# Ruby Lab Assessment – 3

Suryakumar P 21MIS1146

#Library Catalog:  
#Code:

library\_catalog = {}

def add\_book\_to\_catalog(catalog)

  puts "Enter the book's title:"

  title = gets.chomp

  puts "Enter the author's name:"

  author = gets.chomp

  puts "Enter the genre:"

  genre = gets.chomp

  puts "Enter the publication year:"

  year = gets.chomp.to\_i

  id = catalog.size + 1

  catalog[id] = { title: title, author: author, genre: genre, year: year }

end

def books\_published\_after(catalog, year)

  catalog.select { |id, book| book[:year] > year }

end

loop do

  add\_book\_to\_catalog(library\_catalog)

  puts "Do you want to add another book? (yes/no)"

  answer = gets.chomp.downcase

  break if answer != 'yes'

end

puts "Enter the year to search for books published after:"

search\_year = gets.chomp.to\_i

searched\_books = books\_published\_after(library\_catalog, search\_year)

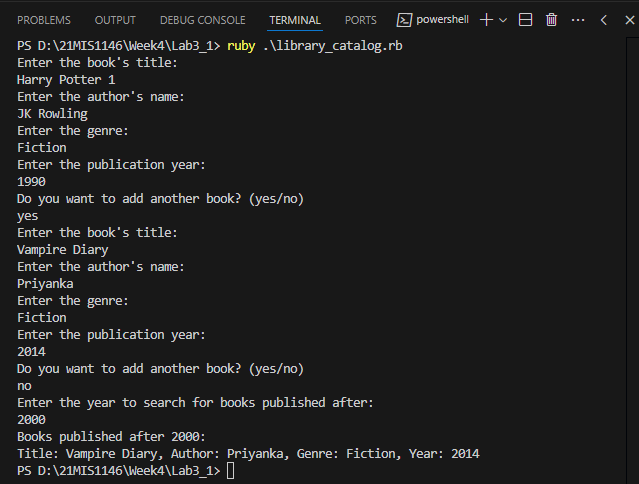
puts "Books published after #{search\_year}:"

searched\_books.each do |id, book|

  puts "Title: #{book[:title]}, Author: #{book[:author]}, Genre: #{book[:genre]}, Year: #{book[:year]}"

end

Output:



Employee Database Management:

Code:

employee\_database = {}

def add\_employee\_to\_database(database)

  puts "Enter the employee's name:"

  name = gets.chomp

  puts "Enter the employee's department:"

  department = gets.chomp

  puts "Enter the employee's salary:"

  salary = gets.chomp.to\_f

  id = database.size + 1

  database[id] = { name: name, department: department, salary: salary }

end

def highest\_paid\_employee(database)

  database.max\_by { |id, employee| employee[:salary] }

end

loop do

  add\_employee\_to\_database(employee\_database)

  puts "Do you want to add another employee? (yes/no)"

  answer = gets.chomp.downcase

  break if answer != 'yes'

end

highest\_paid = highest\_paid\_employee(employee\_database)

if highest\_paid

  id, details = highest\_paid

  puts "The highest-paid employee is:"

  puts "Name: #{details[:name]}"

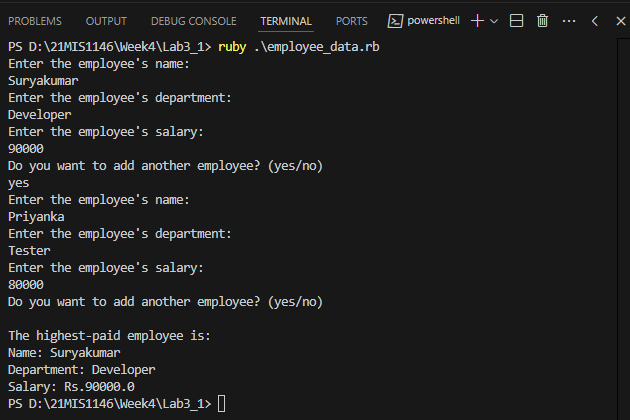
  puts "Department: #{details[:department]}"

  puts "Salary: Rs.#{details[:salary]}"

else

  puts "No employees found in the database."

end

Output:  


