8) 주어진 메인 함수와 실행 화면을 참고하여 배열을 조금 더 쉽게 사용할 수 있는 Array 클래스를 작성하고, 실행 화면처럼 동작하는 프로그램을 작성하세요.

- Array 클래스는 데이터 멤버로 capacity, size, arr을 갖습니다.
  - capacity(데이터를 담을 수 있는 최대 용량),
  - size(데이터가 저장된 수),
  - arr(힙에 생성한 배열의 첫 번째 요소를 가리키는 포인터)

- 배열 마지막 위치에 요소를 추가하는 멤버 함수 insert()를 갖습니다. capacity를 넘어서 요소를 추가할 경우에는 '배열이 가득 차 요소를 추가할 수 없다'는 메시지를 출력합니다.

- 배열의 요소를 출력하는 멤버 함수 print()도 갖습니다.

```
array size? 4
arr =   10   11   12   13
데이터 34는 추가할 수 없습니다. 배열이 가득 찼습니다.
```

```
int main(){
    int count;
    cout << "array size? ";
    cin >> count;

    Array array1(count); //크기가 count 인 배열 생성

    for (int i = 0; i < count; i++){
        array1.insert(i+10);
    }
    array1.print();
    array1.insert(34);
    cout << endl;
    return 0;
}
```

=

=

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
class Array{
private:
    int * arr;
    int capacity;
    int size;
public:
    Array(int capacity);  //매개변수로 받은 값을 크기로 갖는 배열을 생성
    ~Array();
    void insert(int value);
    void print() const;
};
// Constructor
Array::Array(int cap) : capacity(cap) {
    arr = new int[capacity];
    size = 0;
}
// Destructor
Array :: ~Array(){
    delete [] arr;
}
// the insert member function
void Array::insert(int value){
    if (size == capacity){
        cout << "데이터 " << value << "는 추가할 수 없습니다. 배열이 가득 찼습니다."
<< endl;
        return;
    }
    arr[size] = value;
    size++;
}
// The print member function
void Array::print() const{
    cout << "arr = ";
    for (int i = 0; i < size; i++) {
        cout << setw(4) << arr[i] << " ";
    }
    cout << endl;
}
int main(){
    int count;
    cout << "array size? ";
    cin >> count;

    Array array1(count); //크기가 count 인 배열 생성

    for (int i = 0; i < count; i++){
        array1.insert(i+10);
    }
    array1.print();
    array1.insert(34);
    cout << endl;
    return 0;
}
```