条款1：理解模板类型推导

① `template< typename T>`
`void f(const T param);`
`void f(T const param);`

② `fun(int a);` 与 `fun(const int a);` 不构成重载

顶层const不构成重载

③指针的引用：`int *a = &b;` `int *&c = a;`

④鬼畜的函数指针与函数引用，

a. 函数指针的底层const似乎只能由类型别名表示出来

b. 函数引用的底层会被编译器无情忽略

模板 { `template<typename T>` ⟸ 写法固定
`void f(ParamType param);` ⟸ 分情况讨论

调用 `f(expr);` ⟸ 分情况讨论

ParamType：`T`；`T*`；`T&`；`T&&`
`const T`；`const T*`；`const T&`；`const T&&`
`T* const`；`const T* const`

expr：将上面的T全换成int；字面量 ①

`int [10];` `bool(int, int)`
`int (*)[10];` `bool (*)(int, int)`
`int (&)[10];` `bool (&)(int, int)` ②

① 理解顶底层const，缺少补啥　　ParamType param = expr;

② 理解数组与函数能够退化为指针

③ 通用引用(万能引用)，传入左值 ParamType *左值引用

int &与 && 折叠为 int &　　传入右值 ParamType *右值引用

T&&只有这么写

④ 引用不存在退化，即函数/数组在拷贝时不会变为指针