



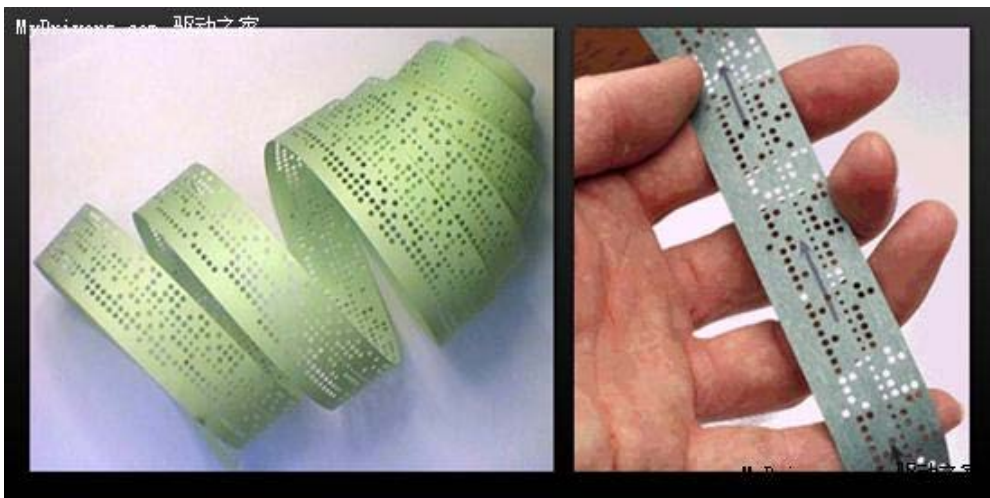
# 数据结构与算法（Python版）

## 栈的应用：十进制转换为二进制

陈斌 北京大学 gisichen@pku.edu.cn

# 十进制转换为二进制

- ❖ 二进制是计算机原理中最基本的概念，作为组成计算机最基本部件的逻辑门电路，其输入和输出均仅为两种状态：0和1
- ❖ 但十进制是人类传统文化中最基本的数值概念，如果没有进制之间的转换，人们跟计算机的交互会相当的困难



# 十进制转换为二进制

❖ 所谓的“进制”，就是用多少个字符来表示整数

十进制是0~9这十个数字字符，二进制是0、1两个字符

❖ 我们经常需要将整数在二进制和十进制之间转换

如：(233)<sub>10</sub>的对应二进制数为(11101001)<sub>2</sub>，具体是这样：

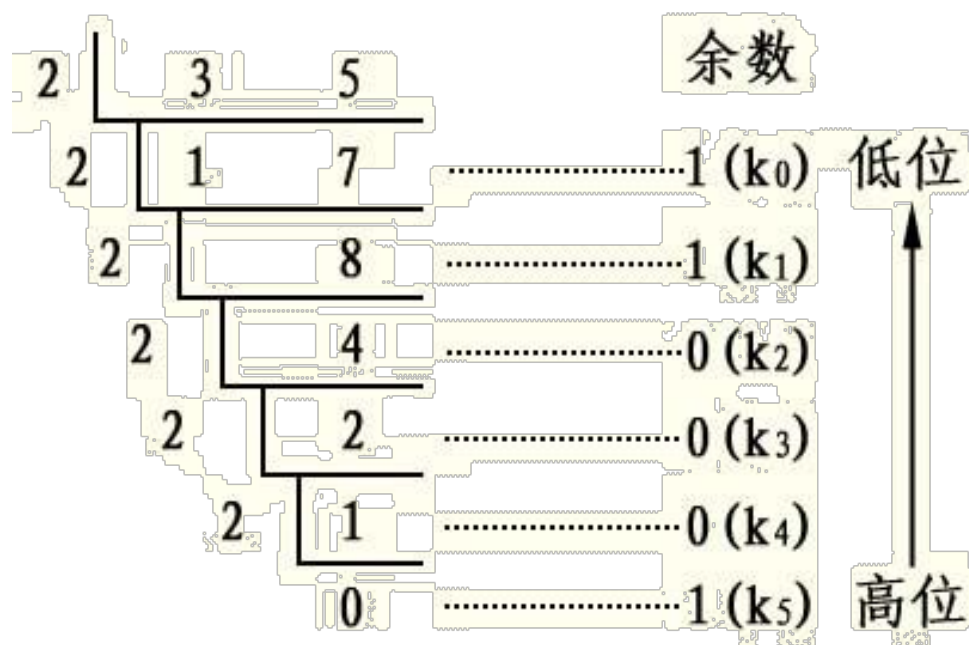
$$(233)_{10} = 2 \times 10^2 + 3 \times 10^1 + 3 \times 10^0$$

