第 2 章 进制转换

2.1 N 进制和 10 进制互转

一、进制的基础知识

1、什么是进制？

进制也就是进位计数制，是人为定义的带进位的计数方法。对于任何一种进制 X 进制，就表示每一位置上的数运算时都是逢 X 进一位。十进制是逢十进一，十六进制是逢十六进一，二进制就是逢二进一，以此类推，x 进制就是逢 x 进位。

2、生活中常见的进制有哪些？

10 进制、60 进制、12 进制、24 进制等；

3、n 进制如何数数？

10 进制：0 1 2 3 4 5 6 7 8 9 10 11……

2 进制：0 1 10 11 100 101 110 111 1000 1001 1010 1011 1100 1101 1110 1111 10000……

8 进制：0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17 20 21……

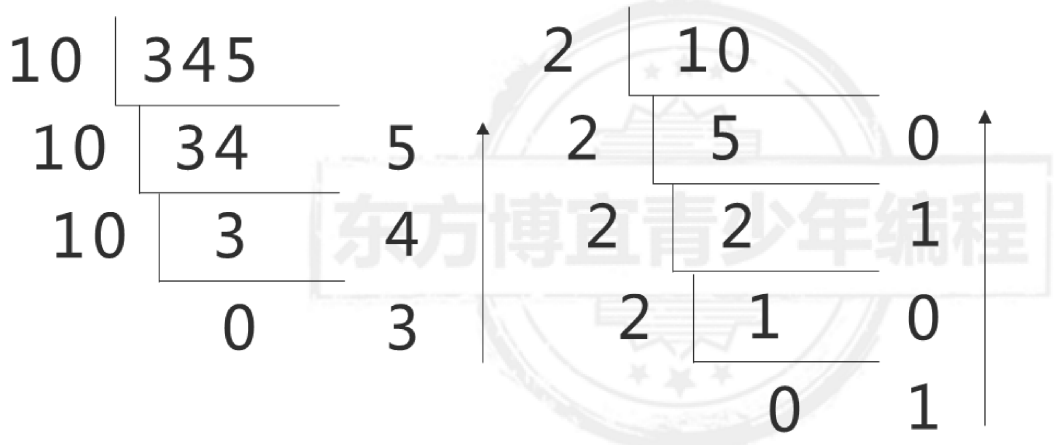
16 进制：0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20……

