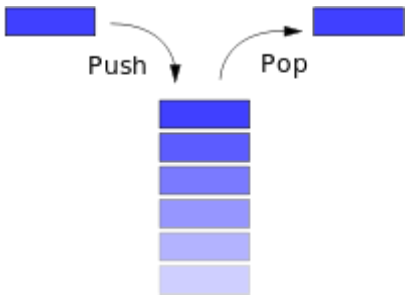
Smart Pointer & Template 练习

栈是一种特殊的数据结构，它的特殊之处在于只允许在一端进行加入数据和输出数据的运算。



此次练习中需要大家实现这样的一个数据结构。要求将其**实现为一个模板 Stack<T>**，可存储类型为 T 的数据（其中要求 T 具有复制构造函数及复制赋值运算符的定义，任意满足要求的类型都需要支持）。具体接口要求如下：

Stack()：默认构造函数，构造一个空栈。

我们要求栈不可被以任何形式复制（思考：若可以复制，动态管理的内存如何处理？）：

Stack(const Stack&) = delete;

Stack& operator=(const Stack&) = delete;

void push(T elem)：上述的 push 操作，将 elem 加入栈中。

bool pop(T& cell): 上述 pop 操作，如栈空，则直接返回 false，否则弹出一个元素到 cell 中并返回 true.

我们要求栈存放的数据容量不预设上限，能根据用户 push 进去的元素个数动态扩容。所以，需要使用动态内存分配，这次作业中我们要求**不允许使用裸指针，必须使用智能指针管理动态分配的内存**。

同时此次作业**不允许直接使用 STL 已实现的容器**，包括但不限于 stack，deque，list 和 vector，请自行实现内部数据结构。(Hint: list?)

提交要求：请将你设计的 Stack 写在 stack.hpp 这一个文件内，用户 include 该文件，即可通过上面约定的接口使用这个 Stack。提交时仅提交 stack.hpp

评分标准：

1. 栈的基本功能实现（50%），如预设上限最多得分 25%
2. Template 使用（20%）
3. 智能指针使用（20%）
4. 代码风格（10%）