$$(11101001)_2 = 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

# 十进制转换为二进制

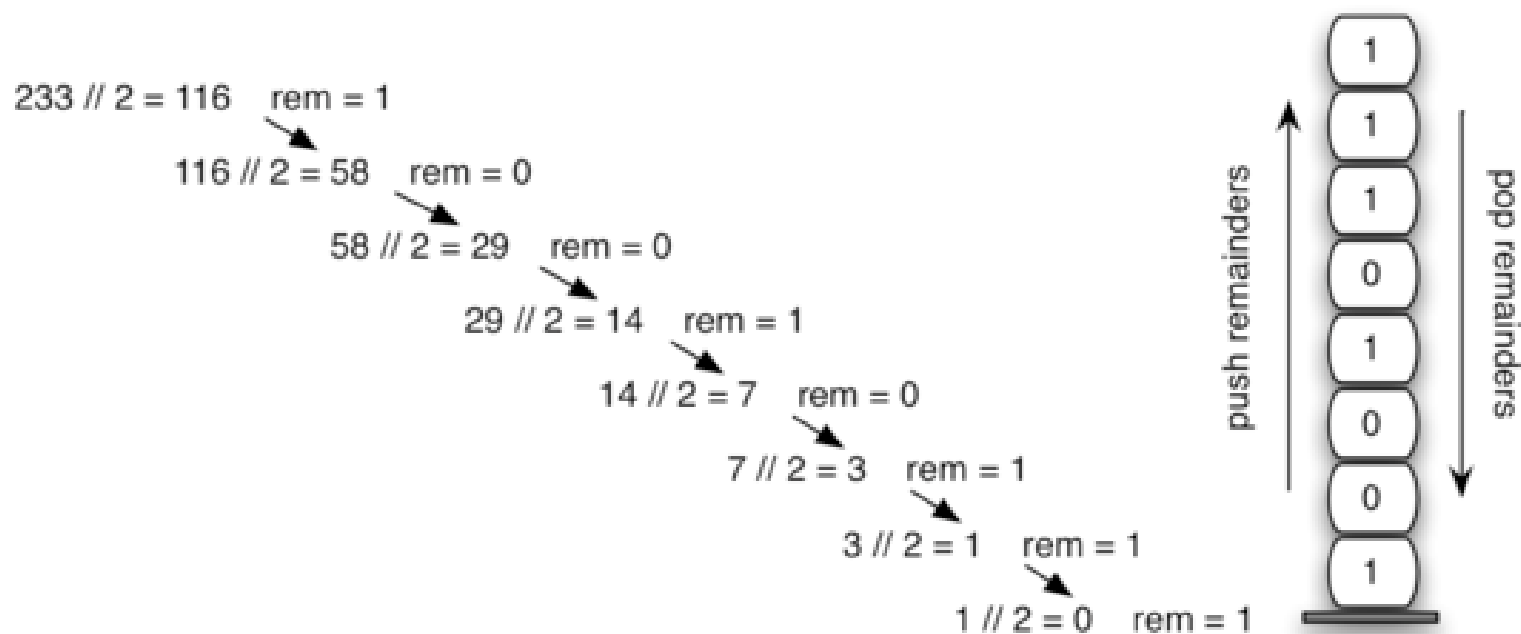
## ❖ 十进制转换为二进制，采用的是“除2求余数”的算法

将整数不断除以2，每次得到的余数就是由低到高的二进制位



# 十进制转换为二进制

- ❖ “除以2”的过程，得到的余数是从低到高的次序，而输出则是从高到低，所以需要有一个栈来反转次序



# 十进制转换为二进制：代码

```
from pythonds.basic.stack import Stack
```

```
def divideBy2(decNumber):  
    remstack = Stack()
```

```
    while decNumber > 0:
```

```
        rem = decNumber % 2
```

求余数

```
        remstack.push(rem)
```

```
        decNumber = decNumber // 2
```

整数除

```
    binString = ""
```

```
    while not remstack.isEmpty():
```

```
        binString = binString + str(remstack.pop())
```

```
    return binString
```

```
print(divideBy2(42))
```



# 扩展到更多进制转换

❖ 十进制转换为二进制的算法，很容易可以扩展为**转换到任意N进制**

只需要将“除以2求余数”算法改为“除以N求余数”算法就可以

❖ 计算机中另外两种常用的进制是八进制和十六进制

$(233)_{10}$  等于  $(351)_8$  和  $(E9)_{16}$

$(351)_8 = 3 \times 8^2 + 5 \times 8^1 + 1 \times 8^0$

$(E9)_{16} = 14 \times 16^1 + 9 \times 16^0$

# 扩展到更多进制转换

## ❖ 主要的问题是如何表示八进制及十六进制

二进制有两个不同数字0、1

十进制有十个不同数字0、1、2、3、4、5、6、7、8、9

八进制可用八个不同数字0、1、2、3、4、5、6、7

十六进制的十六个不同数字则是0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F



# 十进制转换为十六以下任意进制：代码

```
from pythonds.basic.stack import Stack
```

```
def baseConverter(decNumber, base):
```

```
    digits = "0123456789ABCDEF"
```

```
    remstack = Stack()
```

```
    while decNumber > 0:
```

```
        rem = decNumber % base
```

```
        remstack.push(rem)
```

```
        decNumber = decNumber // base
```

```
    newString = ""
```

```
    while not remstack.isEmpty():
```

```
        newString = newString + digits[remstack.pop()]
```

```
    return newString
```

```
print(baseConverter(25,2))
```

```
print(baseConverter(25,16))
```

