Swinburne University of Technology

COS20007 Object Oriented Programming

Preparing for Object Oriented Programming

PDF generated at on Monday 14th August, 2023

1.1P: Preparing for OOP – Answer Sheet

1. Explain the following terminal instructions:
    a. cd: Change Directory- Used to change the current working directory in the terminal
    b. ls: List Files -Used to list the files and directories in the current working directory
    c. pwd: Print Directory – Used to print the absolute path of the current working 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	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	Device Name
Integer	Device Age
Float	Device Software version
Boolean	If device is working or not

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`	A=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`	7.0	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 DELCARING is doing specifying its type and name that is reserving a location in the memory.

Eg
Int a // declare a variable
Int a=35 // declare and intitialize variable a=35

Initialising means giving it an initial value to it this is done after declaring the value making it ready for use.

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 special kind of variable that allows you to pass values into a function or procedure. Parameter make the function more flexible and reusable as it enables to customize the behaviour of the function by passing different values every time it is called.

```
using System;

class Program
{
    // A function that takes two integer parameters and prints their sum
    static void AddAndPrint(int num1, int num2)
    {
        int sum = num1 + num2;
        Console.WriteLine($"The sum of {num1} and {num2} is: {sum}");
    }

    static void Main()
    {
        // Call the function with different values for the parameters
        AddAndPrint(7, 5); // Output: The sum of 7 and 5 is: 12
        AddAndPrint(20, 40); // Output: The sum of 40 and 20 is : 60
    }
}

```

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 the visibility and accessibility of variables within different parts of the code. It defines where a variable is used or accessed in the program.

There are 2 main types of scope -

1. Global scope – Declared at the top of the program, can be accessed and modified from any part of the code. Can cause unintended side effects.

```
using System;

// Global variable
int globalVar = 56;

void Main()
{
    Console.WriteLine("GlobalVar (Global Scope): " + globalVar);
}

Main();

```

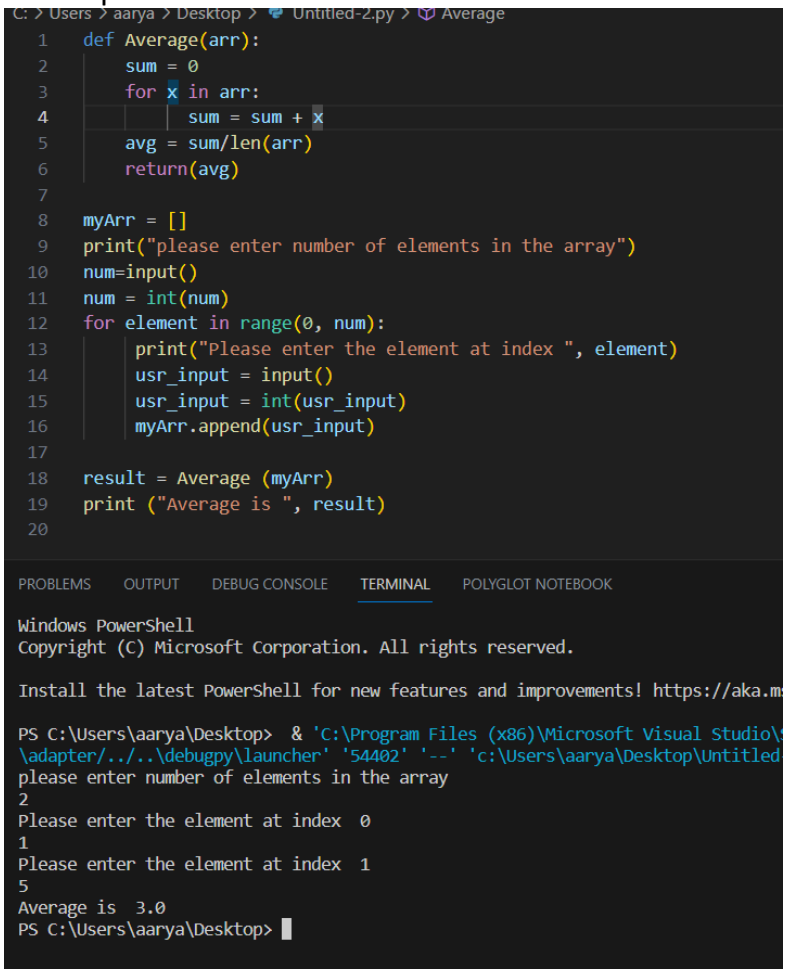
2. Local scope – Variable declared inside a block of code having a local function, only accessible inside the block or function they are declared in. Prevent unintended side effects.

```
using System;

void Function()
{
    // Local variable
    int localVar = 56;
    Console.WriteLine("LocalVar (Local Scope): " + localVar);
}

Function();

```

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.

```
def Average(arr):
    sum = 0
    for x in arr
        sum = sum + x
    avg = sum/len(arr)
    return(avg)
```

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



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.

C: > Users > aarya > Desktop > Untitled-1.py > Average

```python
def Average(arr):
    sum = 0
    for x in arr:
        sum = sum + x
    avg = sum/len(arr)
    return(avg)

myArr = []
print("please enter number of elements in the array")
num=input()
num = int(num)
for element in range(0, num):
    print("Please enter the element at index ", element)
    usr_input = input()
    usr_input = int(usr_input)
    myArr.append(usr_input)

result = Average (myArr)
print ("Average is ", result)

if result >= 10:
    print("double digit")
else:
    print("single digit")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL POLYGLOT NOTEBOOK

```
3
Please enter the element at index  0
55
Please enter the element at index  1
55
Please enter the element at index  2
47
Average is  52.333333333333336
double digit
PS C:\Users\aarya\Desktop>
```