RUBY LAB ASSESSMENT 3

21MIS1021 VIMAL KUMAR S

1. Write a ruby code using the following keywords yield, lambda and procs.

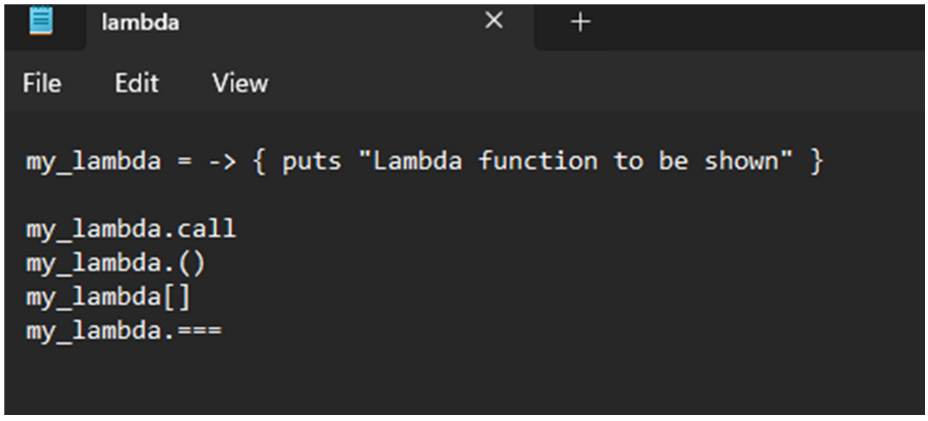
YIELD

```
def one_two_three
  yield 1
  yield 2
  yield 3
end

one_two_three { |number| puts number * 10 }
# 10, 20, 30
```

```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby yield.rb
10
20
30
```

LAMBDA




```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby lambda.rb
Lambda function to be shown
Lambda function to be shown
Lambda function to be shown
Lambda function to be shown
```

PROCS

procs

File Edit View

```
def call_proc
  puts "Before proc"
  my_proc = Proc.new { return 2 }
  my_proc.call
  puts "After proc"
end

p call_proc
# Prints "Before procs" but not "After procs"
```

```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby procs.rb
Before proc
2
```

2. Write a ruby programming using Modules concept.

module

File Edit View

```
module Gfg
C  = 10;
def Gfg.portal
puts "Welcome to Vimal Class"
end
def Gfg.tutorial
puts "Ruby Vimal"
end
def Gfg.topic
puts "Vimal Kumar S"
end
 end

puts Gfg: :C
 Gfg.portal
Gfg.tutorial
Gfg.topic
```

3. Write a ruby programming using Mixins concept.

mixins

File Edit View

```
module Child_1
def a1
puts 'This is Vimal.'
end
end
module Child_2
def a2
puts 'This is Kumar.'
end
 end
module Child_3
def a3
puts 'This is Priyanka.'
end
end

class Parent
include Child_1
include Child_2
include Child_3
def display
puts '3 modules included.'
end
end

object = Parent.new

object.display
object.a1
object.a2
object.a3
```

```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby mixins.rb
3 modules included.
This is Vimal.
This is Kumar.
This is Priyanka.
```

4. Write a ruby programming using Reflection concept.

reflection

File Edit View

```
class Person
attr_accessor :name, :age
def initialize(name,age)
@name=name
@age=age
end

def introduce
puts "Hii My name is Vimal Kumar."
end
end

person=Person.new("Vimal",19)
puts "Class name: #{person.class}"
puts "Class methods: #{person.methods(false)}"
puts "Class variables: #{person.instance_variables}"

method_name="introduce"
if person.respond_to?(method_name)
person.send(method_name)
person.send(method_name)
end

instance_variable_name="@name"
if person.instance_variable_defined?(instance_variable_name)
value=person.instance_variable_get(instance_variable_name)
puts "Value of #(instance_variable_name): #(value)"
end
```

```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby reflection.rb
Class name: Person
Class methods: []
Class variables: [:@name, :@age]
Hii My name is Vimal Kumar.
Hii My name is Vimal Kumar.
Value of #(instance_variable_name): #(value)
```

