Assignment 01 – array.h

user12201928@ee2f31c5f7f7: ~/Assignment_01

```
#pragma once

class DynamicArray {
private:
        int* m_data;
        int m_size;
        int m_capacity;
        int m_numDeleted;
        void doubleCapacity();

public:
        DynamicArray(int initailCapcity = 4);
        ~DynamicArray();

        void add(int val);
        int get(int idx);
        void put(int idx, int val);
        void remove(int idx);
        int size() const;
};
~
```

array.h 에서는 본래 주어진 파일에서 변경한 부분이 없으므로 생략

Assignment 01 – array.cpp

user12201928@ee2f31c5f7f7: ~/Assignment_01

```
#include "array.h"
#include<assert.h>
void DynamicArray::doubleCapacity()
{
        int* old_data = m_data;
        int old_size = m_size;
        int newCap = 2 * m_capacity;
        m_data = new int[newCap];
        for (int i = 0, j = 0; i <= old_size+m_numDeleted; i++){
                if(old_data[i] !=-1){
                        m_data[j] = old_data[i];
                        j++;
                }
        }
        m_size = old_size;
        m_capacity = newCap;
        m_numDeleted = 0;
}
```

void doubleCapacity() = 동적 배열의 용량을 두 배로 증가시켜주는 함수

현재 데이터 배열의 주소와 크기를 old_0000에 각각 저장
새로운 용량을 현재 용량의 두 배로 설정 후 다시 데이터 배열을 할당
이전 데이터 배열을 순회하면서 -1 즉, 삭제가 되지 않은 데이터들을 새로 할당한 배열로 복사
그 후 동적 배열의 크기와 용량을 설정 후 삭제된 데이터의 수를 0으로 재설정

Assignment 01 – array.cpp

user12201928@ee2f31c5f7f7: ~/Assignment_01

```
DynamicArray::DynamicArray(int initailCapcity)
{
        assert(m_capacity >= 0);
        m_capacity = initailCapcity;
        m_size = 0;
        m_numDeleted = 0;
        m_data = new int[m_capacity];
        assert(m_data != 0);
}

DynamicArray::~DynamicArray()
{
        delete[] m_data;
}
```

DynamicArray의 생성자와 소멸자

- 생성자
  assert를 사용해 용량의 크기가 0인지 확인
  초기용량을 배열의 용량으로 설정
  동적 배열의 크기, 삭제된 데이터 수를 초기화 후 동적 배열의 데이터를 저장할 공간을 초기 용량에 맞게 할당
  m_data 가 NULL이면 프로그램 종료

- 소멸자
  동적으로 할당된 메모리 m_data를 삭제

```
void DynamicArray::add(int val)
{
        if ((m_size+m_numDeleted) > m_capacity) doubleCapacity();
        m_data[m_size+m_numDeleted] = val;
        m_size++;
}
```

void add(int val) : 정수형 파라미터를 받고 동적 배열에 새로운 값을 추가 하는 함수

현재 데이터의 크기와 삭제된 데이터 수를 합산한 값이 현재 할당된 용량을 초과하는지 확인
초과한다면 doubleCapacity() 함수 실행
동적 배열의 맨 뒤에 인덱스에 val를 할당 (삭제된 값도 실제 -1로 저장되어 있기에 합산)
데이터의 크기를 증가

```
int DynamicArray::size() const
{
        return m_size;
}
```

int size() : 동적 배열에 값이 들어있는 크기를 반환하는 함수

m_size 값을 반환

12201928 이상혁

Assignment 01 – array.cpp

```
int DynamicArray::get(int idx)
{
        assert(idx<m_size && idx>=0);
        int count = -1;
        for(int i = 0; i < m_capacity; i++){
                if(m_data[i] != -1){
                        count++;
                }
                if(count == idx){
                        return m_data[i];
                }
        }
        return -1;
}
```

```
}
void DynamicArray::put(int idx, int val)
{
        assert(idx < m_size&& idx >= 0);
        int count = -1;
        for(int i = 0;i<m_capacity;i++){
                if(m_data[i]!=-1){
                        count++;
                }
                if(count == idx){
                        m_data[i] = val;
                }
        }
}
```

```
void DynamicArray::remove(int idx)
{
        assert(idx < m_size&& idx >= 0);
        int count = 0;
        for(int i = 0 ; i<idx; i++){
                if(m_data[i]==-1){
                        count++;
                }
        }
        for(int i = idx; i<m_size;i++){
                if(m_data[count+i] != -1 && m_data[i] != -1){
                        m_data[count+i] = -1;
                        m_numDeleted++;
                        m_size--;
                        return;
                }
        }
}
```

void get(int idx) : 동적 배열에서 특정 인덱스의 값을 반환하는 함수

assert를 사용하여 idx가 0이상, 데이터 크기보다 작은지 확인
count라는 변수 선언 -> 현재 배열에서 -1이 아닌 값의 개수를 세는 변수
배열의 요소를 순회하며 -1이 아닌 값이 있으면 count증가
idx 값이 -1(삭제한 요소)를 제외한 수의 개수인 count와 동일하다면 그 값을 반환.
주어진 값이 동일하지 않으면 -1로 에러를 나타냄

void put(int idx, int val) : 동적 배열에서 인덱스의 값을 덮어쓰는 함수

앞 부분의 코드는 get과 동일한 원리 맨마지막에 return을 해주지 않고 정수형 파라미터로 받은 val값을 동적 배열에 할당.

void remove(int idx) : 동적 배열의 특정 인덱스 값을 삭제하는 함수

이 부분도 get과 put과 같이 동일하게 작성해도 되었으나 반대 원리를 사용하였음.

count라는 변수를 통해 -1인 값의 개수를 센 후 주어진 인덱스에서 이 count를 더한 idx의 값을 -1로 대입

즉, 주어진 인덱스에 해당하는 값을 -1로 대체한 후 삭제된 것으로 표시

데이터가 삭제될때마다 numDeleted와 size 값을 업데이트 하여 동적 배열의 상태를 유지

12201928 이상혁

Assignment 01 - test

```
After adding 100 numbers:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65
66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96
97 98 99 100
After updating the first element to 999:
999
After removing the first element 10 times:
11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
```

주어진 main.cpp 코드
먼저 100개의 숫자를 add 후 get을 통해 받아오고 난 뒤에 첫번째 요소를 put을 사용해 99로 변경한다.
첫번째 요소를 10번 반복해 삭제를 한 이후에 다시 get을 통하여 확인을 해보았는데 정상적으로 작동함을 확인할 수 있었다.

```
Adding 0~9 num
0 1 2 3 4 5 6 7 8 9

remove(0) 3 time
3 4 5 6 7 8 9
After put(2,999)
3 4 999 6 7 8 9
Adding 10~49 num
3 4 999 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
 36 37 38 39 40 41 42 43 44 45 46 47 48 49
```

테스트한 코드 실행 결과

0~9까지의 수를 add 한 후 get을 통한 출력 시 정상적으로 나온다.
put으로 idx 2d에 999를 넣었더니 정상적으로 작동이 되었다.
이후 10~49로 doublecap이 정상적으로 작동하지는 확인해 주었다.

```
3 4 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45 46 47 48 49
```

remove(2)를 통해 999를 지워준 결과다.