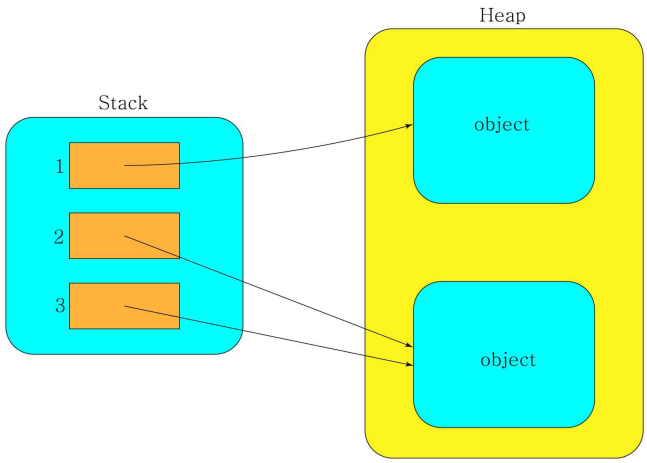
9/25일 튜터링 질문 정리

1. ‘참조 타입 변수’에 대한 질문이 들어왔습니다. 먼저, Java의 경우, C와 달리 포인터를 사용하지 않기에 주소에 대한 부담감은 없으시겠지만, 언어 내부적으로는 주소를 다루기는 합니다. 그렇기에 주소에 대한 말을 안할 수가 없기에 언급을 하도록 하겠습니다.

강의 자료 10p에 있는 그림을 보면 Stack 영역에 객체를 가리키는 변수가 존재하며, 객체는 Heap에 있음을 알 수 있습니다.

강의 자료의 그림을 그대로 가져오면 저작권 문제가 생기기에, 제가 따로 그림을 그려 설명하도록 하겠습니다.



1번 변수는 위 객체를 참조하고, 2, 3번 변수는 아래 객체를 참조하는 그림입니다. 즉, 참조한다는 말 자체가 **주소를 가리킨다는 의미**입니다. 물론, C언어의 포인터처럼 어려운 개념은 아니기에 와닿지는 않겠지만 해당 변수가 그 객체 자체가 아님을 알아야 합니다.

2번 변수와 3번 변수는 같은 객체를 참조하기에, 비교 연산자를 이용하면 true 값을 반환할 것입니다.

2. ‘메서드 시그니처’에 대한 질문이 들어왔습니다.
사실, 이 부분은 많이 사용되는 용어는 아니기에, 예민하게 반응하시지는 않으셔도 되나, 뒤에 용어들이 쏟아질 때 이런 용어들이 헷갈릴 수 있을 것입니다. 한번은 정리하시는게 좋을 것 같아. 저도 정리해드리겠습니다.

메서드 시그니처, 오버로딩
메서드 이름, 매개변수의 개수, 데이터 타입, 순서를 의미한다고 합니다. 오버로딩은 아직 다룬 개념은 아니지만 궁금해하시는 분들도 있을 것 같기에 작년에 제가 짠 함수로 설명을 진행하도록 하겠습니다.

```
public class QuickSortOverload {

    // 기본 퀵소트 메서드 (오버로딩)
    public static void quickSort(int[] arr) {
        quickSort(arr, 0, arr.length - 1); // 전체 배열을 정렬
    }

    // 부분 배열에 대해 퀵소트를 수행하는 메서드
    public static void quickSort(int[] arr, int low, int high) {
        if (low < high) {
            int pi = partition(arr, low, high);
            quickSort(arr, low, pi - 1); // 왼쪽 부분 정렬
            quickSort(arr, pi + 1, high); // 오른쪽 부분 정렬
        }
    }

    // 파티션 메서드
    public static int partition(int[] arr, int low, int high) {
        int pivot = arr[high];
        int i = (low - 1);
        for (int j = low; j < high; j++) {
            if (arr[j] <= pivot) {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        int temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;

        return i + 1;
    }

    // 배열을 출력하는 메서드 (오버로딩)
    public static void printArray(int[] arr) {
        for (int i : arr) {
            System.out.print(i + " ");
```

```
        }
        System.out.println();
    }

    // 오버로딩된 printArray (배열의 일부분만 출력)
    public static void printArray(int[] arr, int low, int high) {
        for (int i = low; i <= high; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    public static void main(String[] args) {
        int[] arr = {10, 7, 8, 9, 1, 5};

        System.out.println("주어진 배열:");
        printArray(arr);  // 전체 배열 출력

        quickSort(arr);  // 전체 배열 정렬

        System.out.println("정렬된 배열:");
        printArray(arr);  // 전체 배열 출력

        System.out.println("일부 배열(1번 인덱스부터 4번 인덱스까지) 출력:");
        printArray(arr, 1, 4);  // 배열의 일부 출력
    }
}
```

위의 코드를 보시면, 오버로딩이라고 언급이 되어있습니다. C언어에서와 달리, 같은 함수 이름을 사용할 수 있습니다. C언어에서 만약 add함수를 만든다면 어떻게 할 수 있을까요? int형과 double형을 구분하여 2개의 함수를 만들어야 합니다. 즉, 함수 이름부터 add_int, add_double과 같이 아예 다르게 만들어야 하는 것입니다. 하지만 Java부터는 함수 이름을 같은 이름으로 사용해도 아무런 문제가 되지 않습니다. 이 개념을 오버로딩이라 하며, 추후 배우시게 될 객체지향 프로그래밍에서, 오버 라이딩과 헷갈리실 수 있으니, 정확히 개념을 잡고 가셔야 합니다.