APS105 Lecture 22
8.5

Dan Zingaro

March 17, 2010

Are Clickers Working?

- I'd like to study the use of clickers in this class and determine how useful they are
- You may volunteer to help (it is completely optional!)
- What does volunteering mean?
  - Allowing me to anonymously include your clicker responses in my data analysis
  - Providing some feedback by responding to a quick survey at the end of the course
- To volunteer: read, sign, and return the consent form
- If you do not wish to volunteer: do nothing, or return a blank form

Feedback from Reading Quiz

- Some students asked me to demo the anagram lab
- I'll do that quickly . . . but there's much more time in lab for this!

Deleting Blobs

```
*       *****
****
*

***
* *
***
```

- Two asterisks are connected if they are touching vertically or horizontally
- A group of connected asterisks is called a blob
- There are three blobs in this array
- Problem: starting at any point in a blob, erase the entire blob

Deleting Blobs...

```
*
****
*
```

- We can start by erasing the asterisk at the top left
- Then, we have no choice; we must move down, and erase that asterisk

```
-
****
*
```

Figure: Erase Top

```
-
-***
*
```

Figure: Move Down and Erase

Deleting Blobs...

```
-
-***
*
```

- Now, we're at a choice point: we can go right or down
- Let's move right
- We must make a note telling us that we must come back here and take the down path later

```
-
--**
*
```

Erasing Blobs Recursively

```
void eraseBlob (char grid[][COLS], int x, int y)
```

- Instead of trying to write code to remember each choice point we have reached, we can think of erasing blobs recursively
- `grid` is a two-dimensional array representing the grid
- We assume the borders of the grid are blanks
- `x` and `y` are the coordinates from which we want to start erasing

Erasing Blobs Recursively...

- Base case: if `grid[x][y]` is a blank, we have nothing to do
- Otherwise, we erase the asterisk at `grid[x][y]`
- We think of `grid[x][y]` as a choice point with four choices (some of which will immediately lead to blanks)
- There are four subproblems to solve
  - Erase the blob to the left
  - Erase the blob to the right
  - Erase the blob above
  - Erase the blob below

Erasing Blobs Recursively...

```
void eraseBlob (char grid[][COLS], int x, int y) {
  if (grid[x][y] == ' ')
    return;
  else {
    grid[x][y] = ' ';
    eraseBlob (grid, x-1, y);
    eraseBlob (grid, x+1, y);
    eraseBlob (grid, x, y - 1);
    eraseBlob (grid, x, y + 1);
  }
}
```

ConcepTest

We would like to count the total number of asterisks in the blob containing location `(x, y)`. Which of the following is the most appropriate base case for this task?

- A. If `(x, y)` is a blank, the total number of asterisks is 0.
- B. If `(x, y)` is a blank, the total number of asterisks is 1.
- C. If `(x, y)` is an asterisk, the total number of asterisks is 0.
- D. If `(x, y)` is an asterisk, the total number of asterisks is 1.

ConcepTest

The following code template counts the total number of asterisks reachable from location `(x, y)`. Select the code to use in place of the comment.

```
int blobSize (char grid[][COLS], int x, int y) {
  int left, right, up, down;
  if (grid[x][y] == ' ')
    return 0;
  else {
    grid[x][y] = ' ';
    left = blobSize (grid, x, y-1);
    right = blobSize (grid, x, y + 1);
    up = blobSize (grid, x - 1, y);
    down = blobSize (grid, x + 1, y);
    return ... // fill in the code
  }
}
```

- A. `1 + left + right + up + down;`
- B. `left + right + up + down;`
- C. `4 + right + left + up + down;`
- D. `right + left + up + down;`

ConcepTest

We've commented out some code. Is the solution still correct?

```
int blobSize (char grid[][COLS], int x, int y) {
  int left, right, up, down;
  if (grid[x][y] == ' ')
    return 0;
  else {
    //grid[x][y] = ' ';
    left = blobSize (grid, x-1, y);
    right = blobSize (grid, x+1, y);
    up = blobSize (grid, x, y-1);
    down = blobSize (grid, x, y+1);
    return 1 + left + right + up + down;
  }
}
```

- A. No, because the code could loop indefinitely
- B. No, because the grid will now be modified when the function finishes
- C. No, but I don't know why
- D. Yes