

**SWE2034 – Ruby Programming**

Guided By – **Dr Yogesh C**

Slot – **L5+L6**

NAME: **VISHWANTH P**

REGISTER.NO: **21MIS1117**

**Lab Assessment - 5**

1. **Write a separate program using the following functions:**

**Fiber – yield and resume**

**Fiber – transfer**

**Fiber – raise**

Code

fiber1 = Fiber.new do

  puts "Fiber1 is running"

  Fiber.yield

  puts "Fiber1 is resumed"

end

fiber1.resume

puts "back to Main"

fiber1.resume

fiber2 = Fiber.new do

  puts "Fiber2 is running"

end

fiber3 = Fiber.new do

  puts "Fiber3 is running"

  fiber2.transfer

end

fiber3.resume

fiber4 = Fiber.new do

  begin

    Fiber.yield

  rescue => e

    puts "Caught an exception: #{e.message}"

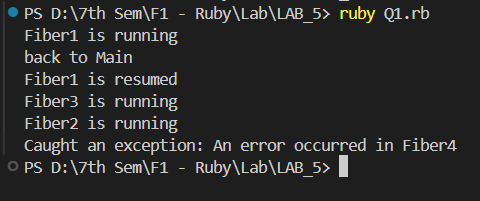
  end

end

fiber4.resume

fiber4.raise "An error occurred in Fiber4"

Output:



1. **Create 10 threads, each of which sleep for a random amount of time and then prints a message.**

Code

require 'thread'

threads = []

10.times do |i|

    threads << Thread.new do

        sleep\_time = rand(1..5)

        sleep(sleep\_time)

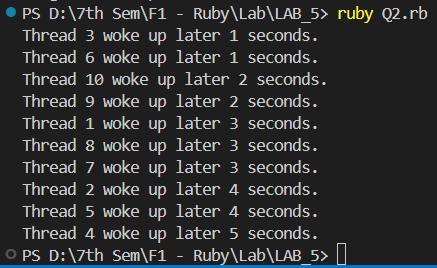
        puts "Thread #{i+1} woke up later #{sleep\_time} seconds."

    end

end

threads.each(&:join)

Output

****

1. **Create a local variable for a main thread, additional threads and fiber and prints the value of it.**

Code

Main\_var = "Main thread variable"

Fiber\_var = "Fiber variable"

Thread\_var = "Thread variable"

fiber = Fiber.new do

    puts " inside fiber : #{Fiber\_var}"

end

thread = Thread.new do

    puts " inside thread :  #{Thread\_var}"

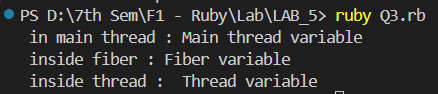
end

puts " in main thread : #{Main\_var}"

fiber.resume

thread.join

Output



1. **Local variable values in Nested Thread within a Fiber.**

Code

fiber = Fiber.new do

    local\_fiber\_var = "Fiber variable"

    Thread.new do

        thread\_var = "Thread variable"

        puts thread\_var

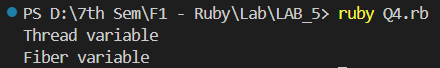
        puts local\_fiber\_var

    end.join

end

fiber.resume

Output



1. **Local variable values in Nested Fiber within a Thread.**

Code

Thread.new do

    thread\_var = "Thread variable"

    fiber=Fiber.new do

        local\_fiber\_var = "fiber variable"

