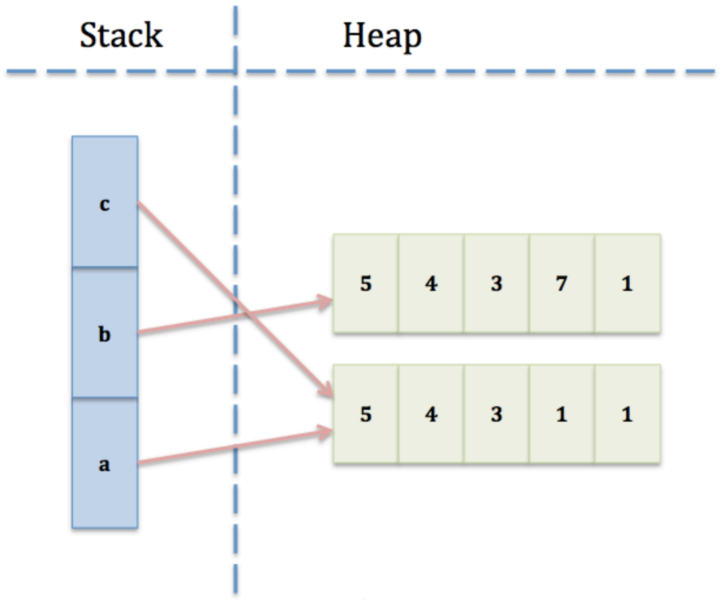
Nicolas Rannou

Problem Set 1

1-Memory management and arrays (5 points)

a) Memory diagram: (3 points)



b) Printed message: (2 points)

The printed message is: “1 7 1”

2-Array practice (10 points)

a) shiftRight() (5 points)

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

  if( (arr == null) || (arr.length < 2) ){
    // don’t do anything
    return;
  }

  // keep value of the first integer
  Integer firstValue = arr[arr.length-1];

  for(Integer i=arr.length-1; i>0; --i){
     arr[i] = arr[i-1];
  }
  // put first integer at last position
  arr[0] = firstValue;
  }
```

b) indexOf() (5 points)

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

  if( arr1 == null || arr2 == null){
    // don't do anything
    return -1;
    }

  // go through array2, minus size of array1 + 1
  for(Integer i=0; i<arr2.length - arr1.length  + 1; ++i){
    Boolean match = true;
    Integer j = 0;
    while(match && j < arr1.length){
      // if no match, no need to compare anymore, match == false so we exit the while
      if(arr2[i+j] != arr1[j]){
        match = false;
        }
      j++;
      }

    // if match == true, we found a solution!
    if(match){
      return i;
    }
  }
  return -1;
  }
```

<u>**3-Recursion and the runtime stack (10 points)**</u>

a) Trace (3 points)

```
main() calls mystery(5,6)
        mystery(5,6) calls mystery(4,4)
                mystery(4,4) calls mystery(3,2)
                        mystery(3,2) calls mystery(2,0)
                                mystery(2,0) returns 2
                        mystery(3,2) returns 2 + 2
                mystery(4,4) returns 4 + 4
        mystery(5,6) returns 6 + 8
main()
```

b) Returned value (2 points)

mystery(5,6) returns **14.**

c) # of method frames on the stack when base case reached (2 points)

4 method frames are in the stack when the base case is reached.

d) Infinite recursion (3 points)

If **a<0** and **b<0**, a*b will never be equal to 0(*) then mystery(a,b) will produce infinite recursion.
(*): mystery(a,b) will call mystery(a-1,b-2), mystery(a-1, b-2) will call mystery(a-2, b-4), etc.

4-Rewriting a method (10 points)

a) Any type of array (4 points)

```
public static Boolean search(Object item, Object[] arr){

  for(Integer i=0;i<arr.length; i++){
    if(arr[i] == item){
      return true;
    }
  }
  return false;
  }
```

b) Recursion instead of iteration (6 points)

```
public static Boolean recursive_search(Object item, Object[] arr, Integer i){
  // base case
  if(i == arr.length){
    return false;
  }

  if(arr[i] == item){
      return true;
    }

  return recursive_search(item, arr, i+1);
  }
```