1. (**실습**: 순차탐색) 다음 코드의 search 함수는 배열 v 내에 key 값의 존재하는 경우 그 위치 번호를, 그렇지 않은 경우 -1을 반환한다고 한다. 순차탐색 방식으로 search 함수를 완성하시오. search 함수의 시간복잡도는 얼마인가?

   - **실습** #1: 배열 v에 대한 순차탐색의 간단한 구현은 0부터 v.length-1까지 범위의 각 i에 대해 v[i]를 탐색키와 비교하는 작업을, v[i]가 탐색키와 일치할 때까지, 반복하는 방식을 사용한다. (Reference: https://en.wikipedia.org/wiki/Linear_search, CC-BY-SA)

   - **실습** #2: 실습 #1의 구현은 각 i에 대해 i 값의 범위 검사를 수행하므로 각 i에 대해 2회 비교(i의 범위 검사 및 탐색키와 v[i]의 일치 검사)를 요구한다. v[v.length] 위치가 사용가능하다고 가정하면 v[v.length]에 탐색키를 저장해 둔 다음, i에 대한 범위 검사 없이 순차탐색을 구현할 수 있다. 즉 탐색키가 발견된 경우 i의 값이 v.length이면 탐색키가 존재하지 않는 것이므로 -1을 반환하고, 그렇지 않다면 i를 반환하면 된다. 그러나 자바에서 v[v.length] 위치는 접근 불가이므로 이러한 구현에 약간의 변형 처리가 추가되어야 한다. 이를 구현하시오. (Reference: https://en.wikipedia.org/wiki/Linear_search, CC-BY-SA)

```
public class Test {
	public static void main(String[] args) {
		int		v[]=new int[1000000];
		Random	random=new Random();
		for (int i = 0; i < v.length; i++) v[i]=random.nextInt(1000000);
		int		key=1234;
		System.out.println(search(v, key));
	}
	private static int search(int[] v, int key) {
	}
}
```

2. (정렬된 배열에 대한 이진탐색: 자바 클래스 활용) 다음은 배열 내 자료를 정렬한 후 배열 내 임의 자료를 이진탐색하는 코드의 예로, 이진탐색을 위해 자바 클래스 Arrays의 binarySearch 메소드를 사용하고 있다.

   - 실습: 아래 코드에서 key 값으로 배열 내에 존재하는 값과 그렇지 않은 값들을 입력하여 실행해 보시오.

```
public class Test {
	public static void main(String[] args) {
		int		v[]= {2,3,6,8,9};
		Arrays.sort(v); // 배열 정렬
		int		key=8;
		int		index=Arrays.binarySearch(v, key); // 정렬된 배열 내 자료 위치 이진 탐색
		if(index<0) System.out.println("배열 내 자료 없음");
		else System.out.println("자료 발견 위치(배열 내 index): "+index);
	}
}
```

3. (**실습**: 정렬된 배열에 대한 이진탐색: 이진탐색함수 구현) 다음은 배열 내 자료를 정렬한 후 배열 내 임의 자료를 이진탐색하는 코드의 예로, 이진탐색을 위해 직접 구현된 binarySearch 메소드를 호출하고 있다. (참조: https://en.wikipedia.org/wiki/Binary_search_algorithm, CC-BY-SA)

- **실습** #1: 아래 binarySearch 함수에서 int m=(left+right)/2; 문장은 left+right가 int형 정수의 최대값을 초과할 경우 문제가 발생한다. 이를 해결하시오.

- **실습** #2: 아래 binarySearch 함수를 재귀적 방법으로 구현하시오.

```
public class Test {
	public static void main(String[] args) {
		int	v[]= {2,3,6,8,9};
		Arrays.sort(v); // 배열 정렬
		int	key=8;
		int	index=binarySearch(v, key); // 정렬된 배열 내 자료 위치 이진 탐색
		if(index<0) System.out.println("배열 내 자료 없음");
		else System.out.println("자료 발견 위치(배열 내 index): "+index);
	}
	private static int binarySearch(int[] v, int key) {
		int	left=0, right=v.length-1;
		while(left<=right){
			int	m=(left+right)/2;
			if(key==v[m]) return m;
			if(key<v[m]) right=m-1;
			else left=m+1;
		}
		return -1;
	}
}
```

References

- C로 쓴 자료구조론 (Fundamentals of Data Structures in C, Horowitz et al.). 이석호 역. 사이텍 미디어. 1993.
- 쉽게 배우는 알고리즘: 관계 중심의 사고법. 문병로. 한빛아카데미. 2013.
- C언어로 쉽게 풀어 쓴 자료구조. 천인국 외 2인. 생능출판사. 2017.
- 김윤명. (2008). 뇌를 자극하는 Java 프로그래밍. 한빛미디어.
- 남궁성. 자바의 정석. 도우출판.
- 김윤명. (2010). 뇌를 자극하는 JSP & Servlet. 한빛미디어.