二、十进制转换为 R 进制

1、十进制整数转换成 R 进制的整数：除 R 取余法 。

2、学习技巧：可以参照短除法，短除法的作用是用来拆出 10 进制 n 的每一位，可以理解为 10 进制转 10 进制，使用的方法是除 10 取余，然后倒过来；

那么 10 进制转 R 进制，自然就是除 R 取余，然后倒过来；



3、课堂练习

$(12)_{10} = (\_\_\_\_\_)_2$

$(126)_{10} = (\_\_\_\_\_)_8$

$(258)_{10} = (\_\_\_\_\_)_{16}$

$(\ 431\ )_{10} = (\qquad\qquad)_{16}$

三、R 进制转换为十进制

1、R 进制转 10 进制整数：___按权展开___。

按权展开：基数为 N 的数字，只要将各位数字与它的权相乘，其积相加,和数就是十进制数。

2、学习技巧：___可以参照 10 进制整数计算机制来学习___；

$12345=5*1+4*10+3*100+2*1000+1*10000$

$=5*10^0+4*10^1+3*10^2+2*10^3+1*10^4$

3、课堂练习

R 进制转 10 进制实例：

$1100_2 = 0\times2^0+0\times2^1+1\times2^2+1\times2^3 = (12)_{10}$

$3506_8 = 6\times8^0+0\times8^1+5\times8^2+3\times8^3 = (1862)_{10}$

$1A_{16} = 10*16^0 + 1*16^1 = (26)_{10}$

4、作业练习

$(110011)_2 = (\qquad\qquad)_{10}$

$(267)_8 = (\qquad\qquad)_{10}$

$(2AF)_{16} = (\qquad\qquad)_{10}$

四、10 进制和 R 进制互转程序实现

1108：【入门】正整数 N 转换成一个二进制数

定义字符串存储 N 转换的二进制数

0	1	2	3	4	5	6	7	8	9	10	11	

用短除法除 2 取余，将余数逆序存入字符串 s。

```
#include <bits/stdc++.h>
using namespace std;

string s;//s = "1010"
int n,x;
char c;
int main(){
	cin>>n;
	while(n != 0){
		x = n % 2;
		//x:0,1
		c = x + '0';
		//结果逆序连接为字符串
		s = c + s;
		n = n / 2;
	}

	if(s == ""){
		cout<<0;
	} else{
```

```
        cout<<s;
    }
}
```

1290：【入门】二进制转换十进制

1	1	0	1									
0	1	2	3	4	5	6	7	8	9	10	11	

思路：从最低位开始(s.size()-1)，倒过来计算（按权展开）

s[i] – '0'

准备变量 t 表示 2 的 n 次方，t = 1 每循环一次，t = t * 2

```
#include <bits/stdc++.h>
using namespace std;

string s;//存放二进制
int r,t = 1,i;//t:表示权重
int main(){
    cin>>s;
    for(i = s.size() - 1;i >= 0;i--){
        r = r + (s[i] - '0') * t;
        t = t * 2;
    }

    cout<<r;
}
```

1289：【入门】正整数 n 转换为 16 进制

0	1	2	3	4	5	6	7	8	9	10	11	

思路：除 16 取余！

逆序存储到字符串时要注意：

整数 0~9，转换为字符'0'~'9'，x + '0'

整数 10~15，转换为字符'A'~'F'，x + 'A' - 10

解法一：分别判断 n%16 结果在 0~9 及 10~15 的哪个范围，分别转换为对应的字符

```
#include <bits/stdc++.h>
using namespace std;

/*
  n 是一个不超过 18 位的正整数
*/
long long n,x;
string s;
char c;
int main(){
    cin>>n;
    while(n != 0){
        x = n % 16;
        //cout<<x<<endl;
        //将 x 转换为字符逆序存入字符串 s
        //x:0~9 -> '0'~'9'
        //x:10~15 -> 'A'~'F'
        if(x < 10){
            c = x + '0';
        }else{
            c = x + 'A' - 10;
```

```
        }

        s = c + s;
        n = n / 16;
    }

    if(s == ""){
        cout<<0;
    }else{
        cout<<s;
    }

}
```

解法二：用字符串存储十六进制对应的字符，简化 16 进制转为字符的过程

```
#include <bits/stdc++.h>
using namespace std;

/*
  n 是一个不超过 18 位的正整数
*/
long long n,x;
string s;
string t = "0123456789ABCDEF";
int main(){
    cin>>n;
    while(n != 0){
        x = n % 16;
        //cout<<x<<endl;
        //将 x 转换为字符逆序存入字符串 s
        //x:0~9 -> '0'~'9'
        //x:10~15 -> 'A'~'F'
        //将 n%16 转换为字符逆序存入 s
        s = t[x] + s;
        n = n / 16;
    }

    if(s == ""){
        cout<<0;
    }else{
        cout<<s;
    }

}
```

注意：

int 最多表达到 2^31-1，10 位整数；

long long 最多表达到 2^63-1，19 位整数；

1292：【入门】十六进制转十进制

2	E	C	F									
0	1	2	3	4	5	6	7	8	9	10	11	

思路：逆序计算，按权展开！

从 s 中获取每一位 s[i]是字符，要转换为实际的整数！

s[i]：'0'~'9'，s[i] – '0'

s[i]：'A'~'F'，s[i] – 'A' + 10

```
#include <bits/stdc++.h>
using namespace std;
```

```
string s;
//t：表示权重，也就是 16 的 i 次方
long long r,t = 1,i;

int main(){
    cin>>s;
    //逆序计算，按权展开
    for(i = s.size() - 1;i >= 0;i--){
        //如果 s[i]是'0'~'9'
        if(isdigit(s[i])){
            r = r + (s[i] - '0') * t;
        }else{
            r = r + (s[i] - 'A' + 10) * t;
        }

        t = t * 16;
    }

    cout<<r;
}
```