5. Write a ruby programming using Meta-programming concept

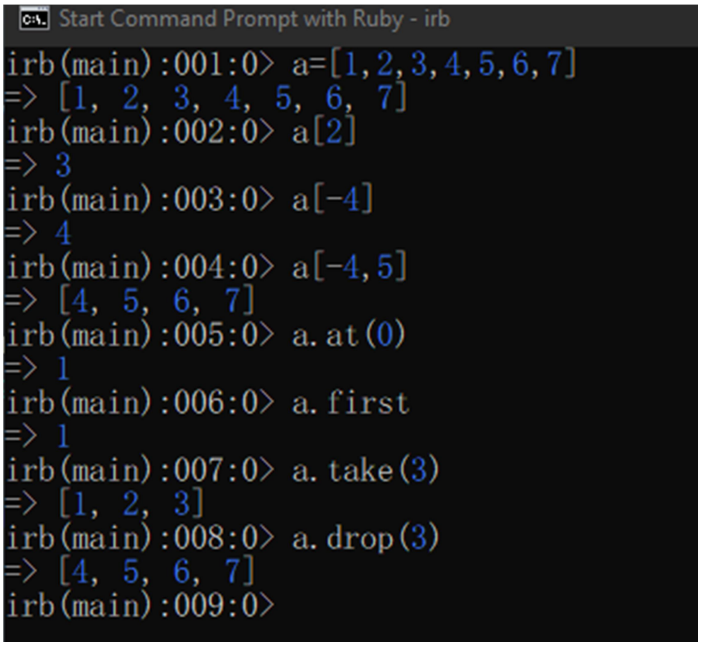
```
File    Edit    View

class MyClass
def self.create_method(name)
self.class_eval do
define_method(name) do |arg|
puts "Hello I am Vimal Priy..! , #(arg)!"
end
end
end
end

obj=MyClass.new
MyClass.create_method(:greet)
obj.greet("World")
```

```
C:\Users\student.PROGRAMMING404\Desktop\21MIS1021 Vimal>ruby meta.rb
Hello I am Vimal Priy..! , #(arg)!
```

6. Create an array a=[1,2,3,4,5,6], and perform the following:
   a. Different ways to access the array elements



   b. Five different methods associated with array.

```
irb(main):012:0> a=[1,2,3,4,5,6]
=> [1, 2, 3, 4, 5, 6]
irb(main):013:0> a.last
=> 6
irb(main):014:0> a.length
=> 6
irb(main):015:0> a.first
=> 1
irb(main):016:0> a.reverse
=> [6, 5, 4, 3, 2, 1]
irb(main):017:0> a.join
=> "123456"
irb(main):018:0>
```

c. Different ways to add and delete an element of an array.

```
Start Command Prompt with Ruby - irb
irb(main):017:0> a.join
=> "123456"
irb(main):018:0> a=[1,2,3,4,5,6]
=> [1, 2, 3, 4, 5, 6]
irb(main):019:0> a.push(5.5)
=> [1, 2, 3, 4, 5, 6, 5.5]
irb(main):020:0> a.<<7
=> [1, 2, 3, 4, 5, 6, 5.5, 7]
irb(main):021:0> a.unshift(0)
=> [0, 1, 2, 3, 4, 5, 6, 5.5, 7]
irb(main):022:0> a.insert(4, 'Vimal')
=> [0, 1, 2, 3, "Vimal", 4, 5, 6, 5.5, 7]
irb(main):023:0> a.pop
=> 7
irb(main):024:0> a.shift
=> 0
irb(main):025:0>
```

d. Introduce two new arrays and perform intersection, concatenation, difference.

```
irb(main):025:0> a=[1,2,3,4,5]& b=[1,2,3]
=> [1, 2, 3]
irb(main):026:0> a=[4,5,6]+ b=[1,2,3]
=> [4, 5, 6, 1, 2, 3]
irb(main):027:0> a=[4,5,6]+ b=[3,4,6]
=> [4, 5, 6, 3, 4, 6]
irb(main):028:0>
```

e. Perform a binary search using array a.