CS106B Instructor: Cynthia Lee
Autumn 2017 November 2, 2017

# MIDTERM EXAM - SOLUTIONS

## 1. ADTs

(a) {4, 7, 3, 2, 6, 5}  {1, 5, 9, 13, 11, 7, 3}

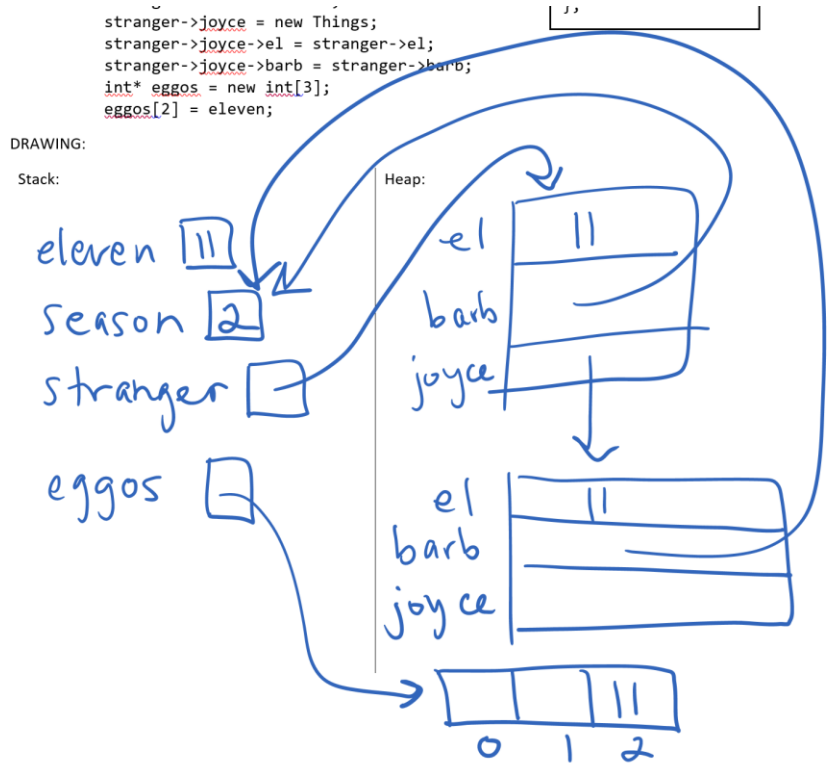
(b) Like any coding problem, there are multiple ways of approaching the solution. Here are two, but others are possible.

```
/* OPTION #1: iterates over sticker and places each pixel where it should go on the
   Background, if that will not be out of bounds */
void rotateAndPlaceImg(Grid<int>& background, Grid<int>& sticker, int row, int col) {
    for (int r = 0; r < sticker.numRows(); r++) {
        for (int c = 0; c < sticker.numCols(); c++) {
            int bgRow = row + sticker.numCols() - 1 - c;
            int bgCol = col + r;
            if (background.inBounds(bgRow, bgCol)) {
                background[bgRow][bgCol] = sticker[r][c];
            }
        }
    }
}
```

```
/* OPTION #2: makes a temporary grid to hold the rotated sticker (so row-col dimensions
   are swapped), then places temp grid on the background within bounds */
void rotateAndPlaceImg(Grid<int>& background, const Grid<int>& sticker, int row, int
col) {
    Grid<int> toPlace(sticker.numCols(), sticker.numRows());
    for(int nRow = 0; nRow < toPlace.numRows(); nRow++) {
        for (int nCol = 0; nCol < toPlace.numCols(); nCol++) {
            toPlace[nRow][nCol] = sticker[nCol][toPlace.numRows() - 1 - nRow];
        }
    }

    for(int newRow = row; newRow < toPlace.numRows(); newRow++) {
        for(int newCol = col; newCol < toPlace.numCols(); newCol++) {
            if (background.inBounds(row + newRow, col + newCol)) {
                background[row + newRow][col + newCol] = toPlace[newRow][newCol];
            }
        }
    }
}
```

## 2. Memory Diagram




Notes:
-- Make a box for pointers to hold the beginning of the arrow. This makes it clear where the space for the storage of the memory address is located.
-- Remember that the only way something ends up on the heap is as the result of a call to "new."
-- You do NOT need to show separate local variables on the stack "attached" to each other, but you must show structs and arrays (stack or heap) as adjacent/attached boxes.
-- Array pointers should point to 0th element of the array.

(we didn't deduct for some of these notes items, but please keep in mind for the future)

## 3. Recursion

```
bool gossipCheck(string gossiper, string victim, int maxDist,
                 Map<string, Vector<string>>& friendships) {
    if (maxDist <= 0) {
        return false;
    }
    if (gossiper == victim) {
        return true;
    }
    for (string friendship : friendships[gossiper]) {
        if (gossipCheck(friendship, victim, maxDist - 1, friendships)) {
            return true;
        }
    }
    return false;
}
```

4. **Link Nodes**

```
ListNode *ptr2 = ptr->next->next->next; // ptr2->6
ptr2->next = ptr->next->next; // 6->5
ptr2->next->next = ptr; // 5->3
delete ptr->next; // delete 4
ptr->next = NULL; //3->NULL
ptr = ptr2; // ptr->6
```

---

5. **Big O**

   $O(N^2)$
   $O(N^2)$
   $O(N^2)$
   $O(N)$

---