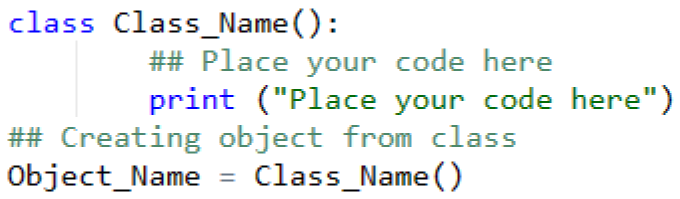
SCHOOL OF
ENGINEERING
BANGKOK UNIVERSITY

Class programming and linked list

AIE 311 : Data structure and Algorithm

Class

- Class is an object of programming
  - Class/ object is very useful to duplicate block of code
  - There are 2 types of creating class
    - Class without initial function/ value
    - Class with initial function/ value
  - E.g., if class has 3 parametres (x, y, z)
    - Every new objects will contain x, y, z

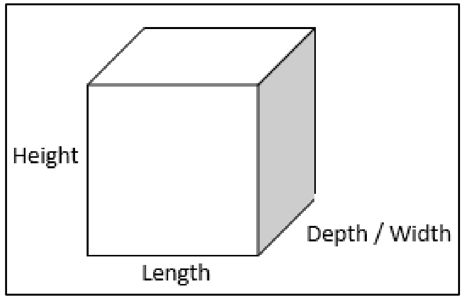


```
class Class_Name():
        ## Place your code here
        print ("Place your code here")
## Creating object from class
Object_Name = Class_Name()
```

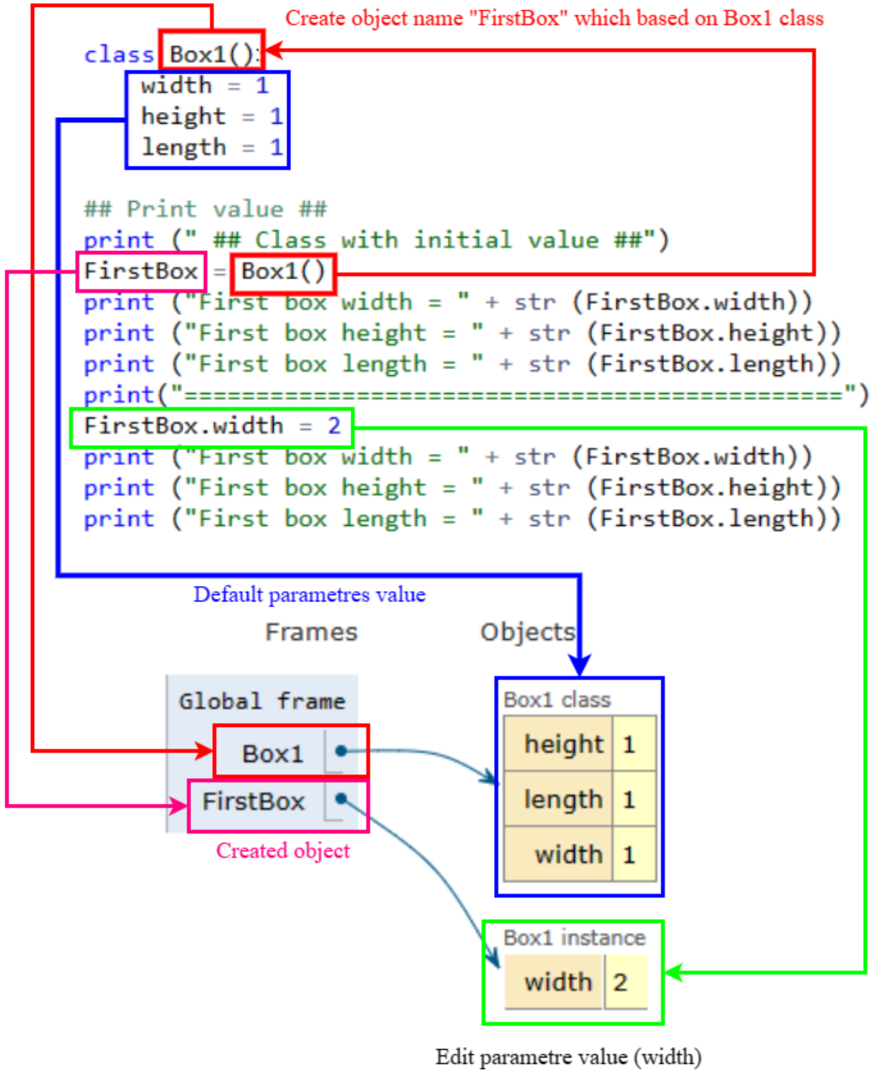
Class example

SCHOOL OF ENGINEERING
BANGKOK UNIVERSITY

- Create "box" class
  - This class has 3 parametres (Height, Length, Width)
    - Every new objects will contain Height, Length, Width



Class with initial function/ value



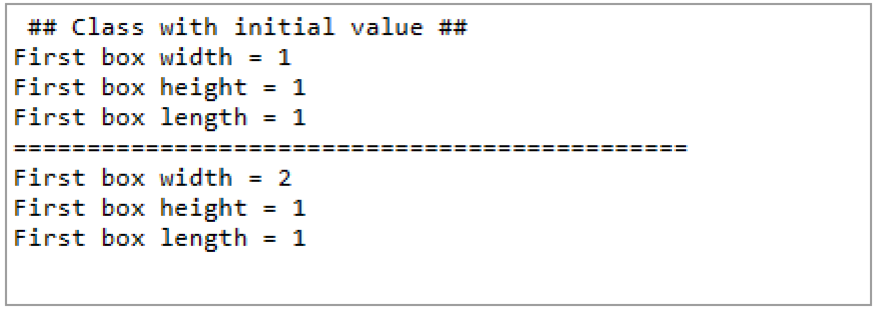
Create object name "FirstBox" which based on Box1 class

```
class Box1():
    width = 1
    height = 1
    length = 1

## Print value ##
print (" ## Class with initial value ##")
FirstBox = Box1()
print ("First box width = " + str (FirstBox.width))
print ("First box height = " + str (FirstBox.height))
print ("First box length = " + str (FirstBox.length))
print("=============================================")
FirstBox.width = 2
print ("First box width = " + str (FirstBox.width))
print ("First box height = " + str (FirstBox.height))
print ("First box length = " + str (FirstBox.length))
```

Default parametres value

Created object

Edit parametre value (width)

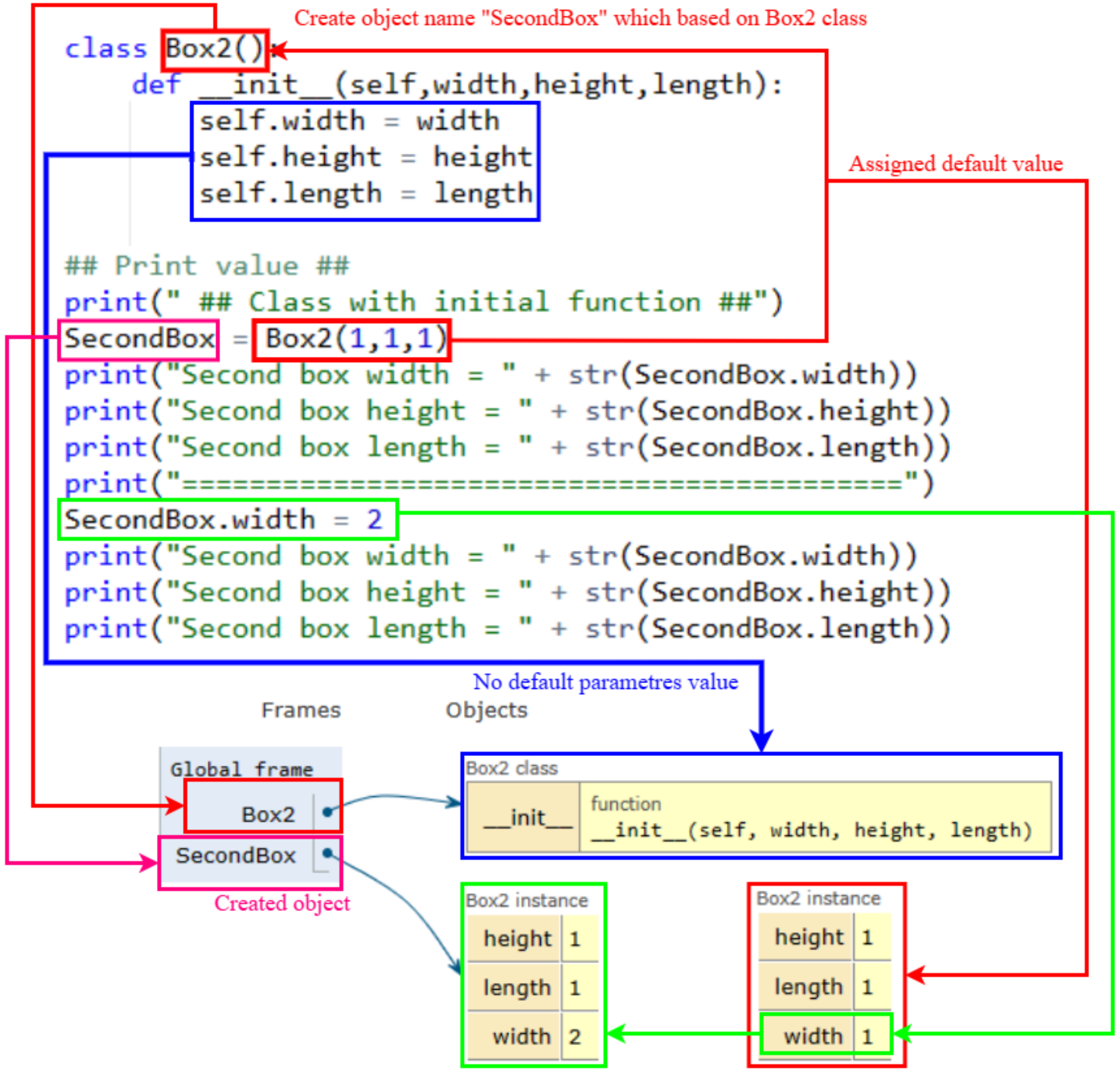


```
 ## Class with initial value ##
First box width = 1
First box height = 1
First box length = 1
=============================================
First box width = 2
First box height = 1
First box length = 1
```

