

1.1P: Preparing for OOP – Answer Sheet

1. Explain the following terminal instructions:
 - a. cd: change directory, navigate and redirect folder
 - b. ls: list, displays the names of the files and directories present in the current directory.
 - c. pwd: print working directory, display the current directory's full path
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	integer
A temperature in Celsius	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	A pet's name
Integer	Someone's home address
Float	The ml of a cup of water
Boolean	If anyone has eaten lunch

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

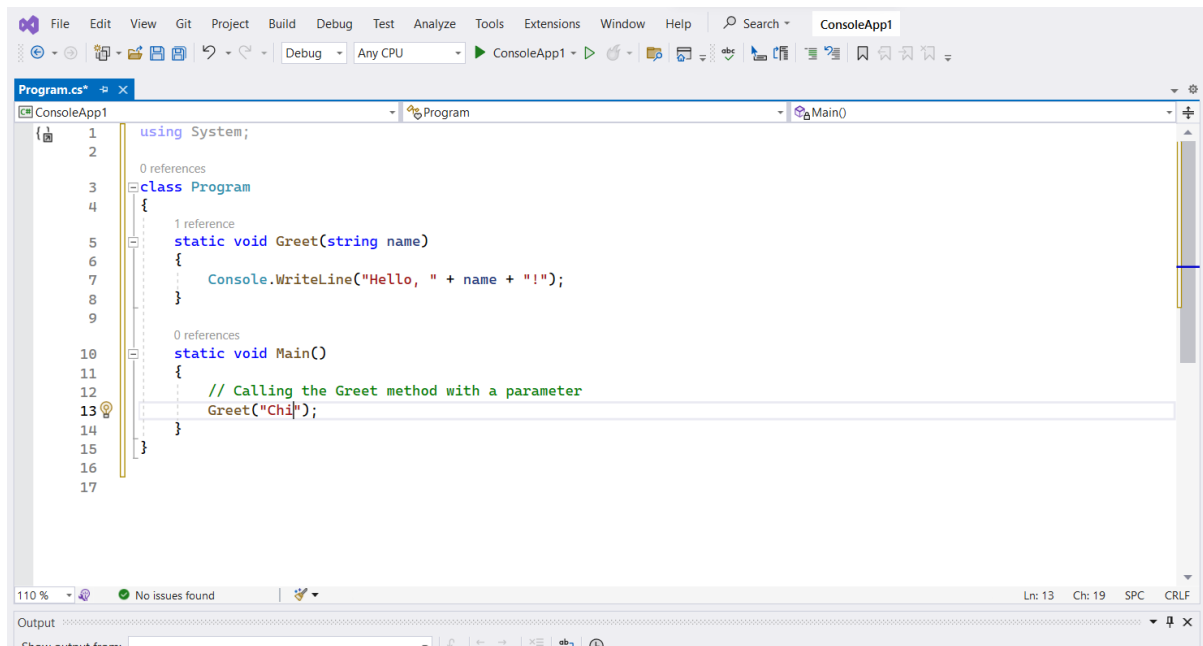
True		True	boolean
a	a = 2.5	2.5	Float
1 + 2 * 3		7	integer
a and False	a = True	False	Boolean
a or False	a = True	True	Boolean
a + b	a = 1 b = 2	3	integer
2 * a	a = 3	6	integer
a * 2 + b	a = 2.5 b = 2	7	integer
a + 2 * b	a = 2.5 b = 2	6.5	float
(a + b) * c	a = 1 b = 1 c = 5	10	Integer
"Fred" + " Smith"		"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 declaring is creating space in the computer for a data type, while initialising is assigning an initial value to that storage location.

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 a variable or value that is passed into a procedure or function. It allows the procedure or function to accept input values and perform



- 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, in the context of procedural programming, refers to the visibility and accessibility of variables, functions, and other program elements within a specific portion of code

Global scope:

D: > COS20007 > Week 1 > AnswerSheet

```

1 # Global scope
2 global_var = 10
3
4 def global_function():
5     print("This is a global function.")
6
7 # Accessing global variables and functions
8 print(global_var)
9 global_function()
10

```

Local scope:

```

10
11 def local_function
12   local_var = 20
13   puts local_var
14 end
15
16 # Accessing local variable within the function
17 local_function
18
19 # Trying to access local variable outside the function will result in an error
20 puts local_var # This will raise an error
21

