Lesson 13

1.泛型就是通用的技术。泛型编程不等于模板，后者只是泛型的其中一种手段。

2.模板就是一种生产函数或者类的模型，以 T add (const T &a, const T &b)为例，当使用 add(1, 2)的时候，编译器产生一个 int add(int, int) 的版本，当使用 add(string("hello"), string("world"))时，编译器产生 string add(const string &, const string &)的版本。

3.vector 不是一个类，而是一个类模板，以它为范本，根据实际调用产生的 vector<int>/vector<string>才是真正的类。

4.模板就是一种代码产生器，或者是一个代码输出函数，输入的为类型，输出的是符合该类型运算的类或者函数。

5.模板的这种代码产生功能，称为一种编译期多态。

6.模板代码在编译时，只编译那些需要的部分，所以当声明 map<Test, int> m;而不做其他使用时，即使 Test 不支持<操作，仍然没有错误。

7.typedef 不能写为全局，因为没有使用的时机。以 typedef Node<T> *NodePtr，因为与模板相关的名字都是不完整的，所以直接使用 NodePtr 会找不到符号（编译器不会反向推断 Node<T>中 T 的类型），使用 Node<T>，这种 typedef 无意义。

8.在 Queue 的编写中，如果：

```
template <typename T>
class Node{
    friend class Queue;
    private:
        T data_;
        Node *next_;
};
```

这里表明 Node<T>的友元类是 Queue，这个 Queue 是普通类，与模板无关。

```
template <typename T>
class Node{
    friend class Queue<T>;
    private:
        T data_;
        Node *next_;
};
```

此时的 Queue 不是一个模板类，无法这样使用，这里需要告诉编译器，Queue 是一个模板类。

```
template <typename T>
class Queue;
```

这两行的作用是告诉编译器，Queue 不是一个普通的类，而是一个模板类，所以下面的 friend 才能使用 Queue<T>。

正确的代码如下：

```
template <typename T>
class Queue;   //这里告诉编译器，Queue 是模板

template <typename T>
class Node{
    friend class Queue<T>; //Node<T>的友元是 Queue<T>
    private:
        T data_;
        Node *next_;
};
```

9.模板之间相互声明友元的要点：

a) 被声明为 friend 的类，首先应该告知编译器，这是一

个模板类，这需要带模板参数列表的前向声明。

b) 声明 friend 时，应该指定具体的模板参数。

10.Linux 进程打开一个文件描述符，涉及到三个数据结构：

a) 进程表项，归单个进程所有，记录该进程打开的所有的文件描述符。

b) 文件表项，由内核管理，可对应一次打开文件操作，里面记录打开文件的标志、文件偏移量以及指向文件的指针，以及该结构的引用计数（此时有多少个 fd 指向它）。

c) V 节点项，每项对应一个文件，记录着该文件的信息（大小、创建时间、修改时间、所有者、权限）。

11.stdin、stdout、stderr 在内部一定是调用了 STDIN_FILENO、STDOUT_FILENO、STDERR_FILENO 这三个文件描述符。

12.read 函数面向的单位是字节，与字符串无关。

13.常用的 open 标志组合：

a) O_RDONLY：代表只读

b) O_WRONLY | O_CREAT：以只写方式打开，文件不存在则创建

c) O_WRONLY | O_CREAT | O_APPEND：只写打开，文件不存在则创建，文件存在则进行追加

d) O_WRONLY | O_CREAT | O_TRUNC：只写打开，不存在则创建，文件存在则清空

e) O_WRONLY | O_CREAT | O_EXCL：只写打开，不存在则创建，文件存在则打开失败。

14.文件空洞：当文件偏移量超过文件大小的时候，就产生了文件空洞，它被记录的文件信息中，但是在磁盘上并不占据过多的磁盘空间。例如我们在文件中偏移 1G 的大小，那么在文件信息中就记录了 1G 的空洞，但是在磁盘上并不占据实际空间。

15.下列情形：

a) 一个进程打开两个不同的文件，进程表项 1 个，文件表项 2 个，V 节点表项 2 个。

b) 一个进程两次打开同一个文件，进程表项 1 个，文件表项 2 个，V 节点表项 1 个。

c) 两个进程打开同一个文件，进程表项 2 个，文件表项 2 个，V 节点表项 1 个。

d) 一个进程打开一个文件，然后通过 dup 进行复制，进程表项 1 个，文件表项 1 个（其中的引用计数为 2），V 节点表项为 1 个。

16.Linux 下分配文件描述符的规则是：寻找最小可用的。

17.通过 dup 复制的 fd，共享偏移量。在内核中的数据结构表示为：1 个进程表项，文件表项也是 1 个，但是有两个文件描述符指向它。

18.文件表项中有一项为引用计数，dup 调用后，两个文件描

述符指向同一个文件表项，该表项中的引用计数为 2，close 其中一个 fd 时，引用计数减一，仅当引用计数为 0 的时候，该表项才会被销毁。

19.dup2 和 dup 的区别：

a) dup 由系统分配 fd，而 dup2 手工指定。

b) 如果 dup2 指定的 fd 已经占用，那么会自动 close

20.实现标准流重定向是通过复制 fd 实现的。

21.复制 fd 有三种方法

a) dup

b) dup2

c) fcntl，设置选项为 F_DUPFD