When create a new class. New parametres are not required

FirstBox = **Box1()**

Class with initial function/ value



```
class Box2():
    def __init__(self,width,height,length):
        self.width = width
        self.height = height
        self.length = length

## Print value ##
print(" ## Class with initial function ##")
SecondBox = Box2(1,1,1)
print("Second box width = " + str(SecondBox.width))
print("Second box height = " + str(SecondBox.height))
print("Second box length = " + str(SecondBox.length))
print("========================================")
SecondBox.width = 2
print("Second box width = " + str(SecondBox.width))
print("Second box height = " + str(SecondBox.height))
print("Second box length = " + str(SecondBox.length))
```

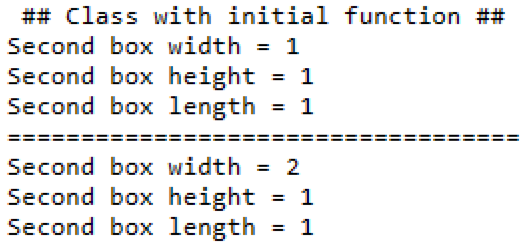
Create object name "SecondBox" which based on Box2 class

Assigned default value

No default parametres value

Created object

Edit parametre value (width)

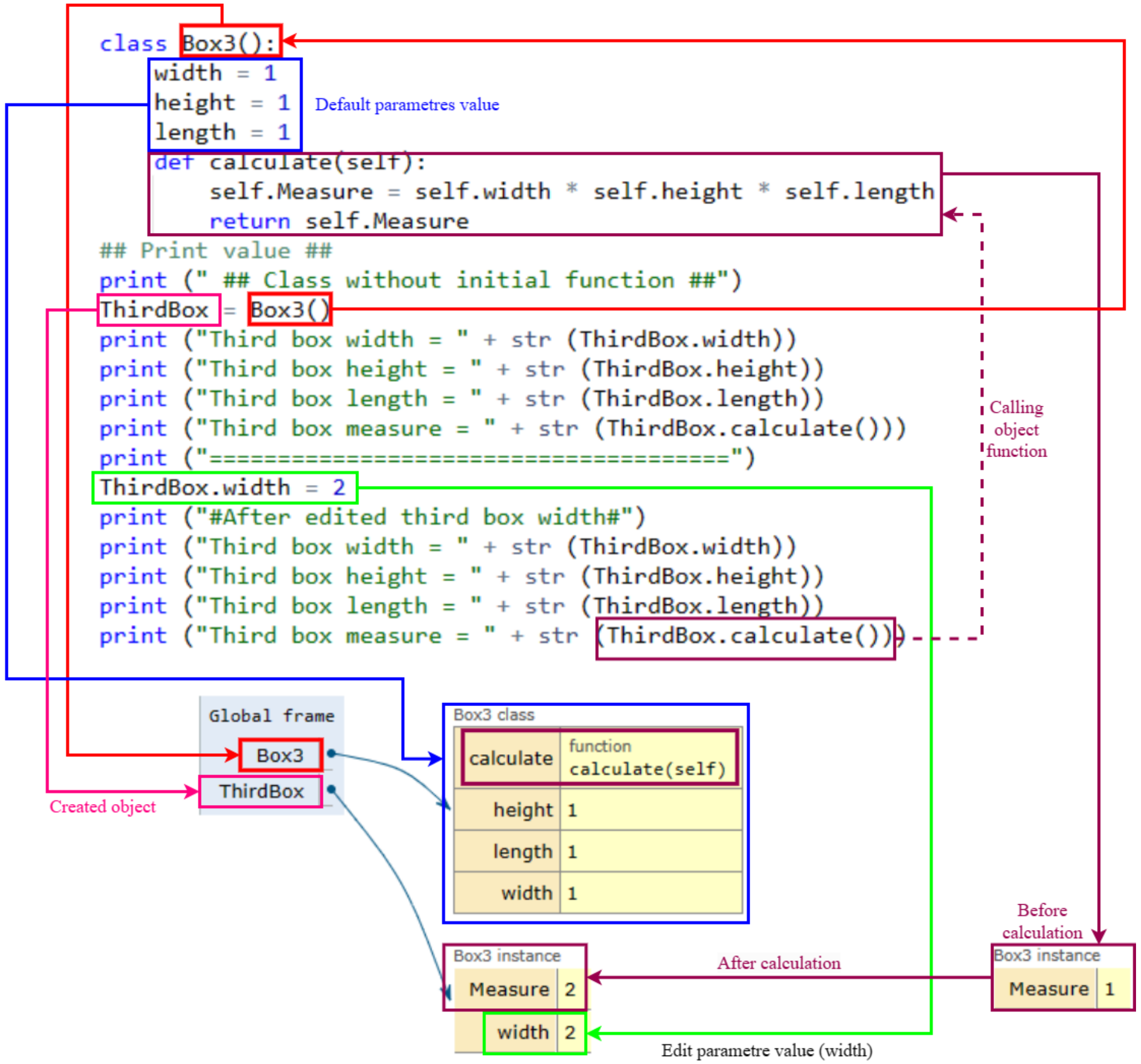


```
 ## Class with initial function ##
Second box width = 1
Second box height = 1
Second box length = 1
========================================
Second box width = 2
Second box height = 1
Second box length = 1
```

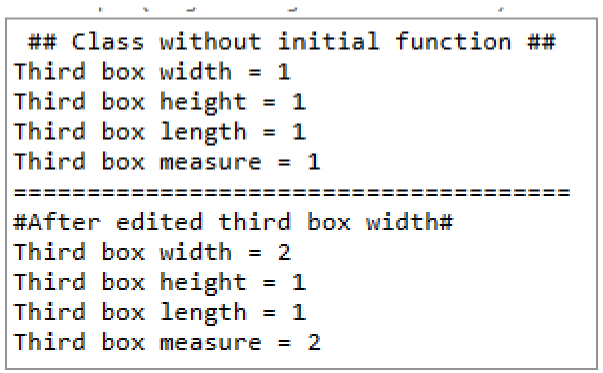
When create a new class. New parametres are required

SecondBox = **Box2(<u>1,1,1</u>)**

Calling function in class



```
class Box3():
    width = 1
    height = 1
    length = 1
    def calculate(self):
        self.Measure = self.width * self.height * self.length
        return self.Measure
## Print value ##
print (" ## Class without initial function ##")
ThirdBox = Box3()
print ("Third box width = " + str (ThirdBox.width))
print ("Third box height = " + str (ThirdBox.height))
print ("Third box length = " + str (ThirdBox.length))
print ("Third box measure = " + str (ThirdBox.calculate()))
print ("=====================================")
ThirdBox.width = 2
print ("#After edited third box width#")
print ("Third box width = " + str (ThirdBox.width))
print ("Third box height = " + str (ThirdBox.height))
print ("Third box length = " + str (ThirdBox.length))
print ("Third box measure = " + str (ThirdBox.calculate()))
```



```
 ## Class without initial function ##
Third box width = 1
Third box height = 1
Third box length = 1
Third box measure = 1
=====================================
#After edited third box width#
Third box width = 2
Third box height = 1
Third box length = 1
Third box measure = 2
```

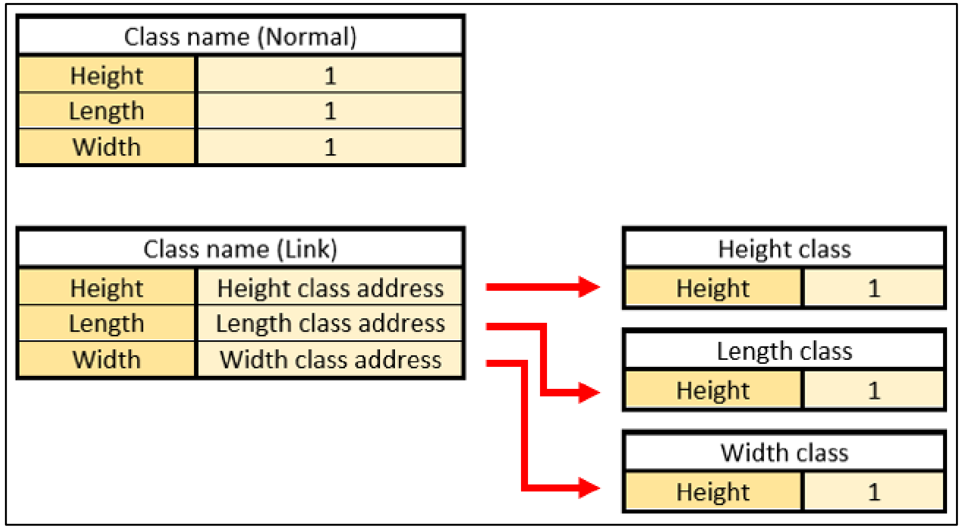
To use function in class the format is showing below

