1.1P: Preparing for OOP – Answer Sheet

Introduction

This answer sheet serves two purposes:

A. It serves as a revision for you of your previous learnings; and

B. It establishes a baseline understanding of tour knowledge in key Computer Science topics.

As such, this answer sheet is divided into the following areas of knowledge:

A. Your experience with UNIX/DOS console commands;

B. Your ability to differentiate between data types (e.g., text) and information categories (e.g., title);

C. Your experience with parsing and evaluating expressions according to rules of precedence;

D. Your understanding of computer science concepts and various programming language constructs;

E. Finally we want you to develop a simple function called *Average*. You will develop a fully functional program in three steps:

   1) Implement the Average function,

   2) Define a main function calling Average, and

   3) Define tests and output the qualifies the result on calling Average on a given array of numbers.

Section A: Console commands

Explain the following terminal instructions:

a) ***cd***:

change directory to change directiories in the terminal

b) ***pwd***:

print working directory output path of the current directory

c) ***mkdir***:

make directory creates new directory

d) ***cat***:

concatenate , view the contents of the file

e) ***ls***:

command lists the files and directories in the current directories

Section B: Data types and Information Classes

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

Information Category	Suggested Data Type
A person's family name	string
A person's age in years	integer
A person's weight in kilograms	float
A telephone number	integer
A temperature on the Kelvin scale	float
The average age of a group of children	float
Whether a student has passed this task	boolean

2. Aside from the examples already provided in question 1, consider the following list and find an example of information that could be stored as:

Data type	Suggested Information Category
String	address
Integer	no of cars owned
Float	bank account balance
Boolean	power on or off

Section C: Parsing and Evaluating Expressions

Fill out the **last** two columns of the following table. Parse and evaluate each expression. Enter the resulting value in column 3 and specify, in column 4, the data type of the expression in a complier "*friendly*" form (i.e., in a language format accepted, for example, in C, C#, or Pascal):

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 = 3	8.0	float
a + 2 * b	a = 2.5 b = 3	8.5	float
(a + b) * c	a = 1 b = 2 c = 3	9	integer
"Fred" + " Flintstone"		Fred Flinstone	string
a + " Rubble"	a = "Barney"	Barney Rubble	string

Section D: Computer Science and Programming Language Concepts:

1. Using an example, explain the difference between **declaring** and **initializing** a variable.

The difference between the two is….<*finish the sentence*>
declaring is specifying its type and anme
intializing is assigning value

Insert your example here:

```
int age = 10;
```

2. Explain the concept *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….<*finish the sentence*>
a variable passed into a function or procedure

Insert your example here:

```
def greet(name):
print("Hello, {name}!")

greet ("Alice")
```

3. Using code examples, describe the term *scope* as it is being used in programming (not in business or project management). In your explanation, focus on procedural programming. Ensure that you cover as many kinds of scopes and their respective differences as possible. Your answer must detail at least two kinds of scopes.

A scope is….<*finish the sentence*>
accesibility of variables within different parts of the program.
local scope means it is only accesible in one function
global scope means it is accesible from anywhere

Insert your examples here:

```
global_variable = 10


def function():
local_variable = 5
print(local_variable)

print(global_variable)
function()
print(local_variable)
```

Section E: Programming Practice:

1. Using procedural style programming, 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: we want to see your understanding of algorithms. You must demonstrate appropriate use of parameters, returning and assigning values, and the use of control statements like a loop. Note — just write the function Average at this point. In the next question we will ask you to *invoke the Average function.* You are not required to develop a complete program or even specify code that outputs anything at this stage. Average is a pure function. Input/output and any business logic processing is the responsibility of the (main line) calling the function Average.

   Insert your code here:

```
def average(numbers)
total = 0
numbers.each do |number|
total += number
end
total.to_f /numbers.length
end
```

2. Using the same language, write a main function you would to set up data (i.e., declare an array and initialize it with proper values), call the Average function, and print out the result. You are not required to provide input processing logic; you can have an inline instantiate of the collection of data values that the Average function needs to calculate the average of. Please use a reasonable data set, that is, the array must contain at least five elements of numerical type.

   Insert your code here:

```
def main():
numbers = [10,20,30,40,50]
avg = average(numbers)
puts "The average is: #{avg}"
end
main
```

3. Again using the same language, extend the main function with some output statements. Print the message “Double digits” if the average is above or equal to 10. Otherwise, print the message “Single digits”. And then, if the average is negative (e.g., the average of a week’s temperature readings at the Australian base in the Antarctic), add an additional line of output highlighting that the “Average value is in the negative”. Provide screenshot(s) of your program running, that is, the code and the run time output.

   Insert your code here:

```
def main
  numbers = [10, 20, 30, 40, 50]
  avg = average(numbers)
  puts "The average is: #{avg}"

  if avg >= 10
    puts "Double digits"
  else
    puts "Single digits"
  end

  if avg < 0
    puts "Average value is in the negative"
  end
end
```

Insert your whole program here:

```
def average(numbers)
  total = 0
  numbers.each do |number|
    total += number
  end
  total.to_f / numbers.length
end

def main
  numbers = [10, 20, 30, 40, 50]
  avg = average(numbers)
  puts "The average is: #{avg}"

  if avg >= 10
    puts "Double digits"
  else
    puts "Single digits"
  end

  if avg < 0
    puts "Average value is in the negative"
  end
end

main
```

End of Task

All students have access to the Adobe Acrobat tools. Please print your solution to PDF and submit via Canvas.