Swinburne University of Technology

COS20007 Object Oriented Programming

1.1P: Preparing for Object Oriented Programming

PDF generated at 15:00 on Sunday 5th March, 2023

1.1P: Preparing for OOP – Answer Sheet

1. Explain the following terminal instructions:
    a. cd: the command ‘cd’ is used to open a specified directory.
    b. ls: the command ‘ls’ in a bash terminal is used to list all items under the current target directory.
    c. pwd: the command ‘pwd’ in a bash terminal is used to print the current target directory to the terminal.

2. Consider the following kinds of information, and suggest the most appropriate data type to store or represent each:

Information	**Suggested Data Type**
A person’s name	String
A person’s age in years	Integer
A phone number	Depends on context and how the info will be used, integer form is commonly used yet if separations between groups of digits is required then a string format may fit.
A temperature in Celsius	Integer/float
The average age of a group of people	Float
Whether a person has eaten lunch	Boolean

3. Aside from the examples already provided in question 2, come up with an example of information that could be stored as:

Data type	**Suggested Information**
String	Name of a place
Integer	Number of pets
Float	Pi to 7 digits
Boolean	Whether it will rain today

4. Fill out the last two columns of the following table, evaluating the value of each expression and identifying the data type the value is most likely to be:

Expression	Given	Value	Data Type
`6`		6	Integer
`True`		True	Boolean
`a`	`a = 2.5`	2.5	Float
`1 + 2 * 3`		7	Integer
`a and False`	`a = True`	False	Boolean
`a or False`	`a = True`	True	Boolean
`a + b`	`a = 1` `b = 2`	3	Integer
`2 * a`	`a = 3`	6	Integer
`a * 2 + b`	`a = 2.5` `b = 2`	7	Integer
`a + 2 * b`	`a = 2.5` `b = 2`	6.5	Float

`(a + b) * c`	`a = 1`
	`b = 1`
	`c = 5`
		10	Integer

`"Fred" + " Smith"`		"Fred Smith"	String
`a + " Smith"`	`a = "Wilma"`	"Wilma Smith"	String

5. Using an example, explain the difference between **declaring** and **initialising** a variable.

The difference between the two is that declaring a variable is to reserve memory and give a name to a new variable which does not have to contain content, initialising a variable is to add a value to that variable.

6. Explain the term **parameter**. Write some code that demonstrates a simple of use of a parameter. You should show a procedure or function that uses a parameter, and how you would call that procedure or function.

A parameter is in most cases a variable that is passed into a function to manipulate the output of said function.

```
a = 2

def main(a)

    print(a*a*a)

end

main(a)
```

7. Using an example, describe the term **scope** as it is used in procedural programming (not in business or project management). Make sure you explain the different kinds of scope.

Scope is a way to define whether a specific function, variable, class, etc can be used within the current target field. For example a function of a parent class cannot be used within an instance of the child class. Two common names for the variation in scope are local and global.

8. In a procedural style, in any language you like, write a function called Average, which accepts an array of integers and returns the average of those integers. Do not use any libraries for calculating the average. You must demonstrate appropriate use of parameters, returning and assigning values, and use of a loop. Note — just write the function at this point, we'll *use* it in the next task. You shouldn't have a complete program or even code that outputs anything yet at the end of this question.

```ruby
def main(array)

    sum = 0

    array.each { |value|

        sum += value

    }

    average = sum.to_f/array.length.to_f

    return average

end
```

9. In the same language, write the code you would need to call that function and print out the result.

```ruby
array = [1, 2, 3, 4]

print(main(array))
```

10. To the code from 9, add code to print the message “Double digits” if the average is above or equal to 10. Otherwise, print the message “Single digits”. Provide a screenshot of your program running.

