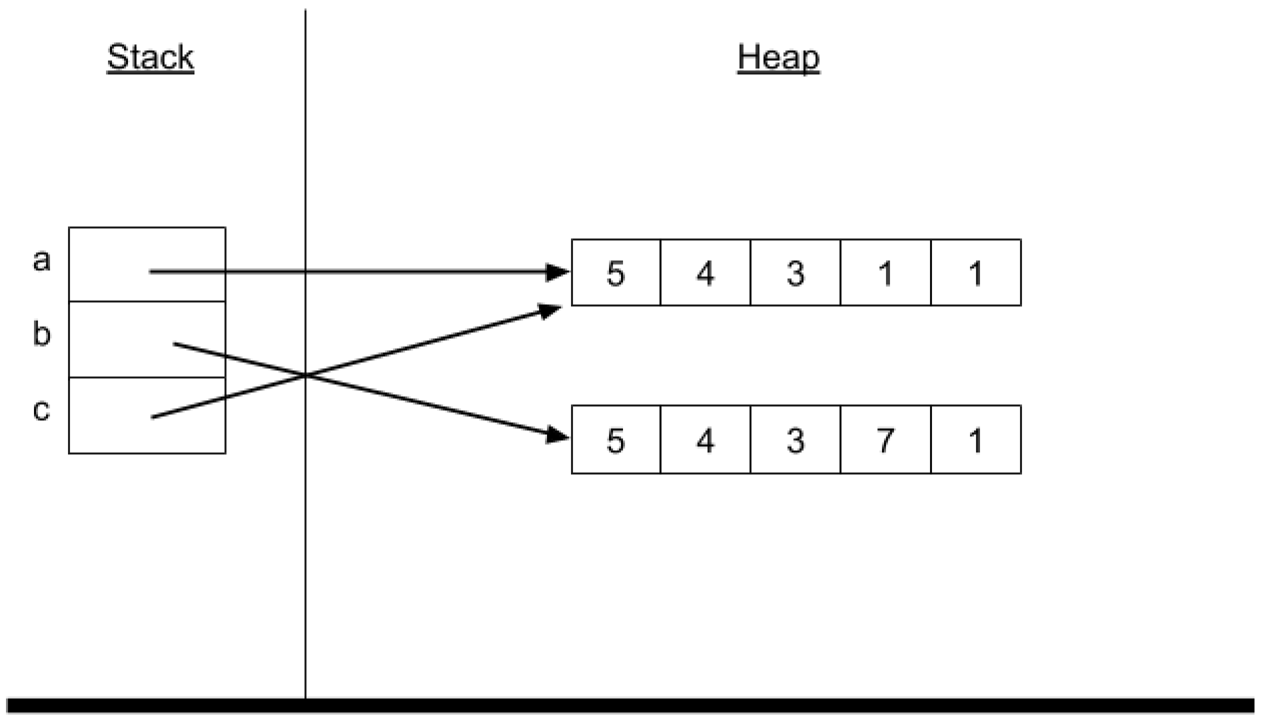
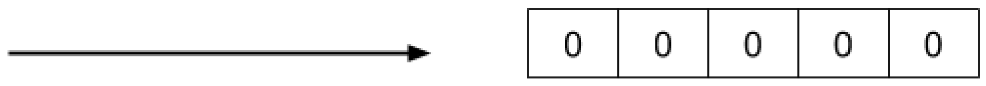
**Problem Set 1, Part I**

**Problem 1: Memory management and arrays**

**1-1)**





**1-2) “1 7 1”**

**Problem 2: Array practice**

**2-1)**

```
public static void shiftRight(int[] arr) {

   if(arr == null) {

      throw new IllegalArgumentException();

   } else if(arr.length <= 1){

      return;

   } else {

       int temp = arr[0];

       arr[0] =  arr[arr.length - 1];

      for(int i = 0; i < arr.length - 1; i++){

            if(i == 0){

              arr[i + 1] = temp;

            } else {

              arr[i+1] = arr[i];

            }
      }
   }
}
```

**2-2)**

```
public static int indexOf(int[] arr1, int[] arr2) {

     int [] indexes = new int[arr2.length];
     int currentIndex = 0;

      for(int i = 0; i < arr2.length; i++){

           if(arr2[i] != arr1[0]){
              continue;
           } else {
              indexes[currentIndex] = i;
              currentIndex++;
           }
      }

      for(int j = 0; j < currentIndex; j++){

           int matches = 0;

           for(int k = indexes[j]; k < indexes[j] + arr1.length; k++){

                if(arr1[k] == arr2[k]){
                     matches += 1;
                }
           }

           if(matches == arr1.length){
                return indexes[j];
           }
     }

      return -1;

}
```

**Problem 3: Recursion and the runtime stack**

**3-1)**

```
mystery(5, 6)
-------------
     a = 5
     b = 6
     myst_rest = mystery(4, 4)

          mystery(4, 4)
          -----------------
          a = 4
          b = 4
          myst_rest = mystery(3, 2)

               mystery(3, 2)
               -----------------
               a = 3
               b = 2
               myst_rest = mystery(2, 0)

                    mystery(2, 0)
                    -----------------
                    a = 2
                    b = 0

                         return 2 which is assigned to myst_rest

                    mystery(3, 2) returns 2 + myst_rest = 4

               mystery(4, 4) returns 4 + myst_rest = 8

    mystery(5, 6) returns 6 + myst_rest = 14 and the method exits
```

**3-2) It returns 14**

**3-3) 4 stack frames**

**3-4) Passing negative numbers for both a and b will result in never reaching the base case and hence infinite recursion. Example, mystery(-1, -1)**

## Problem 4: Rewriting a method

**4-1)**

```
public static boolean search(Object item, Object [] arr) {
      for (int i = 0; i < arr.length; i++) {
        if (arr[i].equals(item)) {
            return true;
        }
    }
    return false;
}
```

**4-2)**

```
public static boolean search(Object item, Object [] arr, int start) {

      if(arr[start].equals(item)){
            return true;
      } else if(start == arr.length){
            return false;
      }

      return search(item, arr, start + 1);
}
```