1386：【基础】小丽找半个回文数？

a 数组存储 417 对应的各个位

7	1	4										
0	1	2	3	4	5	6	7	8	9	10	11	

b 数组存储 417 对应 16 进制的各个位

1	10	1										
0	1	2	3	4	5	6	7	8	9	10	11	

c 数组存储 417 对应 2 进制的各个位

1	2	3	2	1								
0	1	2	3	4	5	6	7	8	9	10	11	

经过计算发现，将 n 拆出每一位装数组（除 10 对 10 取余）
和将 n 转为 16 进制（除 16 对 16 取余，将余数装数组）
和将 n 转为 2 进制（除 2 对 2 取余，将余数装数组）

```
for(i = 0;i < k / 2;i++){
    0->k-1
    1->k-2
    i->k-i-1
}
```

解法一：

```
#include <bits/stdc++.h>
using namespace std;
/*
  如果这个数在 10 进制下不是回文数
  但这个数在 2 进制或者 16 进制下是回文数
*/

/*
  判断整数 n 在 d 进制下是否是回文
  除 d 对 d 取余数，将余数存入数组，判断数组是否是回文
```

```
*/
bool huiwen(int n,int d){
	bool r = true;//假设是回文
	int a[1000] = {0};//初始化为 0，存 n 转 d 进制后的每一位
	int k = 0;
	while(n != 0){
		a[k] = n % d;
		k++;
		n = n / d;
	}
	//判断回文：循环数组长度一半，判断对称位置是否有不等
	for(int i = 0;i < k / 2;i++){
		if(a[i] != a[k - i - 1]){
			r = false;
			break;
		}
	}

	return r;
}


int main(){
	int a[110],n,i;
	cin>>n;
	for(i = 0;i < n;i++){
		cin>>a[i];
	}

	//遍历每个数，判断是否是半个回文
	for(i = 0;i < n;i++){
		if(huiwen(a[i],10) == false && (huiwen(a[i],2) == true || huiwen(a[i],16) == true)){
			cout<<a[i]<<endl;
		}
	}
}
```

解法二：每读入一个数，就判断其是否是半个回文

```
#include <bits/stdc++.h>
using namespace std;
/*
  如果这个数在 10 进制下不是回文数
  但这个数在 2 进制或者 16 进制下是回文数
*/

/*
  判断整数 n 在 d 进制下是否是回文
  除 d 对 d 取余数，将余数存入数组，判断数组是否是回文
*/
bool huiwen(int n,int d){
	bool r = true;//假设是回文
	int a[1000] = {0};//初始化为 0，存 n 转 d 进制后的每一位
	int k = 0;
	while(n != 0){
		a[k] = n % d;
		k++;
		n = n / d;
	}
	//判断回文：循环数组长度一半，判断对称位置是否有不等
	for(int i = 0;i < k / 2;i++){
		if(a[i] != a[k - i - 1]){
			r = false;
			break;
		}
	}

	return r;
}
```

```
int main(){
	int x,n,i;
	cin>>n;
	for(i = 0;i < n;i++){
		cin>>x;

		if(huiwen(x,10) == false && (huiwen(x,2) == true || huiwen(x,16) == true)){
			cout<<x<<endl;
		}
	}
}
```