        puts local\_fiber\_var

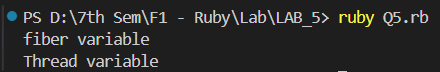
        puts thread\_var

    end

    fiber.resume

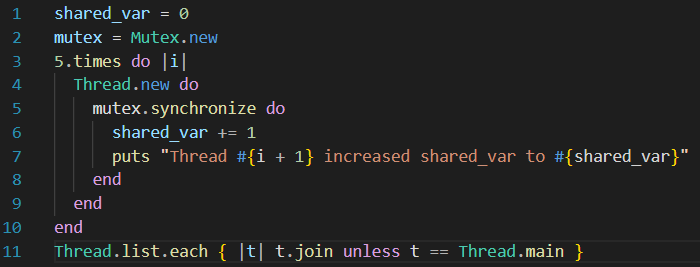
end.join

Output

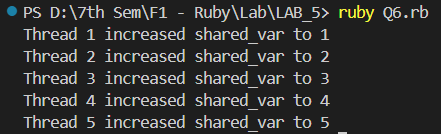


1. **Multi Thread sharing same variable address space.**

Code

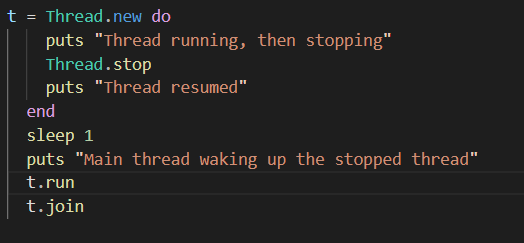


Output

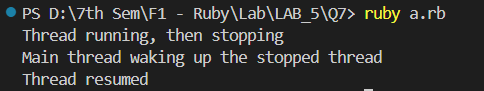


1. **Write a separate program using the following functions:** 
   1. **Thread – Stop and Run**

**Code**

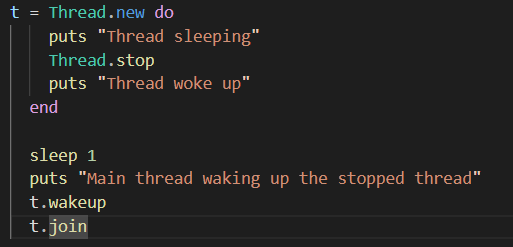
****

**Output**

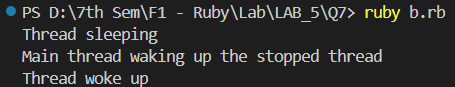
****

* 1. **Thread – Wakeup**

**Code**

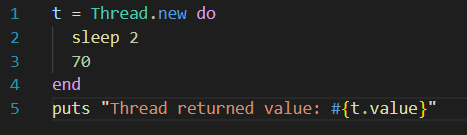
****

**Output**

****

* 1. **Thread – Value**

**Code**

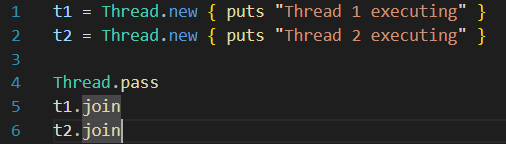
****

**Output**

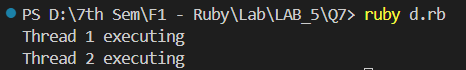
****

* 1. **Thread – Pass**

**Code**

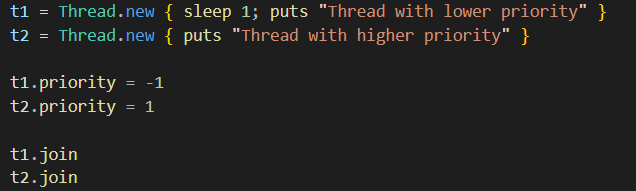
****

**Output**

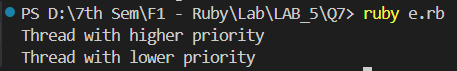
****

* 1. **Thread –Priority**

**Code**

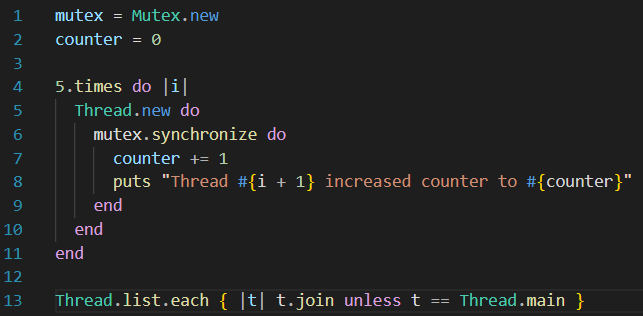
****

**Output**

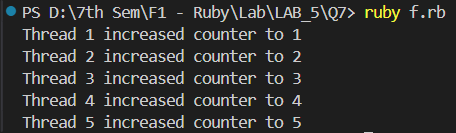
****

* 1. **Thread – Mutex**

**Code**

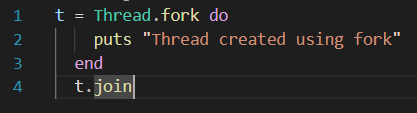
****

**Output**

****

* 1. **Thread – Fork**

**Code**

****

**Output**

****