

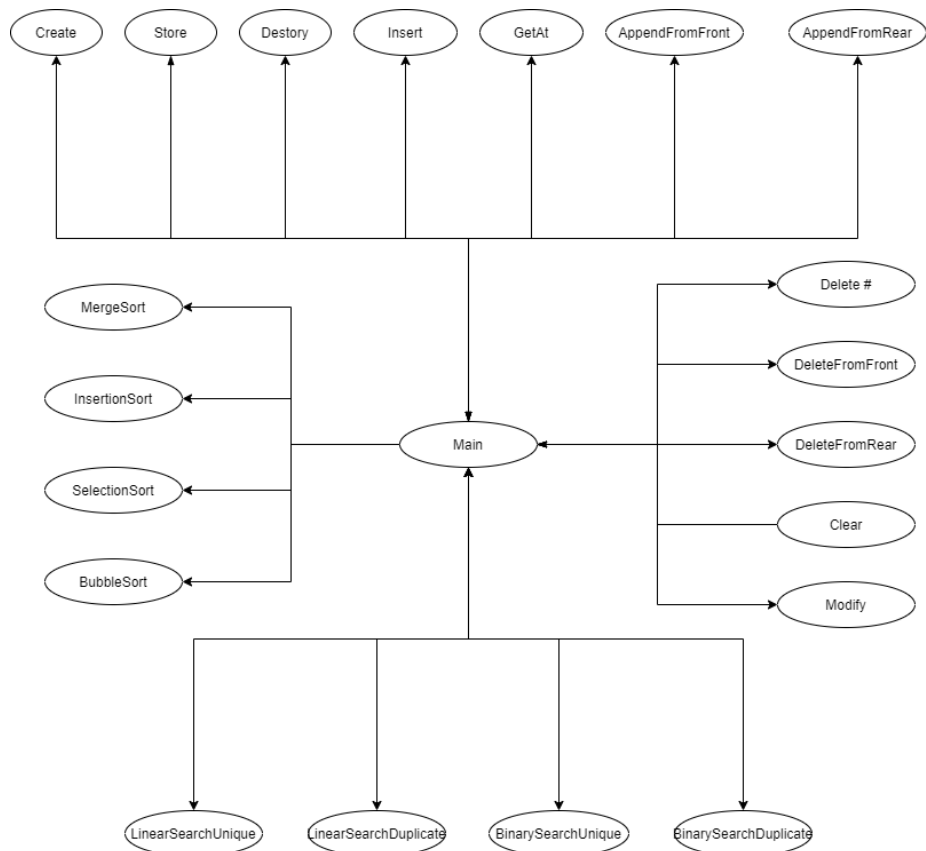
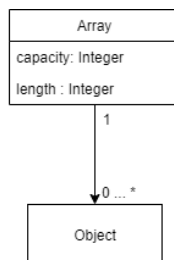
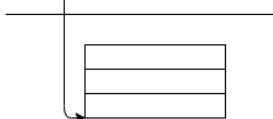
# Array

by D4R6

# Array Design

Array

front	capacity	length
*	3	0



## Create

### 처리 과정

1. 할당량을 입력 받는다.
2. 할당량 만큼 메모리를 할당 한다.
3. 끝낸다.

Create
start
array, capacity
read capacity
array.front(capacity)
array.capacity = capacity
array.length = 0
stop

## Destory

### 처리 과정

1. array 를 할당 해제한다.
2. 끝낸다.

Destory
array
array.front = null
stop

## Store

### 처리 과정

1. 위치와 개체를 입력 받는다.
2. 받은 개체를 위치에 적는다.
3. 위치를 출력한다.
4. 끝낸다.

Store
start
array, index, object
read index, object
array.front(index) = object
array.length = array.length + 1
print index
stop

## GetAt

### 처리 과정

1. 위치를 입력 받는다.
2. 위치의 값을 출력한다.
3. 끝낸다.

GetAt
start
array
read index
print array.front[index]
stop

## PrintAll

### 처리 과정

1. length 만큼 반복한다.
  - 해당 index 값을 출력한다.
2. 끝낸다.

PrintAll
start
array, i = 0
i < array.length
print array.front[i]
i = i + 1
stop

## 처리 과정

1. 위치와 개체를 입력 받는다.
2. 새로운 배열을 할당 한다.
3. 위치보다 작은 동안 반복한다.
  - 기존 배열 요소를 옮겨 적는다.
4. 사용량 만큼 반복한다.
  - 기존 배열 요소를 옮겨 적는다.
5. 기존 배열을 지운다.
6. 새로운 배열의 위치 개체를 적는다.
7. 위치를 출력한다.
8. 끝낸다.