Object_name.functionname()

ThirdBox.calculate()

Link class to other class

- Normal class instance creation
  - Classes contain parametres which parametre contains <u>its value</u>
- Linked class instance creation
  - Classes contain parametres which parametre contains <u>another class</u>



Class name (Normal)	
Height	1
Length	1
Width	1

Class name (Link)	
Height	Height class address
Length	Length class address
Width	Width class address

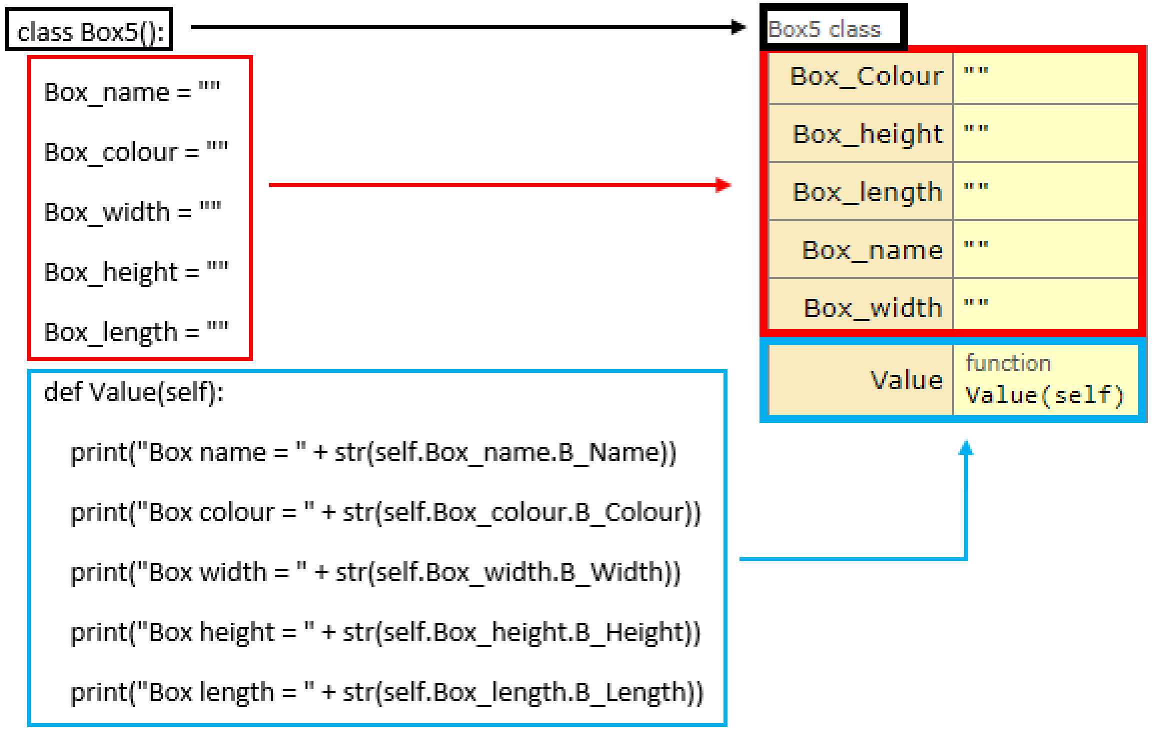
Height class	
Height	1

Length class	
Height	1

Width class	
Height	1

Link class to other class (creation)

- Class Box5()
- Red box
  - Class parametres/ instances
- Blue box
  - Print function



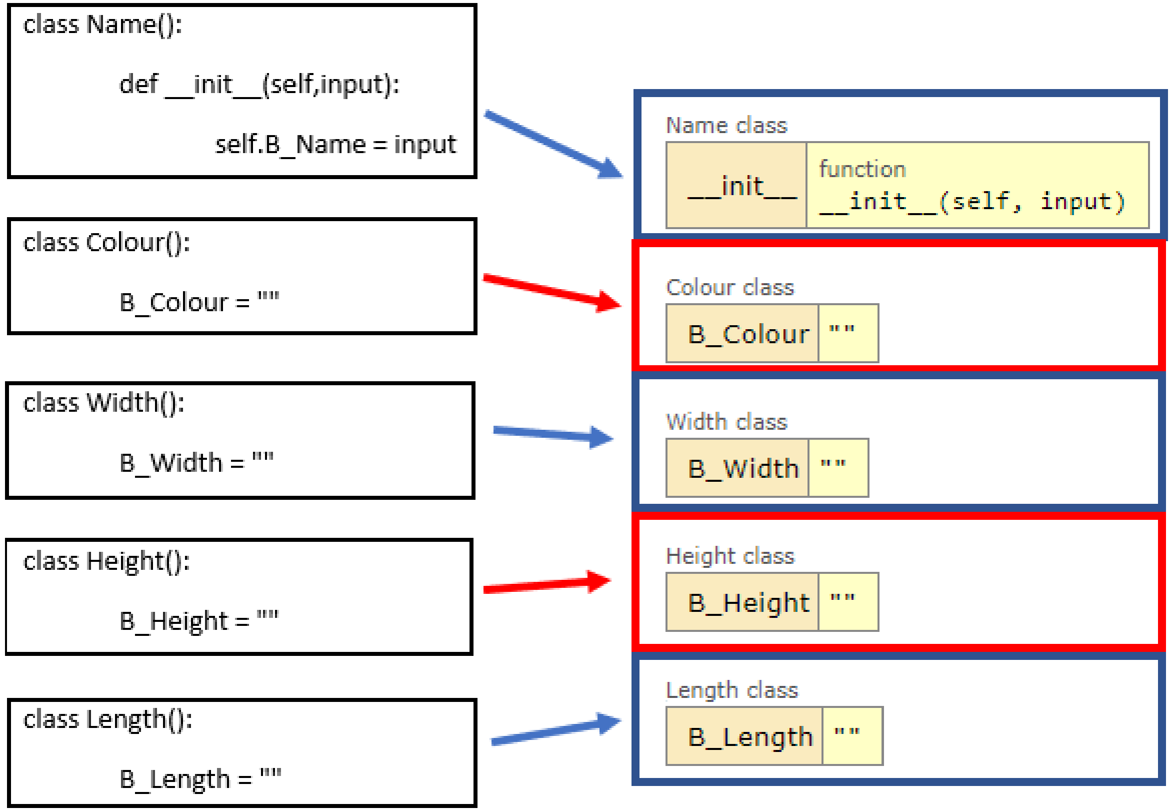
```
class Box5():
    Box_name = ""
    Box_colour = ""
    Box_width = ""
    Box_height = ""
    Box_length = ""
    def Value(self):
        print("Box name = " + str(self.Box_name.B_Name))
        print("Box colour = " + str(self.Box_colour.B_Colour))
        print("Box width = " + str(self.Box_width.B_Width))
        print("Box height = " + str(self.Box_height.B_Height))
        print("Box length = " + str(self.Box_length.B_Length))
```

Box5 class

Box_Colour	""
Box_height	""
Box_length	""
Box_name	""
Box_width	""
Value	function Value(self)

Link class to other class (creation cont.)

- Class Name()
- Class Colour()
- Class Width()
- Class Height()
- Class Length()
- Every classes contain null parametre value



```
class Name():
        def __init__(self,input):
                self.B_Name = input
```

