# Worksheet 1 Basic concepts of OOP Answers

**Task 1**

Write a program to implement the pseudocode for the procedural Fish example:

Sub feed(Byref state, Byref size)

 size 🡨 size + 1

 Output "Fish fed"

 If size = 5 Then

 state 🡨 "FISH"

 End If

End Sub

thisFishState 🡨 "Fish"

thisFishSize 🡨 1

Output thisFishState, " is of size ", thisFishSize

While thisFishState <> "FISH"

 feed(thisFishState, thisFishSize)

Endwhile

Output "It is now a big ", thisFishState

**Python**

def feed(state, size):

 size += 1

 print("Fish fed")

 if size == 5:

 state = "FISH"

 return state, size

thisFishState = "Fish"

thisFishSize = 1

print(thisFishState, "is of size", thisFishSize)

while thisFishState != "FISH":

 thisFishState, thisFishSize = feed(thisFishState, thisFishSize)

print("It is now a big", thisFishState)

**VB.net**

Sub feed(ByRef state, ByRef size)

 size += 1

 Console.WriteLine("Fish fed")

 If size = 5 Then

 state = "FISH"

 End If

End Sub

Sub Main()

 Dim thisFishState As String

 Dim thisFishSize As Integer

 thisFishState = "Fish"

 thisFishSize = 1

 Console.WriteLine(thisFishState & " is of size " & thisFishSize)

 Do While thisFishState <> "FISH"

 feed(thisFishState, thisFishSize)

 Loop

 Console.WriteLine("It is now a big " & thisFishState)

 Console.ReadLine()

End Sub

 **Task 2**

Write program statements to declare the class Animal, as shown in the pseudocode:

Animal = Class

 Public

 Constructor (s, n)

 Procedure feed()

 Function getState()

 Function getSize()

 Private

 state: String

 size: Integer

 Animal.Constructor (s, n)

 Animal.state 🡨 s

 Animal.size 🡨 n

 End Constructor

 Procedure Animal.feed()

 Animal.size 🡨 Animal.size + 1

 If Animal.size = 5 Then

 Animal.state 🡨 "FISH"

 End If

 End Procedure

 Function Animal.getState()

 Return Animal.state

 End Function

 Function Animal.getSize()

 Return Animal.size

 End Function

End Class

**Python**

class Animal:

 def \_\_init\_\_(self,s, n):

 self.\_\_state = s

 self.\_\_size = n

 def getState(self):

 return self.\_\_state

 def getSize(self):

 return self.\_\_size

 def feed(self):

 self.\_\_size += 1

 print("Fish fed")

 if self.\_\_size == 5:

 self.\_\_state = "FISH"

**VB.net**

 Class Animal

 Private state As String

 Private size As Integer

 Public Sub New(ByVal s, ByVal n)

 state = s

 size = n

 End Sub

 Function getState() As String

 Return state

 End Function

 Function getSize() As Integer

 Return size

 End Function

 Public Sub feed()

 size = size + 1

 Console.WriteLine("Fish fed")

 If size = 5 Then

 state = "FISH"

 End If

 End Sub

 End Class

 **Task 3**

Write program statements to implement the pseudocode for the OOP Fish example:

thisFish 🡨 new Animal("Fish", 1)

Output thisFish.getState()

Output " is of size ", thisFish.getSize()

While thisFish.getState() <> "FISH"

 thisFish.feed()

Endwhile

Output "It is now a big "

Output thisFish.getState()

**Python**

thisFish = Animal("Fish", 1)

print(thisFish.getState(), "is of size", thisFish.getSize())

while thisFish.getState() != "FISH":

 thisFish.feed()

print("It is now a big", thisFish.getState())

**VB.net**

Dim thisFish As New Animal("Fish", 1)

Console.Write(thisFish.getState())

Console.WriteLine(" is of size " & thisFish.getSize())

Do While thisFish.getState() <> "FISH"

 thisFish.feed()

Loop

Console.Write("It is now a big " & thisFish.getState())

Console.ReadLine()

 **Task 4**

Write program code for a class for a car object. The attributes required are:

Registration

Make

Mileage

DateOfInspection

The constructor is to set the mileage driven to 0 and registration and make are supplied as parameter values during instantiation.

Other methods required are:

Getters for registration, make, mileage and date of inspection

Setter for inspection data (mileage driven and date of inspection)

**Python**

class Car:

 def \_\_init\_\_(self,registration, make):

 self.\_\_registration = registration

 self.\_\_make = make

 self.\_\_mileage = 0

 self.\_\_dateOfInspection = ""

 def getRegistration(self):

 return self.\_\_registration

 def getMake(self):

 return self.\_\_make

 def getMileage(self):

 return self.\_\_mileage

 def getDateOfInspection(self):

 return self.\_\_dateOfInspection

 def setInspectionData(self, mileage, date):

 self.\_\_mileage = mileage

 self.\_\_dateOfInspection = date

**VB.net**

Class Car

 Private registration As String

 Private make As String

 Private mileage As Integer

 Private dateOfInspection As Date

 Public Sub New(ByVal r, ByVal m)

 registration = r

 make = m

 mileage = 0

 dateOfInspection = #01/01/1999#

 End Sub

 Function getRegistration() As String

 Return registration

 End Function

 Function getMake() As String

 Return make

 End Function

 Function getMileage() As Integer

 Return mileage

 End Function

 Function getDateOfInspection() As Date

 Return dateOfInspection

 End Function

 Public Sub setInspectionData(ByVal m, ByVal d)

 mileage = m

 dateOfInspection = d

 End Sub

End Class

 **Task 5**

Write program code to test your car class. Instantiate a car with a chosen registration and make. Set inspection data with a number of miles and an inspection date. Then use each of the getter methods and output the data with relevant messages.

**Python**

myCar = Car("ABC", "VW")

print("I have a", myCar.getMake(),end="" )

print(" with registration", myCar.getRegistration())

myCar.setInspectionData(1234, "12/4/2016")

print("My", myCar.getMake(), "registration ",end="")

print(myCar.getRegistration(), "was inspected on ",end="")

print(myCar.getDateOfInspection(), "and has done ",end="")

print(myCar.getMileage(), "miles")

**VB.net**

Dim myCar As Car = New Car("ABC", "VW")

Console.Write("I have a " & myCar.getMake())

Console.WriteLine("with registration" & myCar.getRegistration())

myCar.setInspectionData(1234, "12/4/2016")

Console.Write("My " & myCar.getMake() & " registration ")

Console.Write(myCar.getRegistration() & "was inspected on") Console.Write(myCar.getDateOfInspection() & "and has done")

Console.WriteLine(myCar.getMileage() & "miles")

Console.ReadLine()