三、可选作业

1288：【入门】正整数 n 转换为 8 进制

1291：【入门】八进制转十进制

1405：【基础】小丽找潜在的素数？

1547：【基础】小 X 转进制

1415：【入门】10 进制转 D 进制

2.2 二进制和八进制、十六进制互转

一、2、8、16 进制的相互转换

1、**原理：**每位八进制数相当于 3 位 二进制数，每位十六进制数相当于 4 位 二进制数。在转换时，中间的 0 不能省略，开头不够时可以补 0 。

2、**实例：将 2 进制 1011010 分别转换为 8 进制和 16 进制**

3、**实例：将 16 进制数 2F78 转换为 2 进制数**

4、**实例：将 8 进制 153 转换为 2 进制**

1	5	3
~~00~~1	101	011

5、作业练习

(137)$_8$ = (_____)$_2$　　(2076)$_8$ = (_____)$_2$

(2A0F)$_{16}$ = (_____)$_2$　　(2A0F)$_{16}$ = (_____)$_2$

(11110111110110)$_2$ = (_____)$_8$

(11110111110110)$_2$ = (_____)$_{16}$

二、2 进制、8 进制、16 进制转换的实现

1294：【基础】二进制转十六进制

(0010 1101 0111 1011)$_2$ = (2 D 7 B)$_{16}$

思路：

第一步：判断字符串的长度是否是 4 的倍数，如果不是，则补 0。

s：字符数组

s.size() % 4 == 1，补 3 个 0

s.size() % 4 == 2，补 2 个 0

s.size() % 4 == 3，补 1 个 0

第二步：每 4 位 2 进制转换为 1 位的 16 进制输出。

“1101”转换为对应的十进制整数->13，注意判断转换的结果如果是 0~9，转换为’0’~’9’，如果转换的结果是 10~15，转换为’A’~’F’。

```
//将 4 位的二进制转换为 1 位的 16 进制
char num(string s){

}
```

```
#include <bits/stdc++.h>
using namespace std;

//将 4 位的 2 进制转换为 1 位的 16 进制
char num(string s){
    //1101，从最低位开始按权展开，转换为 10 进制
    //再转换为 16 进制的字符
    int r = 0,i,t = 1;
    for(i = s.size() - 1;i >= 0;i--){
        r = r + (s[i] - '0') * t;
        t = t * 2;
    }

    char c;//存储 1 位的 16 进制字符
    if(r < 10){
        c = r + '0';
    } else{
        c = r + 'A' - 10;
    }
```

```
    return c;
}

int main(){
    string s,t;//存放二进制
    cin>>s;
    //补 0
    if(s.size() % 4 == 1){
        s = "000" + s;
    } else if(s.size() % 4 == 2){
        s = "00" + s;
    }else if(s.size() % 4 == 3){
        s = "0" + s;
    }

    //每 4 位一格，将 4 位的二进制转换为对应的 16 进制
    for(int i = 0;i < s.size();i = i + 4){
        t = s.substr(i,4);
        cout<<num(t);
    }
}
```

1306：【基础】十六进制转二进制

$(1A5)_{16} = (\ 000\,1\ 1010\ 0101\ )_2$

思路：将每一位的 16 进制数，转换为 4 位的二进制数！

第一步：将每位 16 进制转换为 4 位的 2 进制，连接到字符串上！

第二步：删除前导 0，也就是要从第一个非 0 开始输出！

```
#include <bits/stdc++.h>
using namespace std;

string t[16] =
{"0000","0001","0010","0011","0100","0101","0110","0111","1000","1001","1010","1011
","1100","1101","1110","1111"};
int main(){
    //s：存放 16 进制，r：存放 2 进制
    string s,r;
    cin>>s;
    //s = "1A5"
    //s[i]转换为 0~15 之间的整数，然后再求对应的 4 位 2 进制
    int x;
    for(int i = 0;i < s.size();i++){
        //如果是 0~9
        if(isdigit(s[i])){
            x = s[i] - '0';
        } else{
            x = s[i] - 'A' + 10;
        }
        //cout<<x<<endl;
        r = r + t[x];
    }

    //删除前导 0
    //000110101111
    //当 r[0]是'0'，则删除
    while(r[0] == '0'){
        r.erase(0,1);
    }

    if(r == ""){
        cout<<0;
    }else{
```

```
        cout<<r;
    }
}
```

三、作业

1359：【基础】八进制转换二进制

1293：【基础】二进制转换八进制

1295：【基础】十六进制转换