is in our database , congrats you won"**

**when "goose"**

puts **"**#{bird} **is in our database , congrats you won"**

**when "tiger"**

puts **"**#{bird} **is in our database , congrats you won"**

**when "leopard"**

puts **"**#{bird} **is in our database , congrats you won"**

**when "dog"**

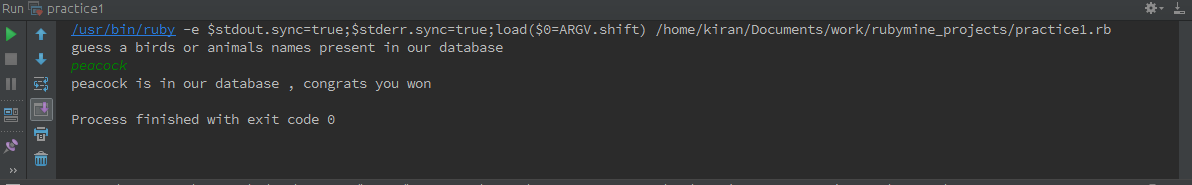
puts **"**#{bird} **is in our database , congrats you won"**

**else**

**puts “sorry, you lost the game”**

**end**

output:



3. Ruby Program that iterates numbers with upto loop

answer:

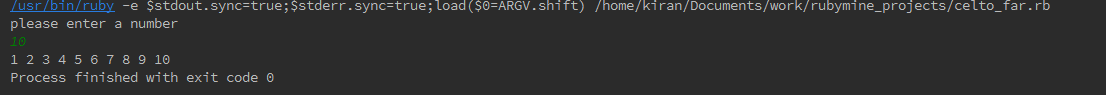
#!/usr/bin/ruby

puts **"please enter a number"**

num=gets.to\_i

1.upto (num){|*i*| print *i* ,**" "**}

output:



4. Write a Ruby program to print numbers from 1 to 50 and also in reverse order

answer:

#!/usr/bin/ruby

**for** *i* **in** 1..50

print *i*,**" "**

**if** *i*==50

puts **""**

j=*i*

**while** j>0

print j,**" "**

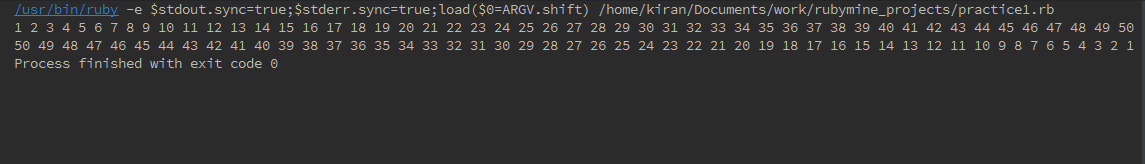
j = j-1

**end**

**end**

**end**

output:



5. Write your own Ruby program using loops and iterators. Explain the difference between loops, iterators and blocks

answer:

*#loops*

*#loops will iterates according to the conditions*

**for** *i* **in** 1..10

puts *i*

**end**

*# blocks are used to work for the extended functionality for methods*

**def** blockeg

puts **"in method"**

**yield** 10

**end**

blockeg{

|*i*| print *i*,**" from block"**

}

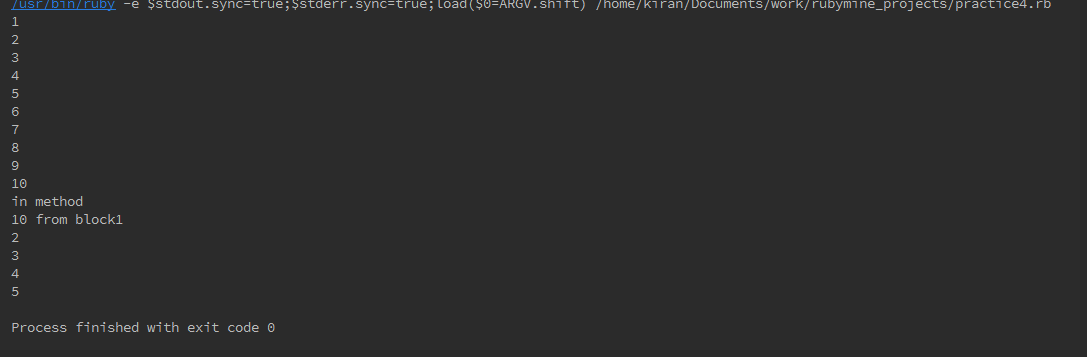
*#***iterates**

*## iterates means like each* ***until*** *the* ***end*** *and mainly used for iterating arrays hashes etc..*

arr=[1,2,3,4,5]

arr.each{|*i*| puts *i*}

output:



6. Write a Ruby program that loops through a array and checks if a pattern existsin the array elements

answer:

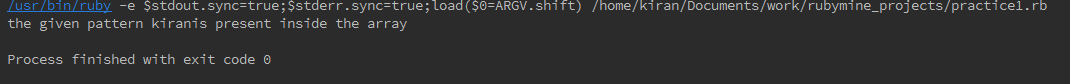
*#arrays*

arr=[**"kiran"**,**"anil"**,**"arun"**,**"pavan"**,**"abhinav"**,**"asish"**,**"bindhu"**]

arr.each{|*a*| puts **"the given pattern** #{*a*}**is present inside the array" if** *a*==**"kiran"**}*#it checks each pattern inside the array*

arr.each {|*b*| puts **"the given pattern** #{*b*} **is present inside the array" if** *b*==**"kamal"** }

output:



7. Write your own Ruby program using a Hash that loops through :

Print all Values while looping with Keys

Print all Keys while looping through Values

Print Keys, Values as pair.

