写在前面

Overall

本实验报告的基本结构如下（按逻辑顺序）：

模块	编号	内容
STL 风格的 泛型的 基础数据结构 容器实现	1	基于双向链表的`linkedList`
	2	基于增长数组的`vector`
	3	基于块状数组的`dataBlock`
	4	实现基于循环增长数组的`deque`
	5	基于`vector`实现`stack`
	10	基于 R-BTree 实现`set`
	11	基于 R-BTree 实现`map`
	15	基于 Heap 实现`priority_queue`
基础树/图结构	6	树上 dfs（基础信息）
	7	图上 bfs（最短路）
	8	二叉树三序遍历
	9	R-BTree 的基本实现
特殊结构 及其应用	12	字典树`Trie`
	13	线段树`segTree`
	14	堆`Heap`
	16	霍夫曼树`Huffman-tree`
在算法中应用	17	算数表达式求值（栈）
	18	括号匹配（栈）
	19	高精度计算

本实验报告的所有代码文件、commit 记录等已经在 Github 上同步了所有相关资料: `https://github.com/GYPpro/DS-Course-Report`

仓库会在 7 月 1 日前保持`private`

容器设计原则

Interface Reference

```
Many containers have several member functions in common, and
share functionalities. The decision of which type of container
to use for a specific need does not generally depend only on
the functionality offered by the container, but also on the
efficiency of some of its members (complexity). This is
especially true for sequence containers, which offer different
trade-offs in complexity between inserting/removing elements
and accessing them.

stack, queue and priority_queue are implemented as container
adaptors. Container adaptors are not full container classes,
but classes that provide a specific interface relying on an
object of one of the container classes (such as deque or list)
to handle the elements. The underlying container is
encapsulated in such a way that its elements are accessed by
the members of the container adaptor independently of the
underlying container class used.
```

在模块`STL 风格的、泛型的、基础数据结构容器实现`中所有的数据结构，遵循以下规范：

- 提供增删改查的接口；
- 提供完整的泛型系统；
- 提供迭代器用于遍历；
- 按结构性质提供随机访问

为了更契合`OOP`中的`SRP`原则，对于原 STL 中的容器适配器，本实验报告中并不通过接口实现，而是另起一个类。