

1.1P: Preparing for OOP – Answer Sheet

1. Explain the following terminal instructions:
 - a. `cd`: 'change directory' – changes the current directory you are working from
 - b. `ls`: 'list files' – this will list all the current files in the current directory.
 - c. `pwd`: 'print working directory' – print name of current directory
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	Int
A phone number	String
A temperature in Celsius	Float
The average age of a group of people	Float
Whether a person has eaten lunch	Bool

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

Data type	Suggested Information
String	A street name
Integer	Street number
Float	Someone's exact height
Boolean	Is a playing card upside down

4. Fill out 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
5		5	Int
True			bool
a	a = 2.5	2.5	Float
1 + 2 * 3		7	Int
a and False	a = True	False	Bool
a or False	a = True	True	Bool
a + b	a = 1 b = 2	3	Int
2 * a	a = 3	6	Int
a * 2 + b	a = 1.5 b = 2	5	Float
a + 2 * b	a = 1.5 b = 2	5.5	Float
(a + b) * c	a = 1 b = 1 c = 5	10	Int
"Fred" + " Smith"		Fred smith	String
a + " Smith"	a = "Wilma"	Wilma smith	String

5. Explain the difference between **declaring** and **initialising** a variable.

The difference between the two is that declaring means creating or stating there is a variable where as initialising it means to give that variable a value.

6. Explain the term **parameter**. Write some code that demonstrates a simple use of a parameter.

A parameter is a special variable that is passed between functions or procedures

```
Execute | > Share | main.rb | STDIN | Result
1 # Hello World Program in Ruby
2 def example(test);
3   puts test
4 end
5
6
7
8 def main()
9   exampleParameter = example ("Passing this phrase to example function as a parameter")
10
11 end
12
13
14 main
```

\$ruby main.rb
Passing this phrase to example function as a parameter

7. Using an example, describe the term **scope**.

Scope is the visibility that the variable has in a program. This can be global (defined outside of a function group) or local (defined within)

```
Execute | > Share | main.rb | STDIN | Result
1 number = 1
2
3 def add_two(number)
4   number + 2
5
6 end
7
8 puts "this is from calling the function with the local number variable: " + add_two().to_s
9 puts "this is from calling the number global variable as defined outside of the function: " + number.to_s
```

\$ruby main.rb
this is from calling the function with the local number variable: 5
this is from calling the number global variable as defined outside of the function: 1

8. In any procedural language you like, write a function called Average, which accepts an array of integers and returns the average of those integers.

```
Execute | > Share | main.rb | STDIN | Result
1 def average(avArr);
2   avArr = avArr.sum / avArr.length
3   puts avArr
4 end
5
6
7
8 def main
9   arr = [1, 3, 6, 9, 50]
10  average(arr)
11 end
12
13
14 main
```

\$ruby main.rb
13

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

```
Execute | > Share | main.rb | STDIN | Result
1 def average(avArr);
2   avArr = avArr.sum / avArr.length
3   puts avArr
4 end
5
6
7
8 def main
9   arr = [1, 3, 6, 9, 50]
10  average(arr)
11 end
12
13
14 main
```

\$ruby main.rb
13

10. To the code from 9, add code to print the message “Double digits” if the average is above 10. Otherwise, print the message “Single digits”.



```
1 def average(avArr);  
2   avArr = avArr.sum / avArr.length  
3   return avArr  
4  
5 end  
6  
7  
8 def main()  
9   arr = [1, 3, 6, 9, 50]  
10  avArr = average(arr)  
11  if avArr >= 10  
12    puts ("Double Digits")  
13  else  
14    puts ("Single Digits")  
15  end  
16  
17 end  
18  
19 main  
20
```

Result

```
$ruby main.rb  
Double Digits
```