Insert
start
array, index, object, temps0, i = 1, j = 1
read index, object
temps(array.capacity + 1)
while( i < index )
temps(i) = array.front(j)
j = j + 1
i = i + 1
i = i + 1
while(j ≤ array.length)
temps(i) = array.front(j)
i = i + 1
j = j + 1
array.front = null
array.front = temps
array.capacity = array.capacity + 1
array.front(index) = object
array.length = array.length + 1
print index
stop

## 처리 과정

1. 개체를 입력 받는다.
2. 새로운 배열을 할당 한다.
3. length 만큼 반복한다.
  - 새로운 배열 두 번째 칸부터 기존 배열 내용을 기록한다.
4. 기존 배열을 새로운 배열로 대체한다.
5. 새로운 배열 첫 칸에 개체를 기록한다.
6. 위치를 출력한다.
7. 끝낸다.

AppendFromFront
start
array, index = 1, object, temps(), i = 1
read object
temps(array.capacity + 1)
while(i ≤ array.length)
temp(i + 1) = array.front(i)
i = i + 1
array.front = null
array.front = temp
array.capacity = array.capacity + 1
array.front(index) = object
array.length = array.length + 1
print index
stop

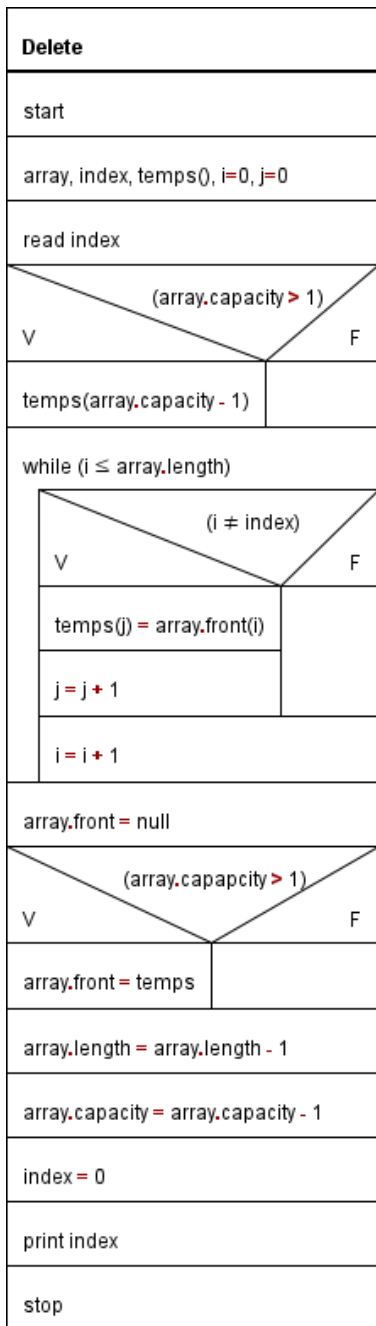
## 처리 과정

1. 개체를 입력 받는다.
2. 새로운 배열을 할당 한다.
3. length 만큼 반복한다.
  - 새로운 배열 첫 번째 칸부터 기존 배열 내용을 기록한다.
4. 기존 배열을 새로운 배열로 대체한다.
5. 새로운 배열 끝 칸에 개체를 기록한다.
6. 위치를 출력한다.
7. 끝낸다.

AppendFromRear
start
array, index = 1, object, temps()
read object
temps(array.capacity + 1)
while(index ≤ array.length)
temp(index) = array.front(index)
index = index + 1
array.front = null
array.front = temp
array.capacity = array.capacity + 1
array.front(index) = object
array.length = array.length + 1
print index
stop

## 처리 과정

1. 위치를 입력 받는다
2. 크기가 1 줄어든 배열을 만든다.
3. 사용량까지 반복한다.
  - 입력 받은 위치가 아니면 복사 반복.
  - 입력 받은 위치이면 skip
4. 기존 배열을 새로운 배열로 대체한다.
5. 위치 0을 적는다.
6. 위치를 출력한다.
7. 끝낸다.





## 처리 과정

1. 전체 용량보다 1 작은 배열을 생성한다.
2. 사용량까지 반복한다.
  - front 를 제외하고 복사 반복.
3. 기존 배열을 새로운 배열로 대체한다.
4. 위치 0을 적는다.
5. 위치를 출력한다.
6. 끝낸다.

DeleteFromFront	
start	
array, temps(), i=1	
(array.capacity > 1)	
V	F
temps(array.capacity - 1)	
while (i ≤ array.length)	
temps(i - 1) = array.front(i)	
array.front = null	
(array.capacity > 1)	
V	F
array.front = temps	
array.length = array.length - 1	
array.capacity = array.capacity - 1	
index = 0	
print index	
stop	

## 처리 과정

1. 전체 용량보다 1 작은 배열을 생성한다.
2. 사용량 -1 까지 반복한다.
  - 복사 반복.
3. 기존 배열을 새로운 배열로 대체한다.
4. 위치 0을 적는다.
5. 위치를 출력한다.
6. 끝낸다.

DeleteFromRear	
start	
array, temps(), i=1	
(array.capacity > 1)	
V	F
temps(array.capacity - 1)	
while (i ≤ array.length - 1)	
temps(i) = array.front(i)	
array.front = null	
(array.capacity > 1)	
V	F
array.front = temps	
array.length = array.length - 1	
array.capacity = array.capacity - 1	
index = 0	
print index	
stop	