Questionnaires

Questionnaire Week 4

1. **Which of the following pieces of code can be used to define a class.**

```
# Code 1
class Questionnaire ():
	def __init__ (self, question, answer):
		self.listQuestion.append (question)
		self.listAnswers.append (answer)

# Code 2
class Questionnaire ():
	listQuestion = []
	listAnswers = []
	def __init__ (self, question, answer):
		self.listQuestion.append(question)
		self.listAnswers.append(answer)

# Code 3
class Questionnaire ():
	listQuestion = []
	listAnswers = []
	def __init__ (question, answer):
		self.listQuestion = question
		self.listAnswers = answer

# Code 4
class Questionnaire ():
	def __init__ (self, question, answer):
		self.listQuestion = question
		self.listAnswers = answer
```

2. **Assume we have the following code, how can we access to the class information (e.g. attributes)?**

```
class Rectangle ():
	width = 0
	height = 0
	def __init__ (self, w, h):
		self.width = w
		self.height = h
```

```
        def getWidth (self):
                return self.width
if __name__ == '__main__':
        # Option 1
        myRectangle = Rectangle(5, 5)
        myRectangle.self.width
        #Option 2
        self.width
        #Option 3
        self.getWidth()
        #Option 4
        myRectangle.width
        #Option 5
        myRectangle.getWidth()
```

3. **Assuming the following class definition, fix the errors:**

```
class Rectangle ():
        __width = 0
        __height = 0
        def Rectangle (w, h):
                width = w
                height = h
        def getWidth ():
                return width
        def Area ():
                return width * height
        def __Perimeter ():
                return 2*width + 2*height
if __name__ == '__main__':
        myRectangle = Rectangle (5, 5)
        print "Area: " + str (myRectangle.Area())
        print "Perimeter: "+ str (myRectangle.Perimeter())
```

```
# Solution
class Rectangle ():
        __width =0
        __height =0
        def __init__ (self, w, h):
                self.__width = w
                self.__height = h
        def getWidth (self):
                return self.__width
```

```
        def Area (self):
                return self.__width * self.__height
        def Perimeter (self):
                return 2*self.__width + 2*self.__height
if __name__ == '__main__':
        myRectangle = Rectangle (5, 5)
        print "Area: " + str (myRectangle.Area())
        print "Perimeter: "+ str (myRectangle.Perimeter())
```

4. **Assuming the following class definition in a Python program, which of the following sentences can be used to define Rectangle as a child class of Figure?**

```
class Figure ():
        __name = ""
        def __init__(self, name):
                self.__name = name

        def getName (self):
                return self.__name

class Rectangle ():
        # Variable definitions...
        def __init__ (self, w, h):
                # Variable initializers...
        def getWidth (self):
                return self.__width
        Code 1:
class Figure (object):
        ...
class Rectangle (Figure):
        ...
        def __init__ (self, w, h):
                self.__width = w
                self.__height = h
                super (Rectangle, self).__init__("Rectangle")
        Code 2:
class Figure ():
        ...
class Rectangle (Figure):
        ...
        def __init__ (self, w, h):
                self.__width = w
```

```
                self.__height = h
                super (Rectangle, self).__init__("Rectangle")
        Code 3:
class Figure ():
        ...
class Rectangle (Figure):
        ...
        def __init__ (self, w, h):
                self.__width = w
                self.__height = h
                Figure.__init__(self, "Rectangle")
        Code 4:
class Figure ():
        ...
class Rectangle (Figure):
        ...
        def __init__ (self, w, h):
                self.__width = w
                self.__height = h
                Figure.Figure("Rectangle")
```

5. **Given the following piece of code, correct or add necessary parts**

```
class OperatingSystem ():
        private name = ""
        private year = ""
        def OperatingSystem(name, year):
                this.name = name
                this.year = year
        def setName (name):
                this.name = name
        def getName ():
                return this.name
        def setYear (year):
                this.year = year
        def getYear ():
                return this.year
        def private strPrintOperatingSystem ():
                return this.name + " "+ this.year
        def printOperatingSystem ():
                print this.strPrintOperatingSystem()

class Linux ():
        type = ""
```

```
        creator = ""
        def Linux (typ, crea):
                type = typ
                creator = crea
                OperatingSystem ("Linux", "2014")
        def printLinux ():
                print " "+ this.type + " " + this.creator
if __name__ == "__main__":
        print "Hello"
        ubuntu = Linux ("Ubuntu","Richard Stalman")
        print " Ubuntu Year: " + ubuntu.getYear()
        ubuntu.printLinux()
        ubuntu.printOperativeSystem()

# Solution

class OperatingSystem (object):
        __name = ""
        __year = ""
        def __init__ (self, name, year):
                self.__name = name
                self.__year = year
        def setName (self, name):
                self.__name = name
        def getName (self):
                return self.__name
        def setYear (self, year):
                self.__year = year
        def getYear (self):
                return self.__year
        def __strPrintOperativeSystem (self):
                return self.__name + " " + self.__year
        def printOperatingSystem (self):
                print self.__strPrintOperativeSystem()

class Linux (OperatingSystem):
        __type = ""
        __creator = ""
        def __init__ (self, type, creator):
                self.__type = type
                self.__creator = creator
                OperatingSystem.__init__(self, "Linux", "2014")
        def printLinux (self):
                print " " + self.__type + " " + self.__creator

if __name__ == "__main__":
```

```
        print "Hello"
        ubuntu = Linux ("Ubuntu","Richard Stalman")
        print " Ubuntu Year: " + ubuntu.getYear()
        ubuntu.printLinux()
        ubuntu.printOperatingeSystem()
```