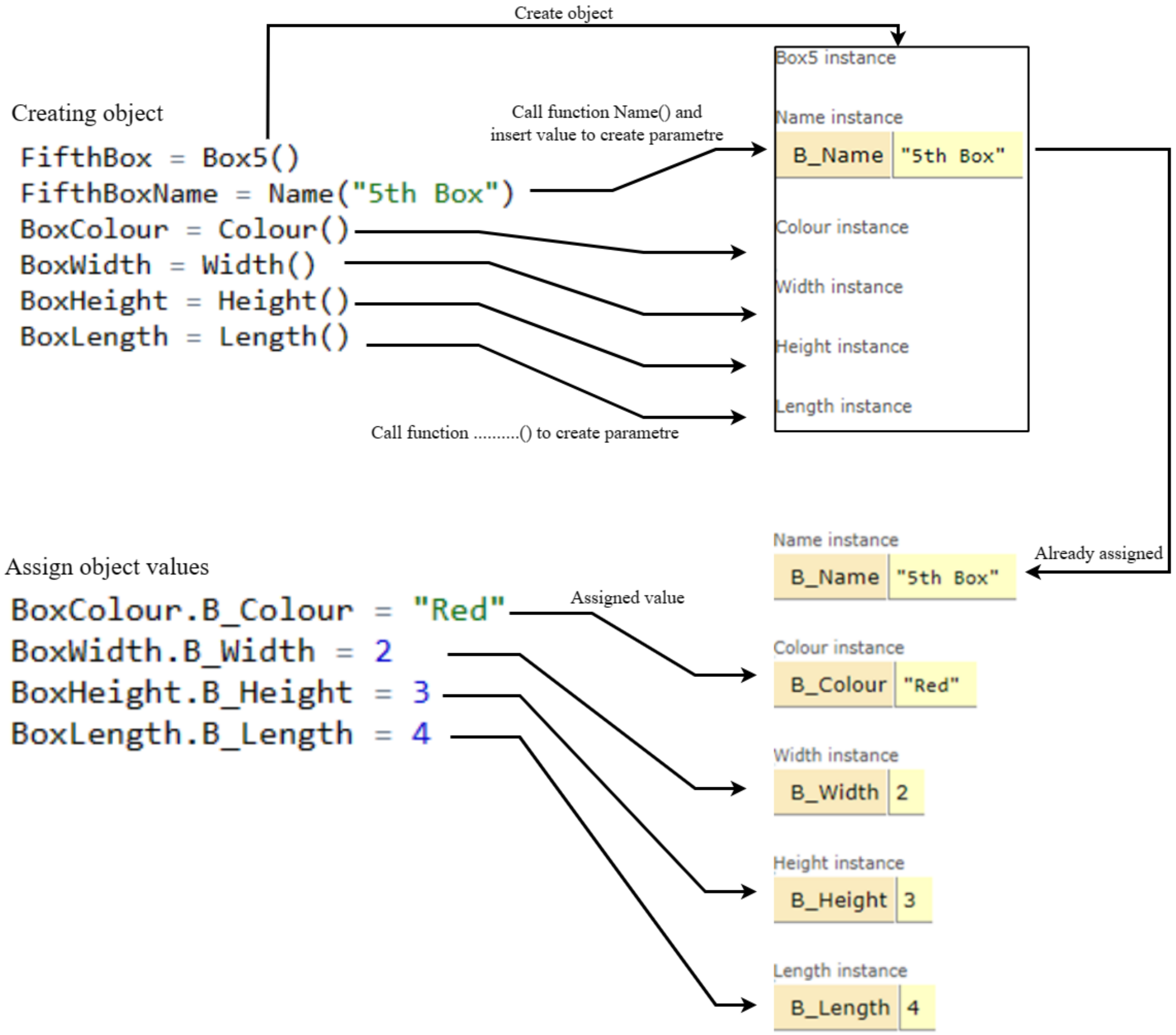
```
class Colour():
        B_Colour = ""
```

```
class Width():
        B_Width = ""
```

```
class Height():
        B_Height = ""
```

```
class Length():
        B_Length = ""
```

Link class to other class (creation object)



Create object

Creating object

```
FifthBox = Box5()
FifthBoxName = Name("5th Box")
BoxColour = Colour()
BoxWidth = Width()
BoxHeight = Height()
BoxLength = Length()
```

Call function Name() and insert value to create parametre

Call function ....() to create parametre

Box5 instance

Name instance

B_Name	"5th Box"

Colour instance

Width instance

Height instance

Length instance

Already assigned

Assign object values

```
BoxColour.B_Colour = "Red"
BoxWidth.B_Width = 2
BoxHeight.B_Height = 3
BoxLength.B_Length = 4
```

Assigned value

Name instance

B_Name	"5th Box"

Colour instance

B_Colour	"Red"

Width instance

B_Width	2

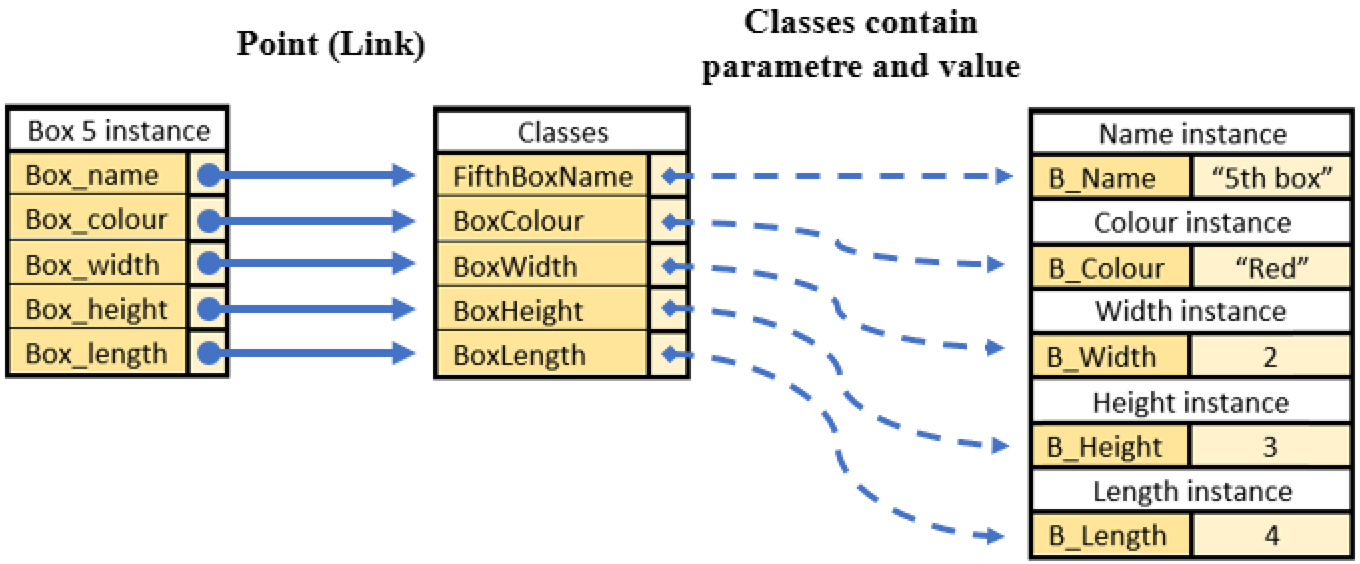
Height instance

B_Height	3

Length instance

B_Length	4

Link class to other class (Link)



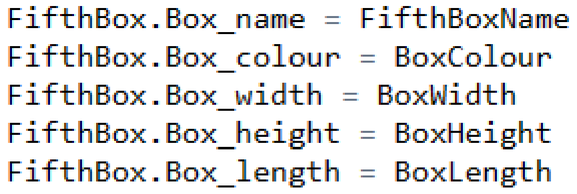
Point (Link)

Classes contain parametre and value

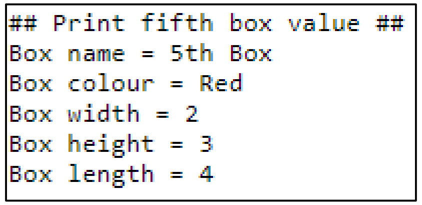
Box 5 instance
Box_name
Box_colour
Box_width
Box_height
Box_length

Classes
FifthBoxName
BoxColour
BoxWidth
BoxHeight
BoxLength

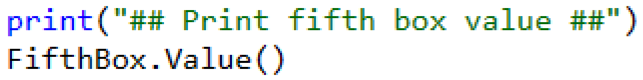
Name instance	
B_Name	"5th box"
Colour instance	
B_Colour	"Red"
Width instance	
B_Width	2
Height instance	
B_Height	3
Length instance	
B_Length	4



```
FifthBox.Box_name = FifthBoxName
FifthBox.Box_colour = BoxColour
FifthBox.Box_width = BoxWidth
FifthBox.Box_height = BoxHeight
FifthBox.Box_length = BoxLength
```



```
## Print fifth box value ##
Box name = 5th Box
Box colour = Red
Box width = 2
Box height = 3
Box length = 4
```

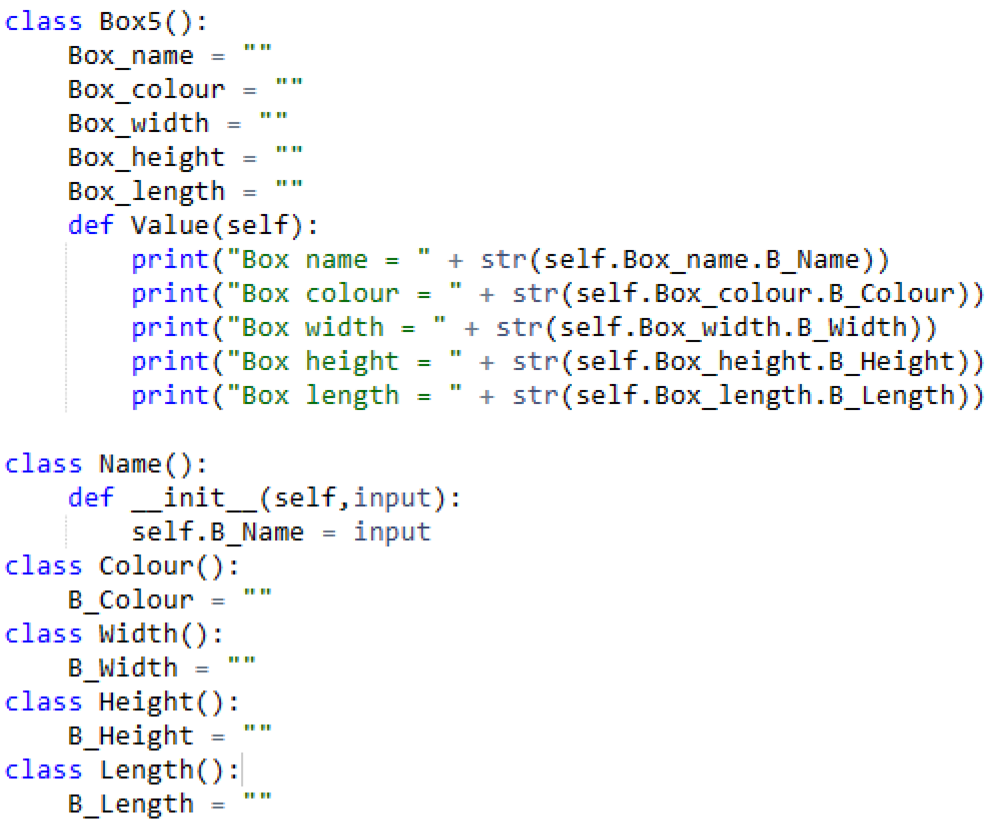


```
print("## Print fifth box value ##")
FifthBox.Value()
```

```
def Value(self):
    print("Box name = " + str(self.Box_name.B_Name))
    print("Box colour = " + str(self.Box_colour.B_Colour))
    print("Box width = " + str(self.Box_width.B_Width))
    print("Box height = " + str(self.Box_height.B_Height))
    print("Box length = " + str(self.Box_length.B_Length))
```

