

# LintCode 参考程序

Lionx

2016 年 4 月 19 日



# 目录

<b>第一章 入门 (Naive)</b>	<b>5</b>
1.1 Problem ID: 228 Middle of Linked List . . . . .	6
1.1.1 Description . . . . .	6
1.1.2 Example . . . . .	6
1.1.3 Code . . . . .	6
<b>第二章 容易 (Easy)</b>	<b>9</b>
2.1 Problem ID: 496 Toy Factory . . . . .	10
2.1.1 Description . . . . .	10
2.1.2 Example . . . . .	10
2.1.3 Code . . . . .	10
2.2 Problem ID: 497 Shape Factory . . . . .	12
2.2.1 Description . . . . .	12
2.2.2 Example . . . . .	12
2.2.3 Code . . . . .	12
<b>第三章 中等 (Medium)</b>	<b>15</b>
<b>第四章 困难 (Hard)</b>	<b>17</b>
<b>第五章 超难 (Super)</b>	<b>19</b>



# 第一章 入门 (Naive)

## 1.1 Problem ID: 228 Middle of Linked List

### 1.1.1 Description

Find the middle node of a linked list.

### 1.1.2 Example

Given 1->2->3, return the node with value 2.

Given 1->2, return the node with value 1.

### 1.1.3 Code

C++

```
1  /**
2   * Definition of ListNode
3   * class ListNode {
4   * public:
5   *     int val;
6   *     ListNode *next;
7   *     ListNode(int val) {
8   *         this->val = val;
9   *         this->next = NULL;
10  *     }
11  * }
12  */
13  class Solution{
14  public:
15      /**
16       * @param head: the head of linked list.
17       * @return: a middle node of the linked list
18       */
19      ListNode *middleNode(ListNode *head) {
20          // Write your code here
21          if(head == NULL){
22              return NULL;
23          }
24          ListNode *fast = head;
25          ListNode *slow = head;
26          while(fast->next != NULL && fast->next->next != NULL){
27              slow = slow->next;
28              fast = fast->next->next;
29          }
30          return slow;
31      }
32  };
```

**Python**

```
1  """
2  Definition of ListNode
3  class ListNode(object):
4
5      def __init__(self, val, next=None):
6          self.val = val
7          self.next = next
8  """
9
10 class Solution:
11     # @param head: the head of linked list.
12     # @return: a middle node of the linked list
13     def middleNode(self, head):
14         # Write your code here
15         if head is None:
16             return None
17         slow = head;
18         fast = head;
19         while fast.next is not None and fast.next.next is not None:
20             slow = slow.next
21             fast = fast.next.next
22         return slow
```





## 第二章 容易 (Easy)

## 2.1 Problem ID: 496 Toy Factory

### 2.1.1 Description

Factory is a design pattern in common usage. Please implement a ToyFactory which can generate proper toy based on the given type.

### 2.1.2 Example

```
1 ToyFactory tf = ToyFactory();
2 Toy toy = tf.getToy('Dog');
3 toy.talk();
4 >> Wow
5
6 toy = tf.getToy('Cat');
7 toy.talk();
8 >> Meow
```

### 2.1.3 Code

C++

```
1 /**
2  * Your object will be instantiated and called as such:
3  * ToyFactory* tf = new ToyFactory();
4  * Toy* toy = tf->getToy(type);
5  * toy->talk();
6  */
7 class Toy {
8 public:
9     virtual void talk() const=0;
10 };
11
12 class Dog: public Toy {
13     // Write your code here
14     void talk() const{
15         cout << "Wow" << endl;
16     }
17 };
18
19 class Cat: public Toy {
20     // Write your code here
21     void talk() const{
22         cout << "Meow" << endl;
23     }
24 };
25
26 class ToyFactory {
```

```

27 public:
28     /**
29      * @param type a string
30      * @return Get object of the type
31      */
32     Toy* getToy(string& type) {
33         // Write your code here
34         if(type == "Dog"){
35             return new Dog();
36         }
37         if(type == "Cat"){
38             return new Cat();
39         }
40         return NULL;
41     }
42 };

