# HW3. Bag을 이용한 집합 연산 구현

## 1 과제 개요

교재의 ArrayBag 클래스 구현을 참고하여, 합집합, 교집합, 차집합의 기능을 추가로 구현한다.

## 2 과제 요구사항

`hw3 < test.in > test.out` 명령어로 실행되는 프로그램을 작성합니다.

`test.in`파일의 형식은 코드 1과 같습니다.

```
T
N1 M1
A1 A2 ... AN1
B1 B2 ... BM1
N2 M2
A1 A2 ... AN2
B1 B2 ... BM2
...
NT MT
A1 A2 ... ANT
B1 B2 ... BMT
```

Code 1: test.in

첫 번째 줄에는 테스트 케이스의 개수 T가 주어집니다. 각 테스트 케이스의 첫 줄에는 각 집합의 크기 N과 M이 주어지고, 그 다음 두 줄에 걸쳐 집합의 원소들이 공백으로 구분되어 주어집니다.

`test.out`에는 각 테스트 케이스에 대해 코드 2와 같이 연산의 결과를 공백으로 구분해 출력합니다.

```
R1 R2 ... Rk // result of union(A, B)
R1 R2 ... Rk // result of intersection(A, B)
R1 R2 ... Rk // result of difference(A, B)
R1 R2 ... Rk // result of difference(B, A)
```

Code 2: test.out

## 3 제출

### 3.1 제출 파일

- 소스 코드
  - ArrayBag.h
  - ArrayBag.cpp
  - hw3.cpp
- 보고서

  - hw3.tex
  - hw3.pdf
  - 보고서 컴파일에 필요한 모든 파일
  - 그 외 중간 산출물(aux, log 등) 있을 경우 0점 처리
  - 보고서에 들어갈 내용
    * 공부한 C++ 문법
    * 책에 없는 함수 구현 및 설명
    * 실행결과 스크린샷
    * 어려웠던 점

### 3.2 제출 방법 및 기한

- 제출 방법: 클래스룸 과제제출
- 모든 파일을 하나의 파일로 압축해 제출
- 제출 기한: 과제 제시일로부터 2주 뒤 23:59:59
  - 예시) 금요일에 실습 진행할 경우 2주 뒤 목요일 23:59:59
- 지각 제출: 1주일 기한, 50% 감점, TA 이메일로 제출

## 4 참고 사항

- Compile, 실행, 양식 등을 충족하지 못할 경우 0점
- 질문 : pemds81718@gmail.com
- 간단한 검색 등으로 나오는 내용은 답변하지 않습니다.
- 필요한 경우 hw5.cpp과 출력 연산자 오버로딩을 제외한 나머지 파일은 수정(함수 제거/추가/수정) 가능합니다.

## 5 예시 입출력

```
Input:
1
4 5
2 4 6 8
1 2 3 4 5
Output:
1 2 2 3 4 4 5 6 8
2 4
6 8
1 3 5
```

Code 3: 예시 입출력

## 6 VSCode 설정

```
{
  "tasks": [
    {
      "type": "shell",
      "label": "build with clang++",
      "command": "clang++",
      "args": [
        "-fdiagnostics-color=always",
        "-std=c++20",
        "-g",
        "hw3.cpp",
        "-o",
        "hw3"
      ],
      "options": {
        "cwd": "${workspaceFolder}"
      },
      "problemMatcher": ["$gcc"],
      "group": {
        "kind": "build"
      }
    }
  ],
  "version": "2.0.0"
}
```

Code 4: .vscode/tasks.json

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "build: clang++ > debug",
      "type": "lldb",
      "request": "launch",
      "program": "${workspaceFolder}/hw3",
      "args": [],
      "cwd": "${workspaceFolder}",
      "preLaunchTask": "build with clang++",
      "stopOnEntry": false,
      "stdio": [
        "${workspaceFolder}/test.in",
        "${workspaceFolder}/result.out",
        null
      ]
    }
  ]
}
```

Code 5: .vscode/launch.json

윈도우 사용 시 변경사항: 코드 4, 5를 윈도에서 사용할 때는 경로들의 `/`을 `\\`로, `hw3`를 `hw3.exe`로 변경해야 합니다.

# 7 템플릿 코드

