In [1]:

```python
# Example 01
class Shark:
    def swim(self):
        print('Shark is swimming.')
    def swim_backwards(self):
        print('Shark cannot swim bakwards')
    def skeleton(self):
        print('Skeleton of shark is made of crtilage')

class ClownFish:
    def swim(self):
        print('ClownFish is swimming')
    def swim_backwards(self):
        print('ÇlownFish can swim backwards')
    def skeleton(self):
        print('The skeleton of clownfish is made of bones')
```

executed in 16ms, finished 20:53:28 2020-06-22

In [4]:

```python
# Example 02
shark = Shark()
clown = ClownFish()
for fish in (shark,clown):
    fish.skeleton()
    fish.swim()
    fish.swim_backwards()
```

executed in 16ms, finished 20:56:28 2020-06-22

```
Skeleton of shark is made of crtilage
Shark is swimming.
Shark cannot swim bakwards
The skeleton of clownfish is made of bones
ClownFish is swimming
ÇlownFish can swim backwards
```

In [12]:

```python
# Example 03
class AudioFiles:
    def __init__(self, filename):
        if not filename.endswith(self.ext):
            raise Exception('Invalid file format!')
        self.filename = filename

class MP3(AudioFiles):
    ext = '.mp3'
    def play(self):
        print('playing {} as mp3'.format(self.filename))

class Wav(AudioFiles):
    ext = '.wav'
    def play(self):
        print('playing {} as wav'.format(self.filename))

class Ogg(AudioFiles):
    ext = '.ogg'
    def play(self):
        print('playing {} as ogg'.format(self.filename))

class FlacFile:
    def __init__(self, filename):
        if not filename.endswith('.flac'):
            raise Exception('Invalid file format!')
        self.filename = filename
    def play(self):
        print('playing {} as flac'.format(self.filename))
```

executed in 16ms, finished 21:01:36 2020-06-23

In [13]:

```python
mp3 = MP3('hello.mp3')
mp3.play()
flac = FlacFile('jason.flac')
flac.play()
```

executed in 16ms, finished 21:02:05 2020-06-23

```
playing hello.mp3 as mp3
playing jason.flac as flac
```

In [31]:

```python
# Example 04
class Parrot:
    def fly(self):
        print('Parrot can fly!')
    def swim(self):
        print('Parrot cannot swim!')


class Penguin:
    def fly(self):
        print('Penguin cannot fly!')
    def swim(self):
        print('Penguin can swim!')

# Explicit Function on common interface
def testing_birds(name):
    name.fly()
    name.swim()
```

executed in 16ms, finished 21:25:18 2020-06-22

In [32]:

```python
parrot = Parrot()
penguin = Penguin()

# Calling Explicit function
testing_birds(parrot)
testing_birds(penguin)
```

executed in 16ms, finished 21:25:19 2020-06-22

```
Parrot can fly!
Parrot cannot swim!
Penguin cannot fly!
Penguin can swim!
```

In [35]:

```python
# Example 05:
class Rectangle:
    def __init__(self, color, filled, width, length):
        self.__color = color
        self.__filled = filled
        self.__width = width
        self.__length = length
    def setColor(self, color):
        self.__color = color
    def getColor(self):
        return self.__color
    def isFilled(self):
        return  self.__filled
    def setFilled(self, fill):
        self.__filled = fill
    def getArea(self):
        return self.__length * __width

class Circle:
    def __init__(self, color, filled, radius):
        self.__color = color
        self.__filled = filled
        self.__radius = radius
    def setColor(self, color):
        self.__color = color
    def getColor(self):
        return self.__color
    def isFilled(self):
        return self.__filled
    def setFilled(self, fill):
        self.__filled = fill
    def getArea(self):
        return (3.14 * (self.__radius ** 2))
```

executed in 32ms, finished 23:05:06 2020-06-22

In [65]:

```python
# Example 06:
class Shape:
    def __init__(self, color='red', filled=False):
        self.__color = color
        self.__filled = filled
    def getColor(self):
        return self.__color
    def setColor(self, color):
        self.__color = color
    def isFilled(self):
        return self.__filled
    def setFilled(self, fill):
        self.__filled = fill

class Rectangle(Shape):
    def __init__(self, length, breadth):
        super().__init__()
        self.__length = length
        self.__breadth = breadth
    def getLength(self):
        return self.__length
    def setLength(self, length):
        self.__length = length
    def getBreadth(self):
        return self.__breadth
    def setBreadth(self, breadth):
        self.__breadth = breadth
    def getArea(self):
        return (self.__length * self.__breadth)
    def getPerimeter(self):
        return (2 * (self.__length + self.__breadth))

class Circle(Shape):
    def __init__(self, radius):
        super().__init__()
        self.__radius = radius
    def getRadius(self):
        return self.__radius
    def setRadius(self, radius):
        self.__radius = radius
    def getArea(self):
        return (3.14 * (self.__radius ** 2))
    def getPerimeter(self):
        return (2 * 3.14 * self.__radius)
```

executed in 112ms, finished 23:32:24 2020-06-22

In [71]:
```
print('\tRectangle Instance')
r1 = Rectangle(length=10, breadth=20)
print(r1.getArea())
print(r1.getColor())
r1.setColor('Blue')
print(r1.getColor())
print(r1.getPerimeter())
print('\tCircle Instance')
c1 = Circle(radius=30)
print(c1.getColor())
print(c1.isFilled())
print(c1.getArea())
```

executed in 16ms, finished 23:37:52 2020-06-22

```
        Rectangle Instance
200
red
Blue
60
        Circle Instance
red
False
2826.0
```