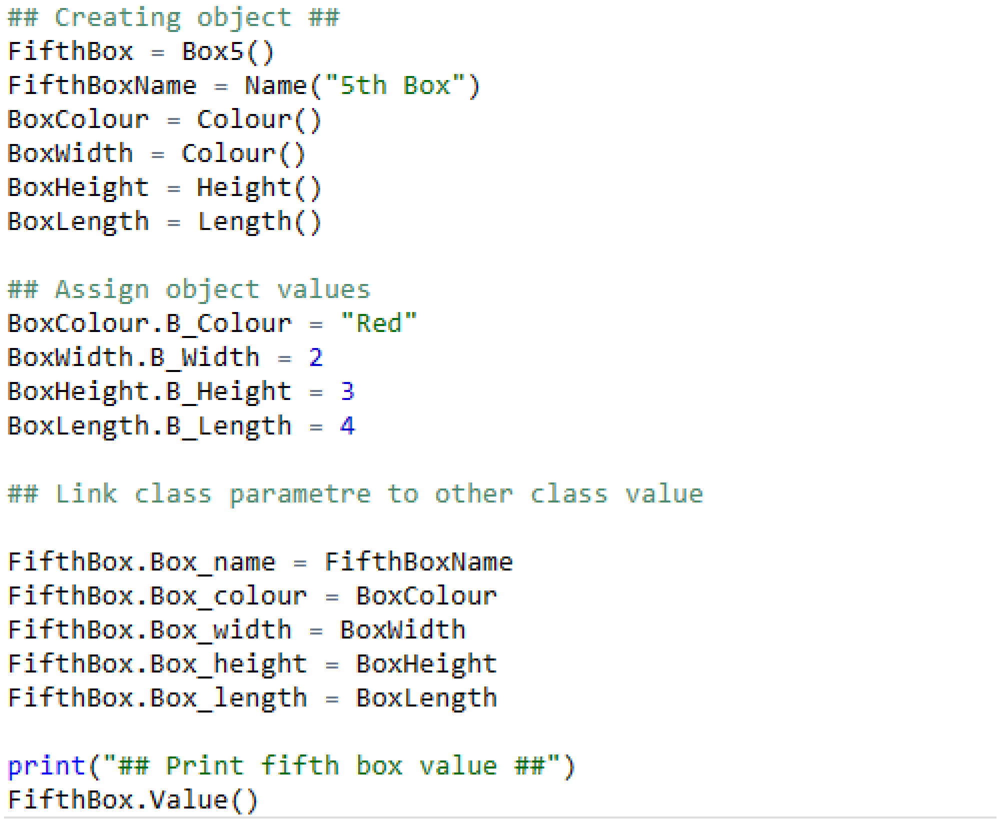
Link class to other class (All)

SCHOOL OF ENGINEERING BANGKOK UNIVERSITY



```python
class Box5():
    Box_name = ""
    Box_colour = ""
    Box_width = ""
    Box_height = ""
    Box_length = ""
    def Value(self):
        print("Box name = " + str(self.Box_name.B_Name))
        print("Box colour = " + str(self.Box_colour.B_Colour))
        print("Box width = " + str(self.Box_width.B_Width))
        print("Box height = " + str(self.Box_height.B_Height))
        print("Box length = " + str(self.Box_length.B_Length))

class Name():
    def __init__(self,input):
        self.B_Name = input
class Colour():
    B_Colour = ""
class Width():
    B_Width = ""
class Height():
    B_Height = ""
class Length():
    B_Length = ""
```

Classes creation

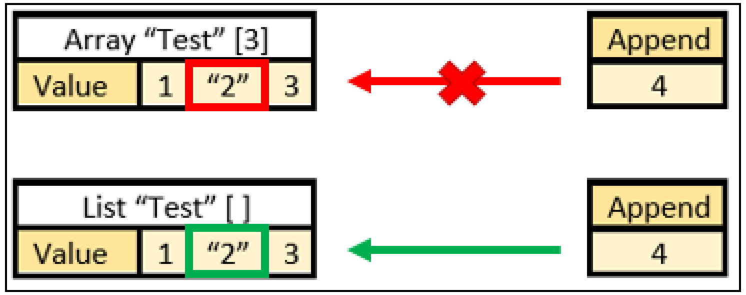


```python
## Creating object ##
FifthBox = Box5()
FifthBoxName = Name("5th Box")
BoxColour = Colour()
BoxWidth = Colour()
BoxHeight = Height()
BoxLength = Length()

## Assign object values
BoxColour.B_Colour = "Red"
BoxWidth.B_Width = 2
BoxHeight.B_Height = 3
BoxLength.B_Length = 4

## Link class parametre to other class value

FifthBox.Box_name = FifthBoxName
FifthBox.Box_colour = BoxColour
FifthBox.Box_width = BoxWidth
FifthBox.Box_height = BoxHeight
FifthBox.Box_length = BoxLength

print("## Print fifth box value ##")
FifthBox.Value()
```

Objects creation and other commands

Linked list (list)

- Normal array cannot append the last index further of declaration
- Array cannot mix different parametre types together



Array “Test” [3]

Value	1	“2”	3

Append

4

List “Test” [ ]

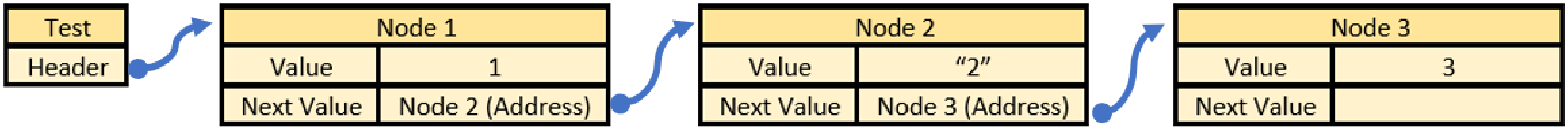
Value	1	“2”	3

Append

4

Linked list diagram (Singly linked lists)

Long format



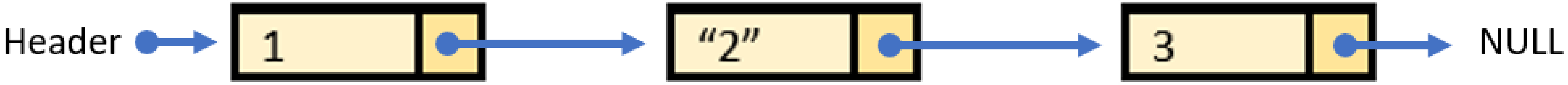
Test
Header

Node 1	
Value	1
Next Value	Node 2 (Address)

Node 2	
Value	“2”
Next Value	Node 3 (Address)

Node 3	
Value	3
Next Value	

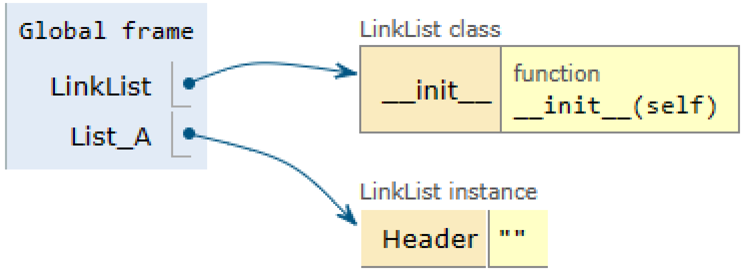
Short format

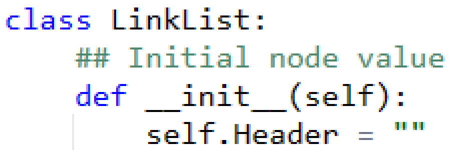


Header → | 1 | ● | → | “2” | ● | → | 3 | ● | → NULL

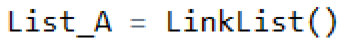
Linked list basic code structure

Create a list header



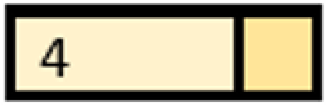


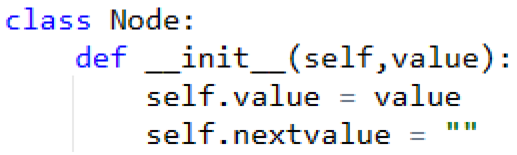
```
class LinkList:
    ## Initial node value
    def __init__(self):
        self.Header = ""
```



