- 计算机内部的数值用**补码**表示
- 多字节赋值给少字节变量时，低字节以“**截断赋值**”的方式
- 变量定义为什么类型：signed 、unsigned
  %d 显示时是**根据变量的类型显示**
- 数值进行加、减法是是根据补码的加、减运算。
  符号位一起参与运算
  超出的符号位丢弃（仅保留1位符号位）
  减法转化为加这个数的负

```
char v6;
v6 = 127;
printf("%d\n",v6);
```

```
 127
```

```
char v6;
v6 = 129;
printf("%d\n",v6);
```

-127

$(129)_{10}$ = 10000001

减1

10000000

取反

11111111

```
char v6;
v6 = 130;
printf("%d\n",v6);
```

-126

$(130)_{10}$ = 10000010

减1

10000001

取反

11111110

```
char v6;
v6 = 128;
printf("%d\n",v6);
```

-128 $(128)_{10}$ = 10000000

10000000

最小的数-128的补码

```
char v6;
v6 = 257;
printf("%d\n",v6);
```

1 $(257)_{10} = 100000001$

截断赋值 00000001

```
unsigned char v6;
v6 = 129;
printf("%d\n",v6);
```

129 $(129)_{10} = 10000001$ 最高位表示数值

```
char v1,v2;
v1 = 127;
v2 = v1+3;
printf("v1=%d,v2=%d\n",v1,v2);
```

v1=127,v2=-126

$(127)_{10}$ = 01111111

\+ 00000011

10000010

10000010 → 减1

10000001 → 取反

11111110

=-126

```
char v1,v2;
v1 = -127;
v2 = v1-2;
printf("v1=%d,v2=%d\n",v1,v2);
```

v1=-127,v2=127

也可直接写出-129的补码

$(-127)_{10}$ = 10000001

\+ 11111110 -2的补码

101111111 最高符号1舍弃（仅保留1位符号位）

- 溢出的概念：
- 现象：两个正数相加，结果为负；或两个负数相加，结果为正
- 原因：超过了其表示的最大范围
  例如：char 范围 -128~127

unsigned char 范围　　0~255

可以先按照十进制数的值进行计算，
按要求转换为二进制，并求其真值

练习1

```
char v1,v2;
v1 = -126;
v2 = v1-2;
printf("v1=%d,v2=%d\n",v1,v2);
```

v1=-126,v2=-128

$(-126)_{10}$ = 10000010

\+ 11111110 **-2的补码**

10000000 **-128的补码**

练习2

```
char v1,v2;
v1 = 126;
v2 = v1+3;
printf("v1=%d,v2=%d\n",v1,v2);
```

v1=126,v2=-127

$(-126)_{10}$ = 01111110

\+ 00000011 **3的补码**

10000001 **-127的补码**

练习3

```
char v1,v2;
v1 = -125;
v2 = v1-5;
printf("v1=%d,v2=%d\n",v1,v2);
```

v1=-125,v2=126

$(-125)_{10}$ = 10000011

\+ 11111011 **-5的补码**

101111110 **最高位舍弃，126的补码**

练习4

```
unsigned char v1,v2;
v1 = 253;
v2 = v1+6;
printf("v1=%d,v2=%d\n",v1,v2);
```

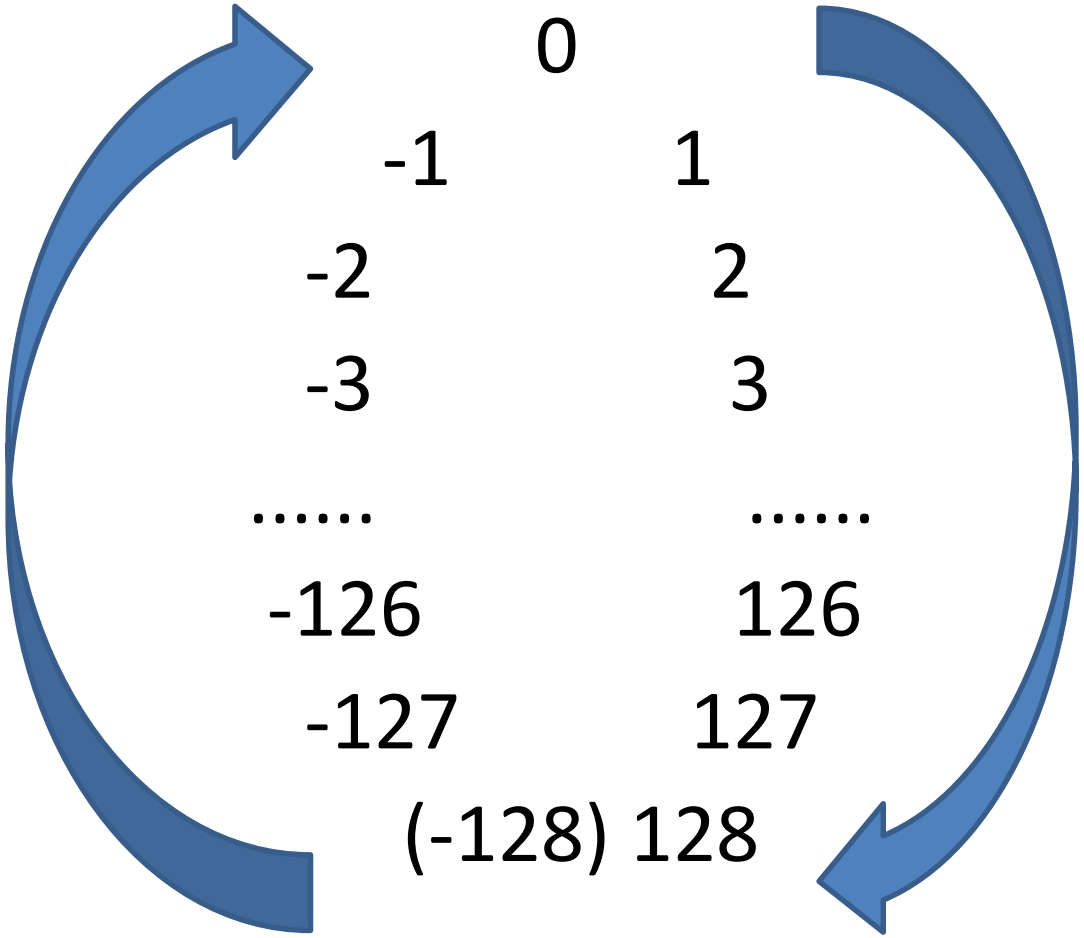
v1=253,v2=3

$(253)_{10}$ = 11111101

\+ 00000110 **6的补码**

100000011 **最高位舍弃，3的补码**

char 类型（-128~127）加法



物极必反，其余类型、运算类似

练习5

```
char v1,v2;
unsigned char v3;
v1 = 126;
v2 = v1+5;
v3=v1+5;
printf("v2=%d,v3=%d\n",v2,v3);
```

v2=-125,v3=131

练习6

```
char v1,v2;
unsigned char v3;
v1 = 3;
v2 = v1-5;
v3=v1-5;
printf("v2=%d,v3=%d\n",v2,v3);
```

v2=-2,v3=254

11111110 -2的补码
按unsigned 解释为254

练习7

```
char v1,v2;
unsigned char v3;
v1 = -126;
v2 = v1-3;
v3= v1-3;
printf("v2=%d,v3=%d\n",v2,v3);
```

v2=127,v3=127　　两个数都溢出

请仔细阅读并理解

- P44 图3.6
- P46 图3.7
- P48 图3.8
- P49 脚注
- P62 图3.15 图3.16