关于AllocatorSequential类的说明

|  |  |  |
| --- | --- | --- |
| 方法 | 说明 | 时间复杂度 |
| AllocatorSequential(int size); | 构造方法，调用init方法初始化存储池 |  |
| void init(int size); | 初始化存储池 |  |
| Variable newVariable(String variableName, int size) throws InsufficientMemoryPoolException; | 为变量申请空间 | 首先适配：  最佳适配：  最差适配： |
| protected int pickFreeBlock(int size); | 查找空闲块（首先适配） |  |
| protected int pickFreeBlock(int size); | 查找空闲块（最佳适配） |  |
| protected int pickFreeBlock(int size); | 查找空闲块（最差适配） |  |
| boolean write(Variable variable, String value) throws InsufficientVariableSizeException; | 向变量存储空间写入数据 | m为变量值长度，忽略m的长度则为 |
| String read(Variable variable) throws VariableNotAssignedException; | 从变量存储空间读数据 | m为变量值长度，忽略m的长度则为 |
| void deleteVariable(Variable variable); | 删除变量 |  |
| void show(AllocatorADT allocator, List<String> sortVariableList); | 展示存储池 |  |
| public int getBlockSize(int pos); |  |  |
| public List<BlockInfo> getBlockInfoList(List<String> sortVariableList); |  |  |
| public List<BitBlockInfo> getBitBlockInformationList(); |  |  |

关于deleteVariable方法。考虑删除一个变量并释放它占用的存储块时，共有9种可能的情况如下：

分别采用如下的处理方法：

①freelist为空：将F标记为唯一的空闲块，并令freelist指向F；

②左侧空闲块P相邻：将P与F合并；

③左侧空闲块P不相邻：将F标记为新的空闲块，调整P的指针，调整原本左指针指向P块的指针；

④右侧空闲块S相邻：将S与F合并，并调整原本指向S块的指针，令freelist指向S与F合并后的块；

⑤右侧空闲块S不相邻：将F标记为新的空闲块，调整原本右指针指向S块的指针；

⑥左侧P相邻，右侧S相邻：将P、F、S合并，调整原本指向S块的指针，，令freelist指向P、F、S合并后的块。

⑦左侧P相邻，右侧S不相邻：将P与F合并；

⑧左侧P不相邻，右侧S相邻：将S与F合并，调整原本指向S块的指针，令freelist指向S与F合并后的块；

⑨左侧P不相邻，右侧S不相邻：将F标记为新的空闲块，并调整P、F、S的指针。

关于AllocatorSequential的其它方法，都来源于书上，且十分简单朴素。