```

## Python

```

1  """
2  Your object will be instantiated and called as such:
3  ty = ToyFactory()
4  toy = ty.getToy(type)
5  toy.talk()
6  """
7  class Toy:
8      def talk(self):
9          raise NotImplementedError( 'This method should have implemented.' )
10
11  class Dog(Toy):
12      # Write your code here
13      def talk(self):
14          print "Wow"
15
16  class Cat(Toy):
17      # Write your code here
18      def talk(self):
19          print "Meow"
20
21
22  class ToyFactory:
23      # @param {string} shapeType a string
24      # @return {Toy} Get object of the type
25      def getToy(self, type):
26          # Write your code here
27          if type == "Dog":
28              return Dog()
29          if type == "Cat":
30              return Cat()
31          return None

```

## 2.2 Problem ID: 497 Shape Factory

### 2.2.1 Description

Factory is design pattern in common usage. Implement a ShapeFactory that can generate correct shape.

### 2.2.2 Example

```

1 ShapeFactory sf = new ShapeFactory();
2 Shape shape = sf.getShape("Square");
3 shape.draw();
4 >>  ——
5 >> |    |
6 >> |    |
7 >>  ——
8
9 shape = sf.getShape("Triangle");
10 shape.draw();
11 >>  /\
12 >> /  \
13 >> /____\
14
15 shape = sf.getShape("Rectangle");
16 shape.draw();
17 >>  ——
18 >> |    |
19 >>  ——

```

### 2.2.3 Code

C++

```

1 /**
2  * Your object will be instantiated and called as such:
3  * ShapeFactory* sf = new ShapeFactory();
4  * Shape* shape = sf->getShape(shapeType);
5  * shape->draw();
6  */
7 class Shape {
8 public:
9     virtual void draw() const=0;
10 };
11
12 class Rectangle: public Shape {
13     // Write your code here
14     void draw() const{
15         cout << "┌───" << endl << "│uuu│" << endl << "└───" << endl;

```

```

16     }
17 };
18
19 class Square: public Shape {
20     // Write your code here
21     void draw() const{
22         cout << "┐───" << endl << "│uuu│" << endl
23         << "│uuu│" << endl << "└───" << endl;
24     }
25 };
26
27 class Triangle: public Shape {
28     // Write your code here
29     void draw() const{
30         cout << "uu/\\" << endl << "u/uu\\" << endl << "/____\\" << endl;
31     }
32 };
33
34 class ShapeFactory {
35 public:
36     /**
37      * @param shapeType a string
38      * @return Get object of type Shape
39      */
40     Shape* getShape(string& shapeType) {
41         // Write your code here
42         if(shapeType == "Square"){
43             return new Square();
44         }
45         if(shapeType == "Rectangle"){
46             return new Rectangle();
47         }
48         if(shapeType == "Triangle"){
49             return new Triangle();
50         }
51         return NULL;
52     }
53 };

```

## Python

```

1  """ """
2  Your object will be instantiated and called as such:
3  sf = ShapeFactory()
4  shape = sf.getShape(shapeType)
5  shape.draw()
6  """ """
7  class Shape:
8      def draw(self):
9          raise NotImplementedError( 'This method should have implemented.' )
10
11 class Triangle(Shape):

```

```
12     # Write your code here.
13     def draw(self):
14         print "uu/\\"
15         print "u/uu\\"
16         print "/____\\"
17
18     class Rectangle(Shape):
19         # Write your code here
20         def draw(self):
21             print "┌───"
22             print "|uuu|"
23             print "└───"
24
25     class Square(Shape):
26         # Write your code here
27         def draw(self):
28             print "┌───"
29             print "|uuu|"
30             print "|uuu|"
31             print "└───"
32
33     class ShapeFactory:
34         # @param {string} shapeType a string
35         # @return {Shape} Get object of type Shape
36         def getShape(self, shapeType):
37             # Write your code here
38             if shapeType == "Square":
39                 return Square()
40             if shapeType == "Triangle":
41                 return Triangle()
42             if shapeType == "Rectangle":
43                 return Rectangle()
44             return None
```

## 第三章 中等 (Medium)





## 第四章 困难（Hard）



## 第五章 超难 (Super)