Test 1 - Incident

**You must submit your code (use the Mark button) within 2 hours of the starting time for the test.**

Use the scaffold code provided for this task. Complete the following code requirements in the Ruby language for the file incident.rb. Your final code must be structured and run.

Requirements:

* You must create a record (using Ruby classes) for an incident. An Incident has the following attributes: an id (Integer), a time of day (24 hour time as a string eg: 14:30), a place (String), and a severity rating (String format: ‘Severe’, ‘Moderate’ or ‘Mild)
* Write a function called read\_an\_incident() that reads from the terminal values for each of the fields in a Incident record and returns the completed record.
* Write a procedure called print\_an\_incident(incident) that takes an incident record and writes each of the fields to the terminal with a description for the field as well as the field value.
* Write a function called read\_incidents() that calls your read\_an\_incident() and returns an array of Incidents.
* Write a procedure called print\_incidents(incidents) that calls your print\_an\_incident(incident) procedure for each incident in the array.

Use the following test data:

Table

Description automatically generated

Your code should run as follows (**with the test data above**):

Table

Description automatically generated with low confidence

SUBMIT YOUR CODE WITHIN 2 HOURS OF THE START OF THE TEST (USE THE MARK BUTTON)

require './input\_functions'

# Complete the code below

# Use input\_functions to read the data from the user

# you will need to define an Incident here:

class Incident

attr\_accessor :id, :place, :time, :severity

def initialize (id, place, time, severity)

# insert lines here

@id = id

@place = place

@time = time

@severity = severity

end

end

def read\_an\_incident()

id = read\_integer("Enter incident id: ")

place = read\_string("Enter place: ")

time = read\_string("Enter time: ")

severity = read\_string("Enter severity: ")

incident = Incident.new(id, place, time, severity)

return incident

end

def read\_incidents()

count = read\_integer("How many incidents are you entering: ")

incidents = Array.new()

i = 0

while i < count

incident = read\_an\_incident()

incidents << incident

i +=1

end

return incidents

end

def print\_an\_incident(incident)

puts("Id " + incident.id.to\_s)

puts("Place " +incident.place)

puts("Time " +incident.time)

puts("Severity " +incident.severity)

end

def print\_incidents(incidents)

i = 0

while i < incidents.length()

print\_an\_incident(incidents[i])

i+=1

end

end

def main()

incidents = read\_incidents()

print\_incidents(incidents)

end

main()