C 语言中的未定义行为

C 语言中的未定义行为（Undefined Behavior）是指 **C 语言标准未做规定的行为**。同时，标准也从没要求编译器判断未定义行为，所以这些行为有编译器自行处理，在不同的编译器可能会产生不同的结果，又或者如果程序调用未定义的行为，可能会成功编译，甚至一开始运行时没有错误，只会在另一个系统上，甚至是在另一个日期运行失败。当一个未定义行为的实例发生时，**正如语言标准所说，“什么事情都可能发生”，也许什么都没有发生**。

所以，避免未定义行为，是个明智的决定。本文将介绍几种未定义行为，同时欢迎读者纠错和补充。

1.同一运算符中多个操作数的计算顺序（&&、||、?和,运算符除外）

例如：x = f()+g();　　//歧义

f()和 g()谁先计算由编译器决定，**如果函数 f 或 g 改变了另一个函数所使用变量的值，那么 x 的结果可能依赖于这两个函数的计算顺序**。

参考：《C 程序设计语言（第 2 版）》 P43

2.函数各参数的求值顺序

例如：　printf("%d,%d\n",++n,power(2,n)); //错误

在不同的编译器可能产生不同的结果，这取决于 n 的自增运算和 power 调用谁在前谁在后。

3.通过指针直接修改 const 常量的值

直接通过赋值修改 const 变量的值，编译器会报错，但通过指针修改则不会，例如:

```
int main()
{
   const int a = 1;
   int *b = (int*)&a;
   *b = 21;
   printf("%d, %d", a, *b);
   return 0;
}
```

a 输出值也由编译器决定。