2020-1학기 자료구조 (채옥삼 교수님) – 프로그램과제1 ADT와 명세서

2016101156 컴퓨터공학과 김현기

**1. 프로그램 구조 및 기능 요약도.**

|  |
| --- |
| Application |
| 프로그램의 전체적인 동작을 관리한다.  ADT :  -SortedList<ItemType> m\_MasterList  -TempType m\_TempList  -StorageType m\_StorageList  -int m\_command  기능 :  -메인 어플리케이션 실행  -커맨드 입력  -각 커맨드에 맞는 함수 실행  -GetCommandMaster  - |

|  |  |  |
| --- | --- | --- |
| MasterList(DoublyLinkedSortedList) | TempList(TempType) | StorageList(StorageType) |
| 저장될 공간을 정한 Item을 전부 모아둔 List  ADT :  - int m\_Length  -DoublyNodeType<ItemType>\* m\_pFirst  -DoublyNodeType<ItemType>\* m\_pLast  기능 :  -물건 개수 반환  -물품 추가  -물품 제거  -물품 수정  -물품 검색(여러 조건으로)  -모든 물품 출력  -물품 검색 후 TempList로 보내기 | 저장될 공간을 정하지 못한 Item의 임시저장소  ADT :  - int NumOfItems  -Queue<ItemType>m\_TempItemList  기능 :  -물건 개수 반환  -물품 추가  -물품 제거  -물품 검색(여러 조건으로)  -물품 수정  -리스트 비우기  -모든 물품 출력  -물품 MasterList로 보내기  -물품 자동 정리 | MasterList의 Item을 정리한 container를 모아둔 List  ADT :  -int Id  -int m\_CurrentNum  -int m\_MaxNum  -SingleLinkedList<ContainerType>  ContainerList  기능 :  -컨테이너 개수 반환  -컨테이너 추가  -컨테이너 제거  -컨테이너 검색  -전체 컨테이너 리스트 출력  -컨테이너 하나 리스트 출력 |

|  |  |
| --- | --- |
| TempItemList(DoublyLinkedQueue) | ContainerList(SingleLinkedList) |
| TempList에서 있는 item을 모아둔 List  ADT :  - DoublyNodeType <ItemType>\* m\_pFront  - DoublyNodeType <ItemType>\* m\_pRear  기능 :  물품 추가  물품 제거  물품 수정  물품 검색 | Container를 모아둔 List  ADT :  -NodeType<ContainerType>\* m\_pList  -int m\_Length  기능 :  -컨테이너 개수 반환  -컨테이너 추가  -컨테이너 제거  -컨테이너 검색 |

|  |
| --- |
| ContainerType |
| Container에서 가지고 있는 정보들  ADT :  -int m\_Id  -int m\_ContainerRep  -string m\_Location  -UnsortedList<SimpleItemType> m\_sItemList  -UnsortedList<string> m\_PhotoList  기능:  물품 추가  물품 제거  물품 수정  물품 출력  사진 추가  사진 제거  사진 출력  컨테이너 대표 물품 종류 출력 |

|  |  |
| --- | --- |
| sItemList(UnsortedList) | PhotoList(UnsortedList) |
| 각 container에 들어있는 물품의 List | Container의 사진을 담고있는 List |

|  |  |
| --- | --- |
| ItemType | SimpleItemType |
| Master List와 TempList에서 가지고 있는 물품의 정보들  ADT :  -int m\_Kind  -string m\_Name  -int m\_Price  -int m\_PurchaseDay  -int m\_Id;  -int m\_ContainerId  -int m\_StorageId | ContainerType에서 가지고 있는 물품의 간단한 정보들  ADT :  -int m\_Kind  -string m\_Name  -int m\_Id; |

**2. 사용된 자료구조**

1) MasterList : DoublyLinkedList

- ArrayList는 정해진 MaxSize보다 커지면 다시 삭제하고 MaxSize를 늘린다음에 다시 만들어야하는데 LinkedList는 정해진 MaxSize가 없이 추가하는 만큼 늘어난다. MasterList 특성상 많은 Item을 저장할 텐데 MaxSize가 될 때마다 배열을 지우고 다시 생성하는 작업을 반복하면 비효율적이라 생각했다. 그리고

2) TempList : DoublyLinkedQueue

- TempList는 저장 장소를 정하지 못한 물품들을 보관하는 임시저장소이다. 따라서 검색 기능보다는 자료의 삽입과 삭제 기능이 주가 될 것이므로 LinkedList로 짤 것이다. 그리고 DoublyLinkedList가 SingleLinkedList보다 메모리는 더 많이 필요하지만 삽입과 삭제 속도가 빠르고 메모리 복원에 유리하기 때문에 DoublyLinkedList를 쓸 것이고 먼저 들어온 아이템부터 저장할 곳을 정해줄 것 이기에 선입선출의 기능을 가진 DoublyLinkedQueue로 구현해 줄 것이다.

3) StorageList : SortedList

-Storage는 보통 몇 개를 만들어 두면 Storage를 삽입하거나 삭제하는 일은 별로 없기 때문에 검색이 위주가 될 것이다. 그래서 LinkedList보다 ArrayList로 짤 것이다. 따라서 순차적으로 배열한 SortedList를 사용할 것이다.

4) ContainerList : SingleLinkedList

-

5) sItemList, PhotoList : UnSortedList

-순서는 상관없고 단순히 정보를 저장하기 위해 ArrayList로 UnSortedList를 구현할 것이다.

**3. 프로그램의 주요 기능**

1) 프로그램 설명

- 자료구조 실습 내용을 바탕으로, 기본적인 아이템추가, 제거, 갱신, 검색 등 물건정리 기능을 수행할 수 있음

- 프로그램은 크게 전체적인 프로그램을 관리하는 Application과 MasterList, StorageList, TempList로 구분할 수 있음

- MasterList는 전제적인 아이템을 관리, StorageList는 서랍장을 여러 개 놓았다고 생각할 수 있고, StorageType안에는 서랍장의 서랍들을 모아두었다 생각할 수 있는 ContainerList가 있다. 그리고 ContainerList에 물건의 리스트인 sItemList가 있다. 그리고 TempList에는 저장할 공간을 아직 정하지 못한 물건들을 관리한다.

- Queue, UnsortedList… 와 같이 다른 Class의 기본이 되는 자료구조는 template function을 이용해 Generic하도록 구현했음

- ItemType, SimpleItemType같은 template에 들어가는 class에는 대입연산자나 비교연산자를 operation overloading을 해준다.

2) 프로그램의 핵심 특징

■ MasterList

- 저장할 곳이 정해진 물품(item)의 입력, 제거, 수정, 검색등 물품을 관리하는 프로그램

- 저장할 곳을 변경하려고 하는데 어디에 놓을지 모르겠는 물품은 다시 TempList로 옮길 수 있다

.

■ StorageType

- 컨테이너의 입력, 제거, 수정, 검색등 컨테이너를 관리함

- 각각의 Container에 많이 저장된 물품의 종류를 바탕으로 그 Container의 대표하는 종류를 가짐

- Container에는 PhotoList랑 SimpleList를 추가, 제거, 수정등 관리할 수 있음

■ TempList

- 어디에 저장될지 모르는 물품을 모아둔 임시저장소

- 자동분류기능 : TempList의 물건과 같은 종류의 물품을 대표 물품으로 가지는 container에 자동분류해줌