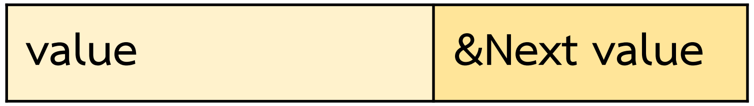
```
List_A = LinkList()
```

Create a new node as header





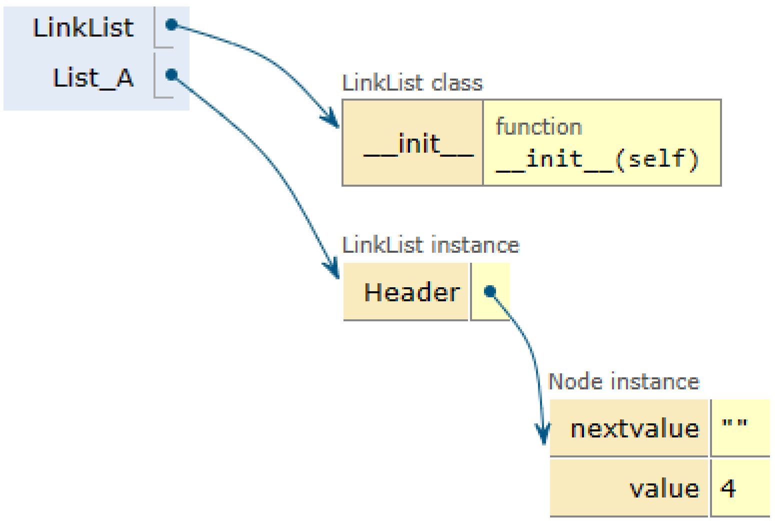
```
class Node:
    def __init__(self,value):
        self.value = value
        self.nextvalue = ""
```



Linked list diagram (append)

SCHOOL OF ENGINEERING BANGKOK UNIVERSITY

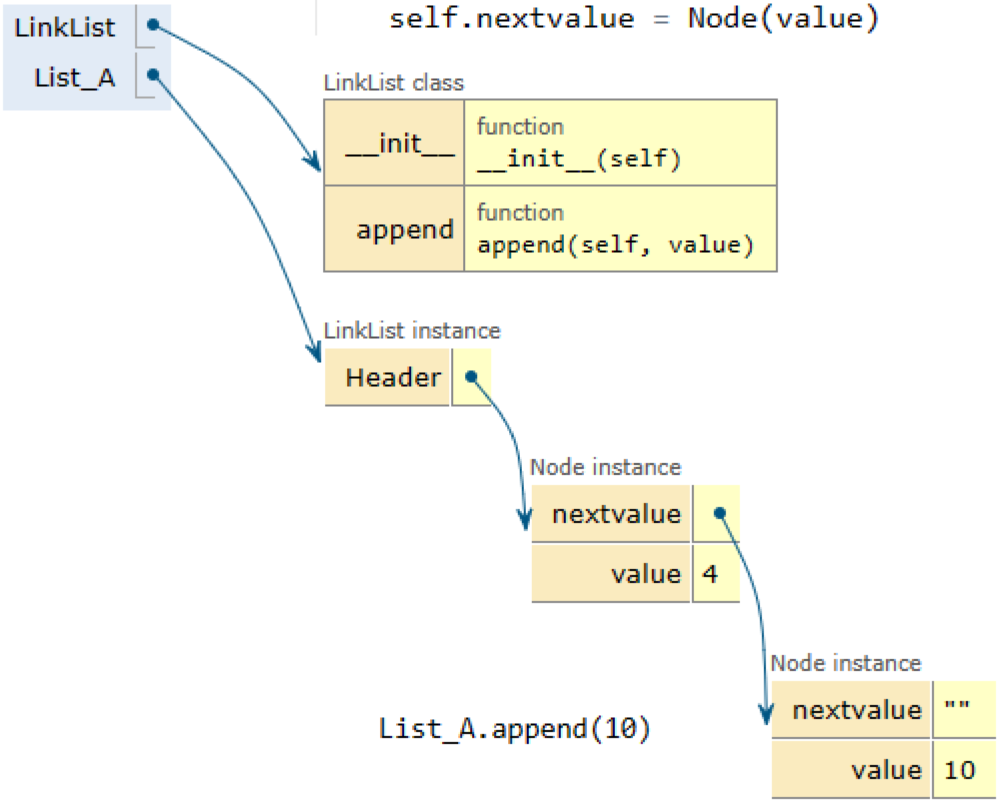
Linked header to a new node



```
List_A.Header = Node(4)
```

Append a new node (function in LinkList class)

```
def append(self,value):
    if self.Header != "":
        self = self.Header
    while self.nextvalue != "" :
        self = self.nextvalue
    self.nextvalue = Node(value)
```

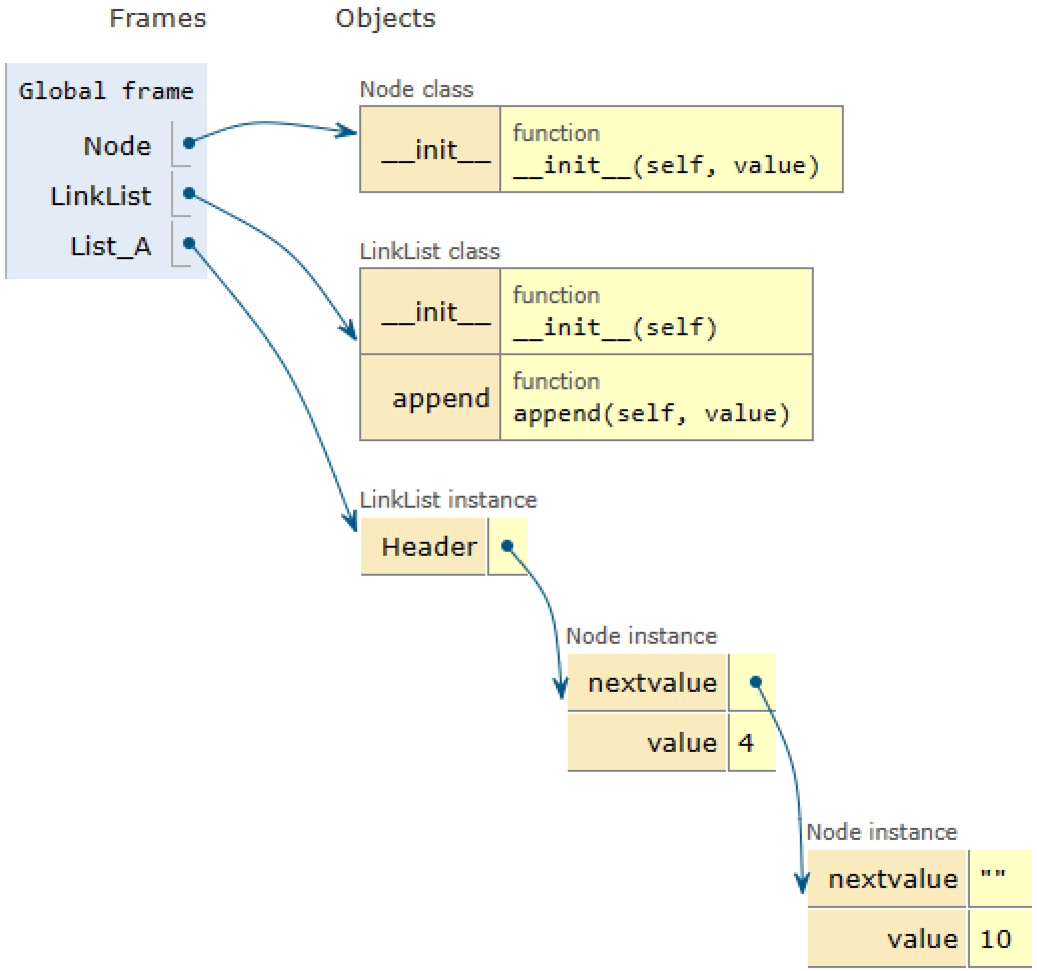


```
List_A.append(10)
```

Linked list diagram (append cont.)

