12. 템플릿과 STL

[실습 12-1]

- 다음 함수는 매개변수로 주어진 int 배열 src에서 배열 minus에 들어있는 같은 정수를 모두 삭제한 새로운 int 배열을 동적으로 할당받아 리턴한다. retSize는 remove() 함수의 실행결과를 리턴하는 배열의 크기를 전달받는다.

```
int* remove(int src[], int sizeSrc, int minus[], int sizeMinus, int& retSize);
```

- 템플릿을 이용하여 remove()를 일반화시키고, double로 구체화하는 경우 어떻게 remove() 함수를 호출해야 하는지 뒷장의 main() 함수에 추가해 보시오.

[실습 12-1] 계속

```
int main() {
   // remove() 함수를 int로 구체화하는 경우
   cout << "정수 배열 {1,2,3,4}에서 정수 배열 {-3,5,10,1,2,3}을 뺍니다" << endl;

   int x[]={1,2,3,4};
   int y[]={-3,5,10,1,2,3};
   int retSize;

   _________ p = remove(x, 4, y, 6, retSize);
   if(retSize == 0) {
      cout << "모두 제거되어 리턴하는 배열이 없습니다." << endl;
      return 0;
   }
   else {
      for(int i=0; i<_________________; i++)    // 배열의 모든 원소 출력
         cout << _____________________ << ' ';
      cout << endl;

      delete [] p; // 할당받은 배열 반환
   }

   // remove() 함수를 double로 구체화하는 경우
   // 이곳에 작성

}
```

[실습 12-2]

- map 컨테이너를 이용하여 (영어, 한글) 단어를 쌍으로 저장하고, 영어로 한글을 검색하는 사전을 작성하는 아래 프로그램을 완성하시오.

```
int main() {
    ____________________dic; // 맵 컨테이너 생성. 키는 영어 단어, 값은 한글 단어

    // 단어 3개를 map에 저장
    dic.insert(make_pair("love", "사랑")); // ("love", "사랑") 저장
    dic.insert(make_pair("apple", "사과")); // ("apple", "사과") 저장
    dic["cherry"] = "체리"; // ("cherry", "체리") 저장

    cout << "저장된 단어 개수 " << ____________________<< endl;

    string eng;
    while (true) {
        cout << "찾고 싶은 단어>> ";
        ____________________// 사용자로부터 키 입력
        if (eng == "exit")
            break;  // "exit"이 입력되면 종료

        if(____________________) // eng '키'를 끝까지 찾았는데 없음
            cout << "없음" << endl;
        else
            cout << ____________________ << endl; // dic에서 eng의 값을 찾아 출력
    }
    cout << "종료합니다..." << endl;
}
```

```
저장된 단어 개수 3
찾고 싶은 단어>> apple
사과
찾고 싶은 단어>> lov
없음
찾고 싶은 단어>> love
사랑
찾고 싶은 단어>> exit
종료합니다...
```

참고 : map 사용하기

- 특징
  - ('키', '값')의 쌍을 원소로 저장하는 제네릭 컨테이너
    - 동일한 '키'를 가진 원소가 중복 저장되면 오류 발생
  - '키'로 '값' 검색
  - 많은 응용에서 필요함
  - **#include <map>** 필요
- 맵 컨테이너 생성 예
  - 영한 사전을 저장하기 위한 맵 컨테이너 생성 및 활용
    - 영어 단어와 한글 단어를 쌍으로 저장하고, 영어 단어로 검색

```
// 맵 생성
map<string, string> dic;                    // 키는 영어 단어, 값은 한글 단어

// 원소 저장
dic.insert(make_pair("love", "사랑"));      // ("love", "사랑") 저장
dic["love"] = "사랑";                       // ("love", "사랑") 저장

// 원소 검색
string kor = dic["love"];                   // kor은 "사랑"
string kor = dic.at("love");                // kor은 "사랑"
```

map 클래스의 주요 멤버와 연산자

멤버와 연산자 함수	설명
insert(pair<> &element)	맵에 '키'와 '값'으로 구성된 pair 객체 element 삽입
at(key_type& key)	맵에서 '키' 값에 해당하는 '값' 리턴
begin()	맵의 첫 번째 원소에 대한 참조 리턴
end()	맵의 끝(마지막 원소 다음)을 가리키는 참조 리턴
empty()	맵이 비어 있으면 true 리턴
find(key_type& key)	맵에서 '키' 값에 해당하는 원소를 가리키는 iterator 리턴
erase(iterator it)	맵에서 it가 가리키는 원소 삭제
size()	맵에 들어 있는 원소의 개수 리턴
operator[key_type& key]()	맵에서 '키' 값에 해당하는 원소를 찾아 '값' 리턴
operator=()	맵 치환(복사)