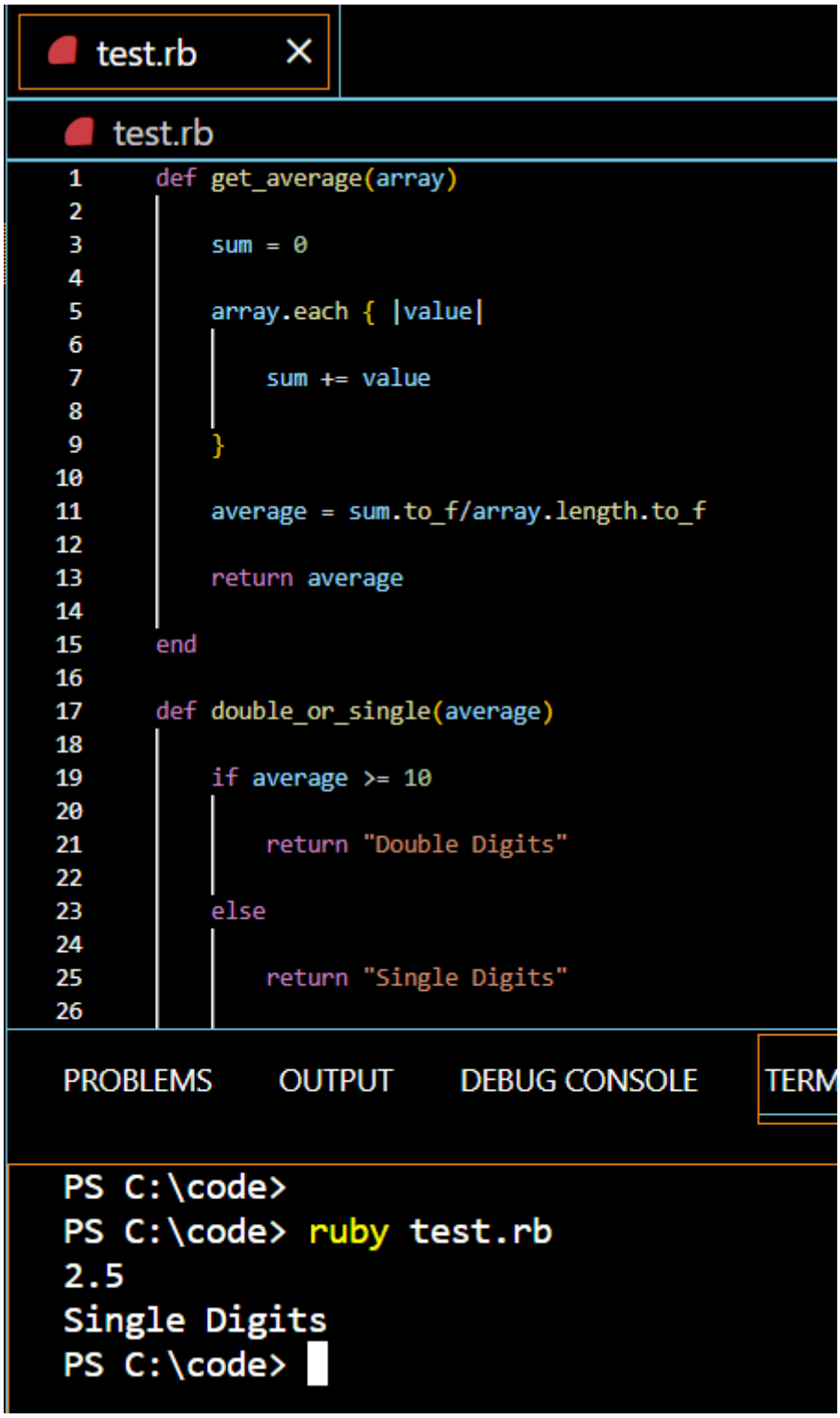
```ruby
def get_average(array)
    sum = 0
    array.each { |value|
        sum += value
    }
    average = sum.to_f/array.length.to_f
    return average
end

def double_or_single(average)
    if average >= 10
        return "Double Digits"
    else
        return "Single Digits"
    end
end

def main(array)
    average = get_average(array)
    print(average)
    puts ''
    print(double_or_single(average))
end

array = [1, 2, 3, 4]

main(array)
```


test.rb
test.rb
1 def get_average(array)
2
3 sum = 0
4
5 array.each { |value|
6
7 sum += value
8
9 }
10
11 average = sum.to_f/array.length.to_f
12
13 return average
14
15 end
16
17 def double_or_single(average)
18
19 if average >= 10
20
21 return "Double Digits"
22
23 else
24
25 return "Single Digits"
26
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERM
PS C:\code>
PS C:\code> ruby test.rb
2.5
Single Digits
PS C:\code>