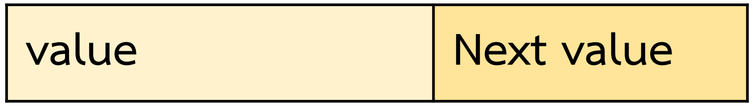
```
class Node:
    def __init__(self,value):
        self.value = value
        self.nextvalue = ""

class LinkList:
    ## Initial node value
    def __init__(self):
        self.Header = ""
    ## Append from last position
    def append(self,value):
        if self.Header != "":
            self = self.Header
        while self.nextvalue != "" :
            self = self.nextvalue
        self.nextvalue = Node(value)

List_A = LinkList()
List_A.Header = Node(4)
List_A.append(10)
```

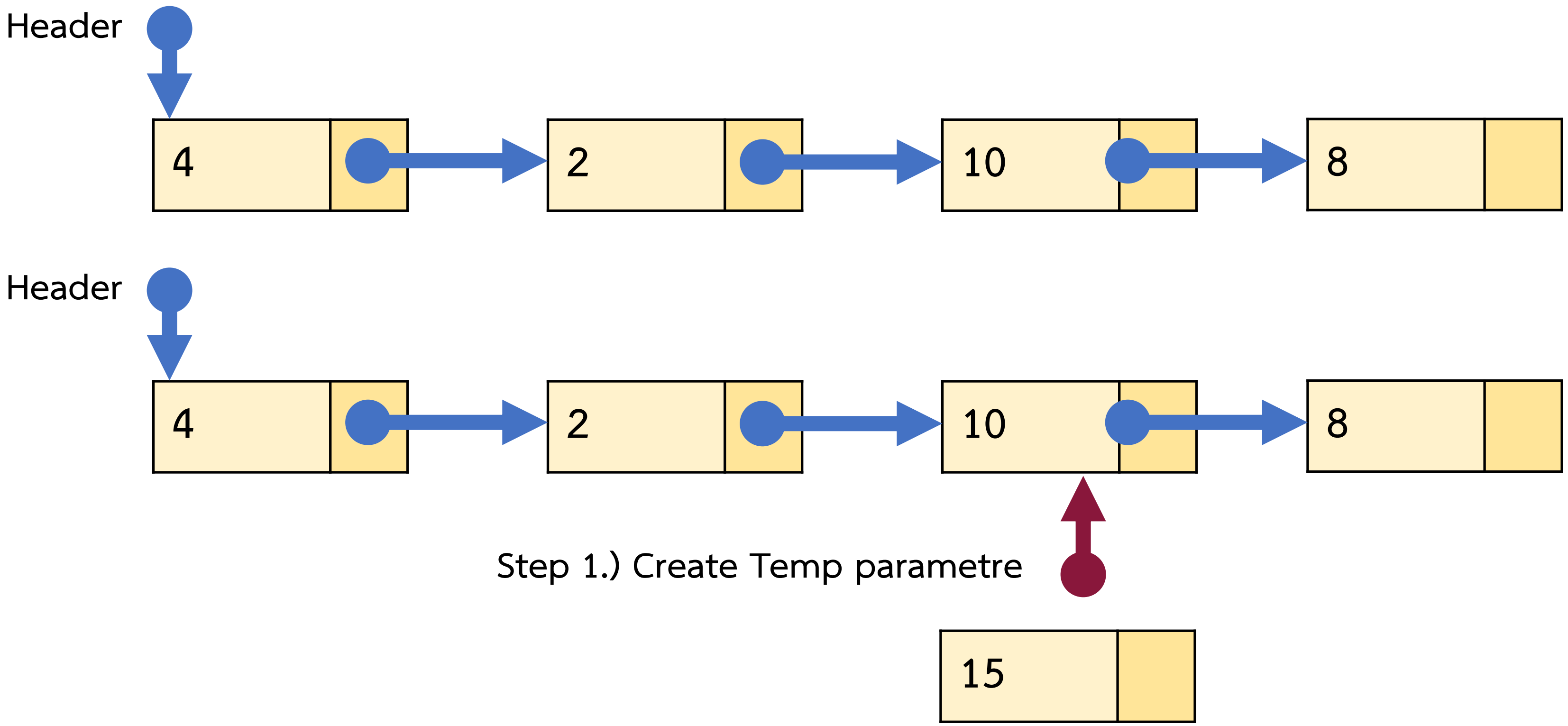
Objects creation and other commands



Linked list diagram (insert)

Need to insert a new nod 7 at 2nd position



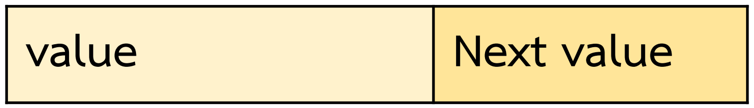


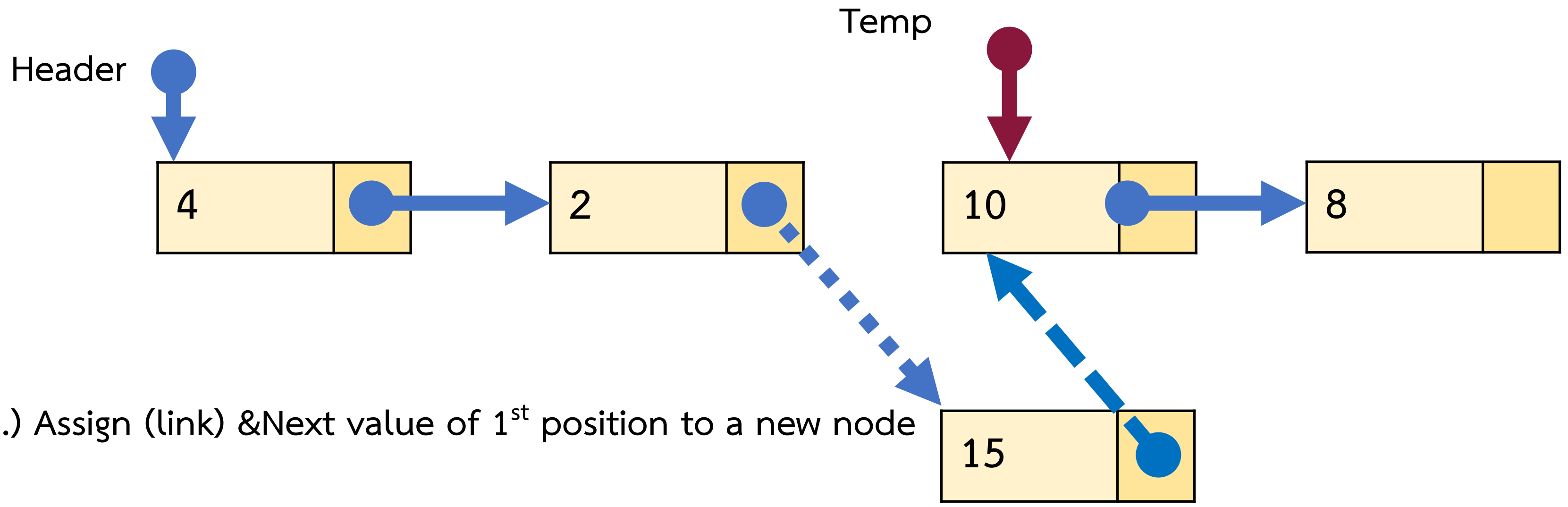
Step 1.) Create Temp parametre

Step 2.) Create a new node

Linked list diagram (insert cont.)

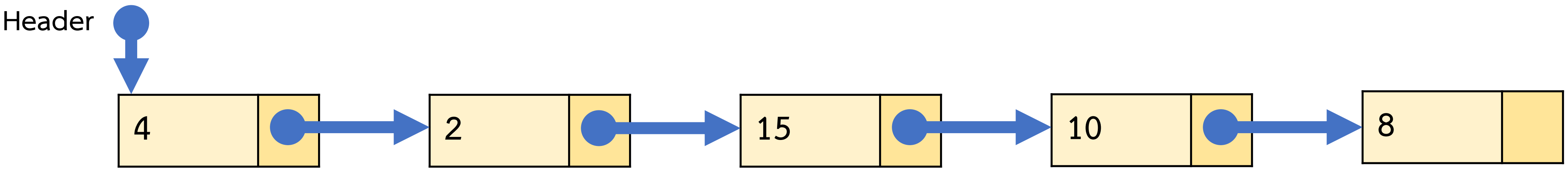
Need to insert a new nod 7 at 2nd position





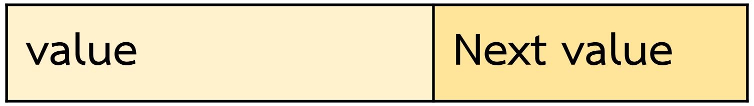
Step 3.) Assign (link) &Next value of 1st position to a new node

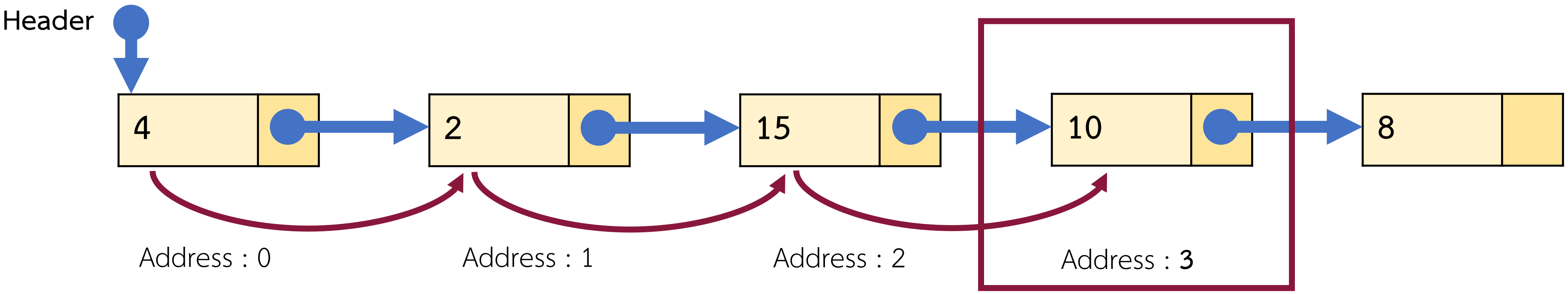
Step 4.) Assign (link) &Next value of new 2nd position to temp