```

Block scope

```

if true
  block_var = 30
  puts block_var # Accessing block variable within the if block
end

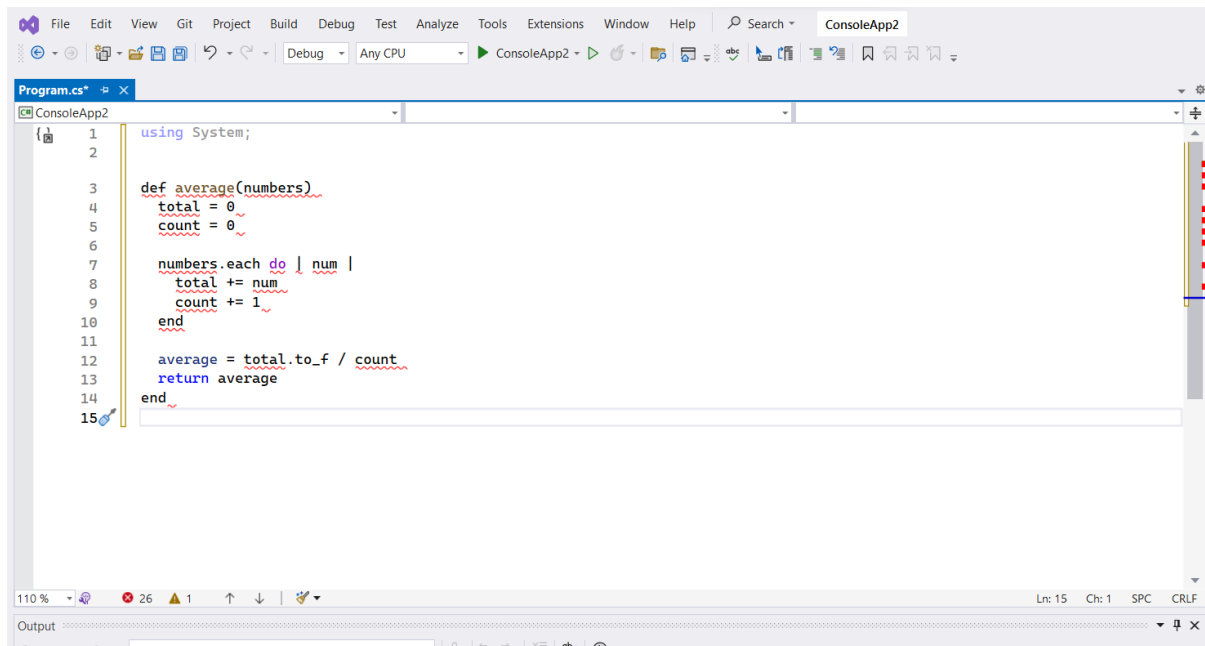
# Trying to access block variable outside the if block will result in an error
puts block_var # This will raise an error

```

Global variables in Ruby are those that begin with a dollar sign (\$), and they can be accessed from anywhere in the program. A method or block's declared variables have a local scope and are only used within that method or block. The specific construct, such as an if statement or a loop, determines the block scope in Ruby, and variables declared within those constructions are only available within them.

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.

Function called average that accepts an array of integers and returns the average:

A screenshot of a Ruby IDE window titled 'ConsoleApp2'. The editor shows a file named 'Program.cs' with the following Ruby code:

```
1 using System;
2
3 def average(numbers)
4   total = 0
5   count = 0
6
7   numbers.each do | num |
8     total += num
9     count += 1
10  end
11
12  average = total.to_f / count
13  return average
14 end
15
```

The status bar at the bottom indicates 'Ln: 15 Ch: 1 SPC CRLF'. The output window is empty.

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

Code to call the average function and print out the result:

```
6
7 numbers = [5, 10, 15, 20, 25]
8 result = average(numbers)
9 puts "Average: #{result}"
10
```

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.

```
1
2 ▾ def average(numbers)
3   total = 0
4   count = 0
5
6 ▾   numbers.each do |num|
7     total += num
8     count += 1
9   end
10
11   average = total.to_f / count
12   return average
13 end
14
15
16 numbers = [5, 10, 15, 20, 25]
17 result = average(numbers)
18 puts "Average: #{result}"
19
20 numbers = [5, 10, 15, 20, 25]
21 result = average(numbers)
22
23 ▾ if result >= 10
24   puts "Double digits"
25 ▾ else
26   puts "Single digits"
27 end
28
29
30 |
```



```
Average: 15.0
Double digits
```