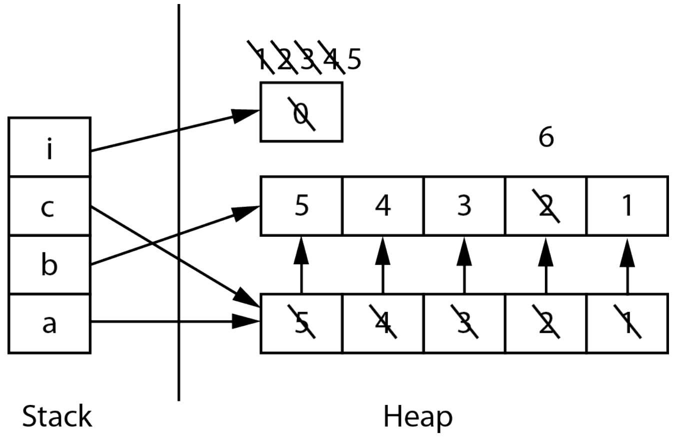
Computer Science E-119

Problem Set 1

Graham Schmidt
Date: September 21, 2011
Harvard ID: 30825489

1. Memory management and arrays

a)



b) 6 6 6

2. Array Practice

a)

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

    int num_shift = 1;

    int[] arr_shift = new int[arr.length];

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

        // Shift to front
        if(i == 0) {
            arr_shift[i] = arr[arr.length - 1];
        } else {
            arr_shift[i] = arr[i - 1];
        }
    }

    return;
}
```

b)

```
    public static int indexOf(int[] list1, int[] list2) {

        // Initialize counter for matching elements and
        // determining longest element to match
        int loop_hits = 0;
        int target_hits = list1.length;
        int loop_count = ((list1.length > list2.length)
                            ? list1.length
                            : list2.length);

        // Store the first element to start checking for hits
        int first_element = list1[0];
        boolean found = false;
        int index = 0;

        // Loop through longest array
        for(int i = 0; i < loop_count; i++) {

            // First element in compared array must hit
            // Capture the index and set a flag to true
            if(list2[i] == first_element) {
                found = true;
                index = i;
            }

            // Update the index for the comparing array
            int j = i - index;

            // If the first element is found, start comparing
            // the next elements until the number of hits
            // equals the length of the compared array
            if(found) {
                if(list2[i] == list1[j]) {
                    loop_hits++;

                    // Return the index if it matches all elements
                    if(loop_hits == target_hits) {
                        return index;
                    }
                } else {
                    // Reset
                    loop_hits = 0;
                    found = false;
                }
            }
        }

        return (-1);
    }
```

3. a)

```
mystery(5, 6)
      6 + mystery(4, 4)
            4 + mystery(3, 2)
                  3 + mystery(2, 0)
                        2
```

b) 15

c) 9 stack frames

d) mystery(-1, 1)

If ‘a’ is negative then it will recurse infinitely.

4. a)

```
    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;
    }
```

b)

```
public static boolean search(Object item, Object[] arr, int position) {
        if(position < arr.length) {
            if(arr[position].equals(item)) {
                return true;
            } else {
                return search(item, arr, position + 1);
            }
        }
        return false;
    }
```