```
#ifndef ARRAY_BAG_H
#define ARRAY_BAG_H
#include <iostream>
#include <vector>

template <typename T>
class ArrayBag {
 private:
  static const int DEFAULT_CAPACITY = 100;
  T items[DEFAULT_CAPACITY];
  int item_cnt;
  int max_items;
  int get_index_of(const T&) const;

 public:
  ArrayBag();
  ArrayBag(int);
  int get_current_size() const;
  bool is_empty() const;
  bool add(const T&);
  bool remove(const T&);
  void clear();
  bool contains(const T&) const;
  int get_frequency_of(const T&) const;
  std::vector<T> to_vector() const;
  template <typename U>
  friend std::ostream& operator<<(std::ostream&, const ArrayBag<U>&);
  template <typename U>
  friend ArrayBag<U> union_bags(const ArrayBag<U>&, const ArrayBag<U>&);
  template <typename U>
  friend ArrayBag<U> intersect_bags(const ArrayBag<U>&, const ArrayBag<U>&);
  ArrayBag<T> difference(const ArrayBag<T>&) const;
};

template <typename T>
ArrayBag<T> union_bags(const ArrayBag<T>&, const ArrayBag<T>&);
template <typename T>
ArrayBag<T> intersect_bags(const ArrayBag<T>&, const ArrayBag<T>&);

#include "ArrayBag.cpp"
#endif
```

Code 6: ArrayBag.h

```
#include <algorithm>
#include <iostream>
#include <vector>

#include "ArrayBag.h"

template <typename T>
ArrayBag<T>::ArrayBag() : item_cnt(0), max_items(DEFAULT_CAPACITY) {}

template <typename T>
ArrayBag<T>::ArrayBag(int capacity) : item_cnt(0), max_items(capacity) {}

template <typename T>
int ArrayBag<T>::get_current_size() const {
}
template <typename T>
bool ArrayBag<T>::is_empty() const {
```

```
}
template <typename T>
bool ArrayBag<T>::add(const T& newEntry) {

}
template <typename T>
bool ArrayBag<T>::remove(const T& anEntry) {

}
template <typename T>
void ArrayBag<T>::clear() {
}
template <typename T>
bool ArrayBag<T>::contains(const T& anEntry) const {
}
template <typename T>
int ArrayBag<T>::get_frequency_of(const T& anEntry) const {

}
template <typename T>
std::vector<T> ArrayBag<T>::to_vector() const {

}
template <typename T>
int ArrayBag<T>::get_index_of(const T& target) const {

}

template <typename U>
std::ostream& operator<<(std::ostream& os, const ArrayBag<U>& bag) {
  std::vector<U> items(bag.to_vector());
  sort(items.begin(), items.end());
  for (const U& item : items) {
    os << item << " ";
  }
  return os;
}

template <typename T>
ArrayBag<T> union_bags(const ArrayBag<T>& bag1, const ArrayBag<T>& bag2) {
}

template <typename T>
ArrayBag<T> intersect_bags(const ArrayBag<T>& bag1, const ArrayBag<T>& bag2) {
}

template <typename T>
ArrayBag<T> ArrayBag<T>::difference(const ArrayBag<T>& another_bag) const {
}
```

Code 7: ArrayBag.cpp

```
#include <iostream>

#include "ArrayBag.h"

int main(int argc, char* argv[]) {
  int T;
  std::cin >> T;

  for (int t = 0; t < T; ++t) {
    int N, M;
    std::cin >> N >> M;
```

```
    ArrayBag<int> A(N);
    ArrayBag<int> B(M);

    for (int i = 0; i < N; ++i) {
      int element;
      std::cin >> element;
      A.add(element);
    }

    for (int i = 0; i < M; ++i) {
      int element;
      std::cin >> element;
      B.add(element);
    }

    std::cout << union_bags(A, B) << std::endl;
    std::cout << intersect_bags(A, B) << std::endl;
    std::cout << A.difference(B) << std::endl;
    std::cout << B.difference(A) << std::endl;
  }

  return 0;
}
```

Code 8: ArrayBag.cpp