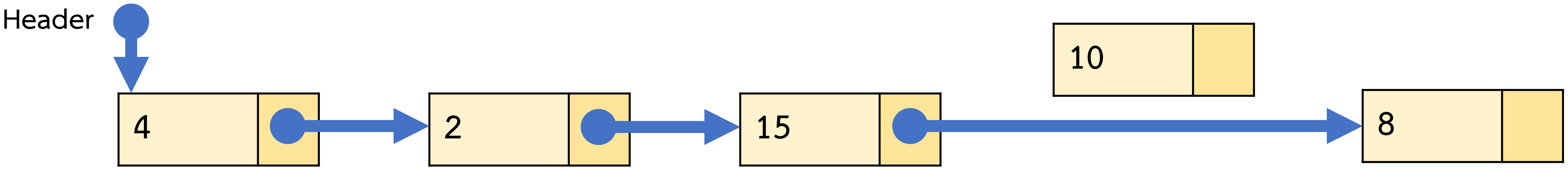
Linked list diagram (remove)

Need to remove a node at **4th** position





Step 1.) Find an exactly position (stop at Address 2 because next address is 3)

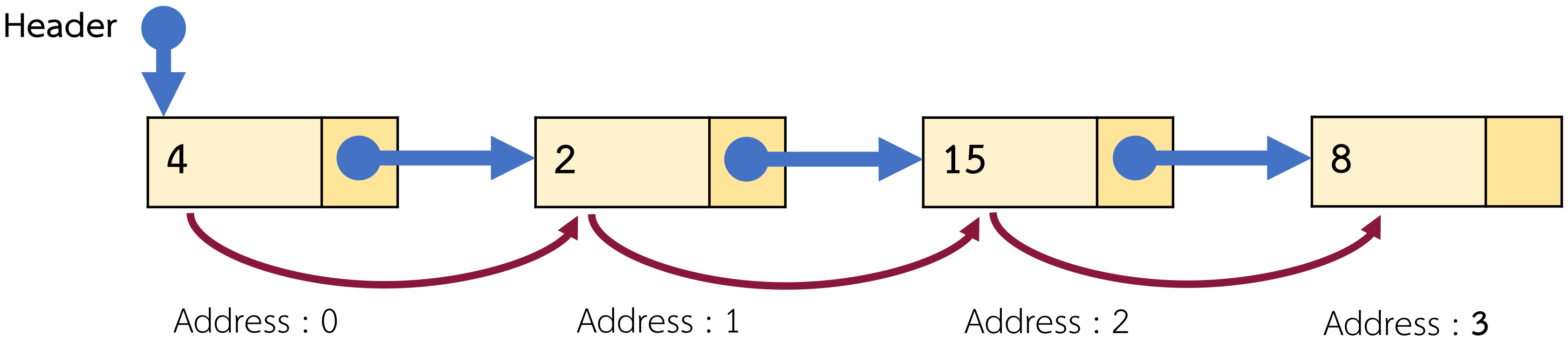


Step 2.) Assign (link) next value of Address 2 to Address 4 (Nextvalue = Nextvalue.Nextvalue)

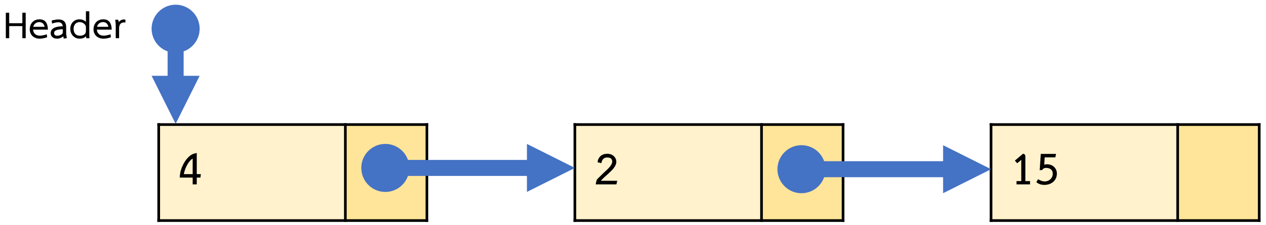
Linked list diagram (remove last index)

Need to remove a node at the **last** position

value	Next value



Step 1.) Find the node where Nextvalue is equal to “” (null). Which in this case, address is equal to 3

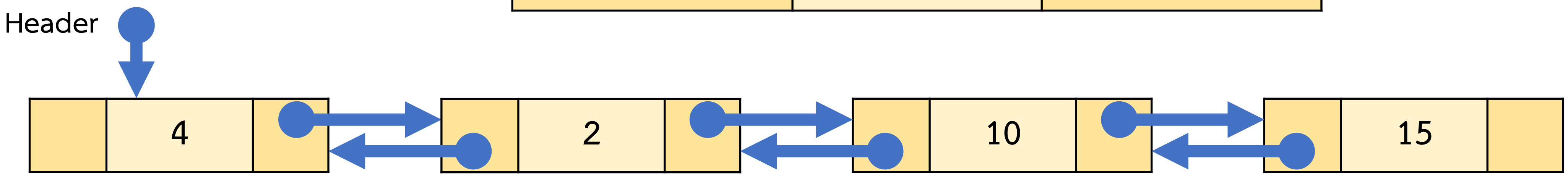


Step 2.) Assign (link) next value of Address 2 to “” (null)

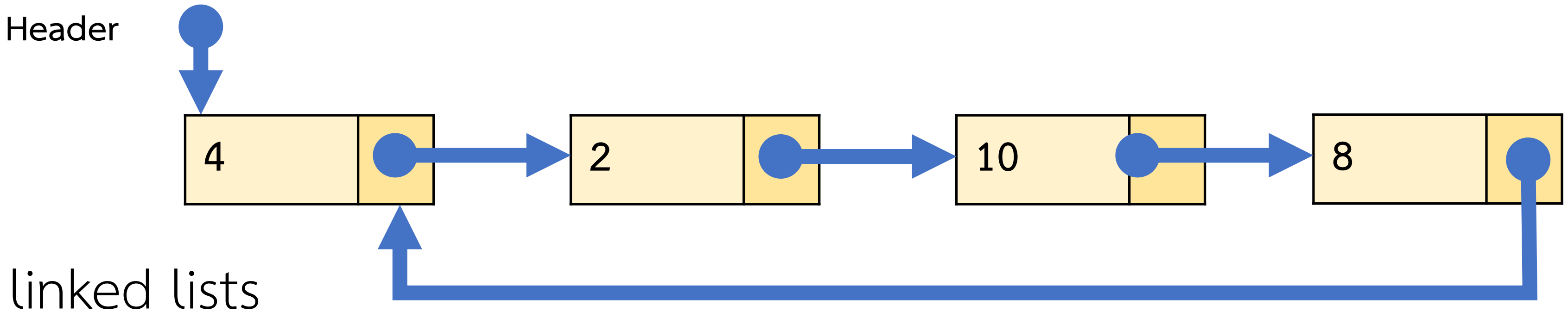
Linked list diagram (Other types)

Doubly linked lists

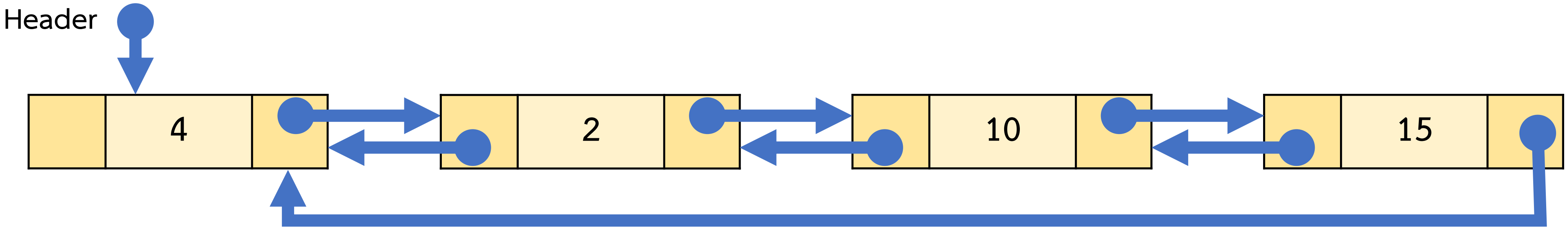
Previous value	value	Next value



Circular linked lists



Circular doubly linked lists

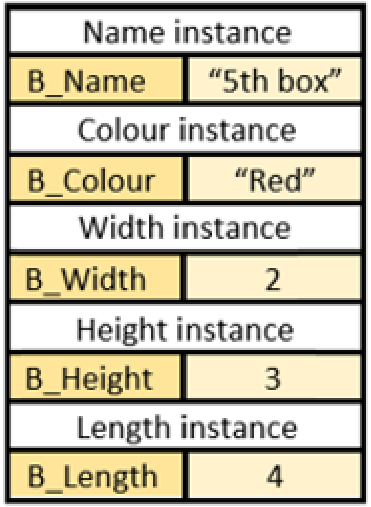


Exercise

- Class exercise
  - Create two cylinders by using class
    - First cylinder parametres
      - Radius = 5
      - Height = 10
    - Second cylinder parametres
      - Radius = 7
      - Height = 13
  - Calculate and print both measure of cylinders
  - Draw data structure of both cylinders (like this example)

$$volume\ of\ cylinder = \pi r^2 h$$

Object/ Data structure



Name instance	
B_Name	“5th box”
Colour instance	
B_Colour	“Red”
Width instance	
B_Width	2
Height instance	
B_Height	3
Length instance	
B_Length	4