In [72]:
```
# Example 07
class Bear:
    def sound(self):
        print('Groarr!')

class Dog:
    def sound(self):
        print('Woof Woof!')

def makeSound(animal):
    animal.sound()
```

executed in 32ms, finished 00:09:54 2020-06-23

In [73]:
```
bear = Bear()
dog = Dog()
makeSound(bear)
makeSound(dog)
```

executed in 72ms, finished 00:10:37 2020-06-23

```
Groarr!
Woof Woof!
```

In [75]:

```python
# Exercise 01
class Automatic_Umbrella:
    def open(self):
        print('Automatic umbrella can be opened with one push button.')

    def waterProof(self):
        print('Automatic umbrella is waterproof.')

    def close(self):
        print('Automatic umbrella can be closed with one push button.')

class Paper_Umbrella:
    def open(self):
        print('Paper Umbrella is opened manually by dragging shaft.')

    def waterProof(self):
        print('Paper Umbrella is not waterproof.')

    def close(self):
        print('Paper Umbrella is closed manually by dragging shaft.')
```

executed in 11ms, finished 01:50:09 2020-06-23

In [80]:

```python
# Driver Code
autoUmbrella = Automatic_Umbrella()
paperUmbrella = Paper_Umbrella()
for umbrella in (autoUmbrella, paperUmbrella):
    umbrella.open()
    umbrella.waterProof()
    umbrella.close()
```

executed in 6ms, finished 01:51:44 2020-06-23

```
Automatic umbrella can be opened with one push button.
Automatic umbrella is waterproof.
Automatic umbrella can be closed with one push button.
Paper Umbrella is opened manually by dragging shaft.
Paper Umbrella is not waterproof.
Paper Umbrella is closed manually by dragging shaft.
```

In [4]:

```python
# Exercise 02
class ClassicalPhone:
    def pressButton(self):
        print('Button is pressed.')

    def touch(self):
        print('This phone does not support touch screen.')

    def getPhoneSize(self):
        print('4 inches Screen Display')

    def hasScreenSplit(self):
        print('No, this phone doesn\'t support screen split.')

    def getOS(self):
        print('Nokia OS.')

class SmartPhone:
    def pressButton(self):
        print('Smartphone doesn\'t have button.')

    def touch(self):
        print('Smartphone have touch screen.')

    def getPhoneSize(self):
        print('8 inches HD Screen Display')

    def hasScreenSplit(self):
        print('Smartphone can split screen.')

    def getOS(self):
        print('Anrdoid 10')

# Explicit Function
def phoneTest(phoneName):
    phoneName.pressButton()
    phoneName.touch()
    phoneName.getPhoneSize()
    phoneName.hasScreenSplit()
    phoneName.getOS()
```

executed in 11ms, finished 19:49:56 2020-06-23

In [6]:

```
# Drive Code
nokia = ClassicalPhone()
samsung = SmartPhone()

for phone in (nokia, samsung):
    phoneTest(phone)
```

executed in 8ms, finished 19:55:09 2020-06-23

```
Button is pressed.
This phone does not support touch screen.
4 inches Screen Display
No, this phone doesn't support screen split.
Nokia OS.
Smartphone doesn't have button.
Smartphone have touch screen.
8 inches HD Screen Display
Smartphone can split screen.
Anrdoid 10
```

In [8]:

```
# Task 03
class GrasslandButterfly:
    def getColor(self):
        print('They have orange wings.')

    def getPattern(self):
        print('Row of white spots at edges of wings.')

    def livingNature(self):
        print('They live around flowers and meadows.')

    def living_as_pupa(self):
        print('One Month in Pupa')

    def living_as_caterpillar(self):
        print('Two Weeks as Caterpillar')

class ExoticButterfly:
    def getColor(self):
        print('They have iridescent blue wings.')

    def getPattern(self):
        print('Several eyespots on downwings.')

    def livingNature(self):
        print('They live in tropical areas around the equator.')

    def living_as_pupa(self):
        print('Three weeks in Pupa')

    def living_as_caterpillar(self):
        print('One week as caterpillar')
```

executed in 13ms, finished 20:48:28 2020-06-23

In [11]:

```
# Driver Code
butterfly1 = GrasslandButterfly()
butterfly2 = ExoticButterfly()

for butterfly in (butterfly1, butterfly2):
    butterfly.getColor()
    butterfly.getPattern()
    butterfly.livingNature()
    butterfly.living_as_pupa()
    butterfly.living_as_caterpillar()
```

executed in 8ms, finished 20:51:38 2020-06-23

```
They have orange wings.
Row of white spots at edges of wings.
They live around flowers and meadows.
One Month in Pupa
Two Weeks as Caterpillar
They have iridescent blue wings.
Several eyespots on downwings.
They live in tropical areas around the equator.
Three weeks in Pupa
One week as caterpillar
```