Market Place

Code:

marketplace = {}

def add\_product\_to\_marketplace(marketplace)

  puts "Enter the product name:"

  name = gets.chomp

  puts "Enter the product price:"

  price = gets.chomp.to\_f

  puts "Enter the product quantity:"

  quantity = gets.chomp.to\_i

  id = marketplace.size + 1

  marketplace[id] = { name: name, price: price, quantity: quantity }

end

def total\_value\_of\_products(marketplace)

  marketplace.sum { |id, product| product[:price] \* product[:quantity] }

end

loop do

  add\_product\_to\_marketplace(marketplace)

  puts "Do you want to add another product? (yes/no)"

  answer = gets.chomp.downcase

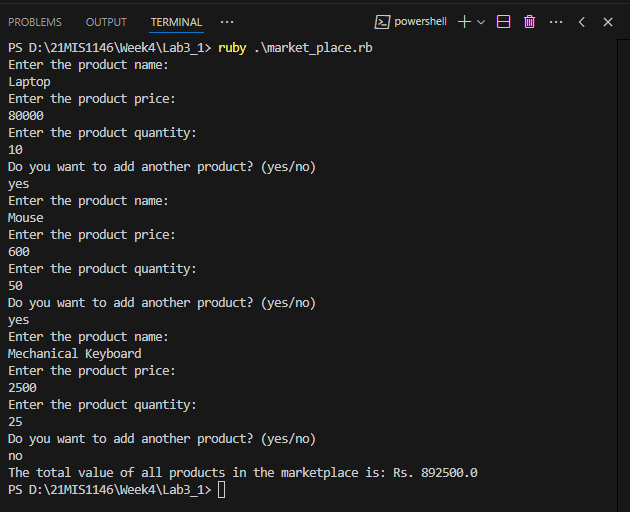
  break if answer != 'yes'

end

total\_value = total\_value\_of\_products(marketplace)

puts "The total value of all products in the marketplace is: Rs. #{total\_value}"

Output:



Student Grade:

Code:  
def grade\_to\_points(grade)

  case grade

  when 'S' then 10

  when 'A' then 9

  when 'B' then 8

  when 'C' then 7

  when 'D' then 6

  when 'F' then 0

  else 0

  end

end

def calculate\_cgpa(students\_grades)

  students\_grades.each do |student, grades|

    total\_points = grades.map { |grade| grade\_to\_points(grade) }.sum

    cgpa = total\_points.to\_f / grades.size

    puts "#{student}'s CGPA: #{cgpa.round(2)}"

  end

end

def get\_student\_grades

  students\_grades = {}

  puts "Enter the number of students:"

  number\_of\_students = gets.chomp.to\_i

  number\_of\_students.times do

    puts "Enter the student's name:"

    student\_name = gets.chomp

    puts "Enter the grades for #{student\_name} (separated by spaces):"

    grades = gets.chomp.split

    students\_grades[student\_name] = grades

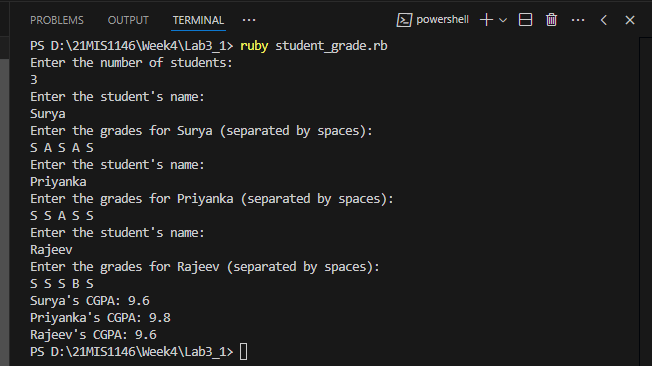
  end

  students\_grades

end

students\_grades = get\_student\_grades

calculate\_cgpa(students\_grades)

Output:  


Music Player:

Code:

playlist = {}

def add\_song\_to\_playlist(playlist)

  puts "Enter the song title:"

  title = gets.chomp

  puts "Enter the artist name:"

  artist = gets.chomp

  puts "Enter the genre:"

  genre = gets.chomp

  id = playlist.size + 1

  playlist[id] = { title: title, artist: artist, genre: genre }

end

def shuffle\_and\_play(playlist)

  shuffled\_playlist = playlist.keys.shuffle

  puts "Playing songs in random order:"

  shuffled\_playlist.each do |id|

    song = playlist[id]

    puts "Now playing: '#{song[:title]}' by #{song[:artist]} [#{song[:genre]}]"

  end

end

loop do

  add\_song\_to\_playlist(playlist)

  puts "Do you want to add another song? (yes/no)"

  answer = gets.chomp.downcase

  break if answer != 'yes'

end

shuffle\_and\_play(playlist)

Output:

