```
//(2.4)reinterpret_cast:编译的时候就会进行类型转换的检查
//reinterpret: 重新解释。(将操作数内容解释为另一种不同的类型【能把操作数的类型都转了】);
//处理无关类型的转换。也就是两个转换类型之间没有什么关系。就等于可以乱转,自由转。怎么转都行，很随意。
//常用于如下两种转换:
//a)将一个整型（地址）转换成指针，一种类型指针转换成另一种类型指针,按照转换后的内容重新解释内存中的内容;
//b)也可以从一个指针类型转换成一个整型。
/*int i = 10;
int *pi = &i;
int *p2 = reinterpret_cast<int *>(&i);
char *pc = reinterpret_cast<char *>(pi);*/

int i = 10;
int *pi = &i;
void *pvoid = reinterpret_cast<void *>(pi);
//...
int *pi2 = reinterpret_cast<int *>(pvoid);

//被认为是危险的类型转换；随便转，怎么搞都行，编译器都不报错。
```

```
int iv1 = 100;
long long lv1 = 8898899400; //8字节的。    十六进制: 2   126A    6DC8
int *piv1 = (int *)iv1;  //c语言风格； 0x00000064 {???}
int *piv2 = reinterpret_cast<int *>(iv1); //0x00000064 {???}
piv2 = reinterpret_cast<int *>(lv1); //0x 126a 6dc8 {???} 把前边的2丢了，因为piv2是4字节。
long long ne = reinterpret_cast<long long>(piv2); //指针类型转整型 = 308964808    = 126A   6DC8
```