Answer:

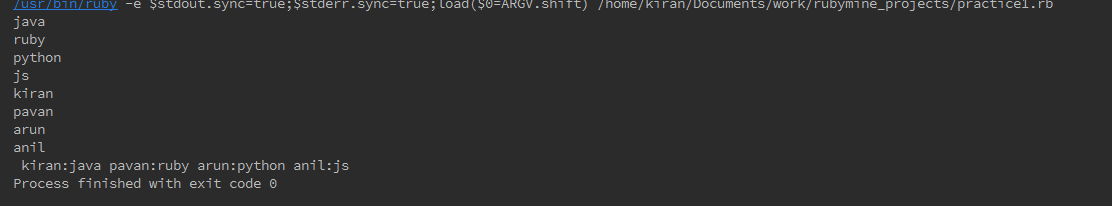
samplehash={**"kiran"***=>***"java"**,**"pavan"***=>***"ruby"**,**"arun"***=>***"python"**,**"anil"***=>***"js"**}

puts samplehash.values

puts samplehash.keys

samplehash.each { |*a*,*b*| print **" "**,*a*, **":"**, *b* }

output:



8. Write a Ruby program that takes number as input and recursively calculates the power of 2 until the calculated number is less than 10000 and prints the maximum power for that number.

Answer:

**class** *Calculation*

attr\_accessor :num

**def** expo(*num*)

@num=*num*

**if** @num<10000

puts *num*

@num =@num \*\* 2

**self**.expo(@num)

**end**

**end**

**end**

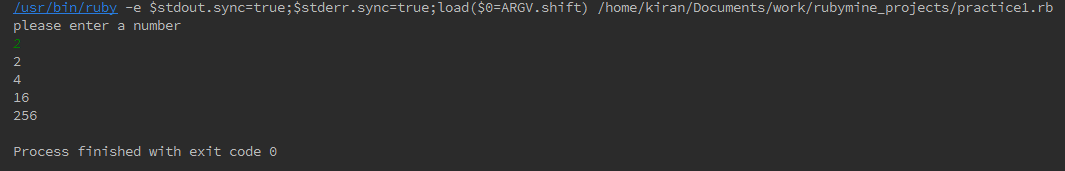
puts **"please enter a number"**

num=gets.to\_i

c=Calculation.new

c.expo(num)

output:



d

9. Ruby program to convert Celsius temperature to Fahrenheit

answer:

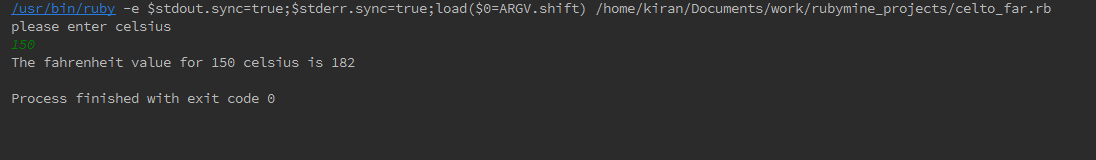
puts **"please enter celsius "**

celsius=gets.to\_i

fahrenheit = (9/5) \* celsius + 32

puts **"The fahrenheit value for** #{celsius} **celsius is** #{fahrenheit} **"**

output:



10. Write a program to create a Calculator class with add(), substract(), multiply() and divide(), then take two numbers and choice of operation from user and display output using objects

answer:

**class** *Calculator*

attr\_accessor :num1,:num2

**def** initialize(*num1*,*num2*)

@num1=*num1*

@num2=*num2*

**end**

**def** add()

puts @num1 + @num2

**end**

**def** sub()

puts @num1 - @num2

**end**

**def** mul()

puts @num1 \* @num2

**end**

**def** div()

puts @num1 % @num2

**end**

**end**

puts **"please enter a number for calculation"**

@num1=gets.to\_i

puts **"please enter a number for calculation"**

@num2=gets.to\_i

c=Calculator.new(@num1,@num2)

puts **"what do you want to do "**

puts **"+ or - or x or %"**

calc=gets.chomp

**case** calc

**when "+"**

c.add

**when "-"**

c.sub

**when "x"**

c.mul

**when "%"**

c.div

**else**

puts **"please enter a correct operator for calculation"**

**end**

output:

