Homework 5 - ADTs, Stacks and creating a bubble-pop game

In this assignment, you will do the following:

1. Gain a working knowledge of Abstract Data Types and how they can be implemented in C.
2. A little practice writing and implementing stack-based algorithms.
3. Experience *implementing* an ADT from a given specification.
4. Using your result from (3) to build a relatively playable version of a fun puzzle game.

Due Dates

Part I: Working With ADTs in C - Stack Case Study

But before getting to the game implementation, you need a working knowledge of Abstract Data Types and how they can be supported in C.

In the same directory as this handout, you will find a subdirectory `part-1`. There you will find another handout/worksheet called `Stack-ADT-Exercises`.

For `part-1`, you will do the following:

- Part 1A: Complete the worksheet (simply make a copy and edit it as a google-doc).
- Part 1B: Implement the matching program using stacks as described in the handout.

Part II: "Bubble Pop"

You've probably played puzzle games a lot like this…

The game starts with a grid of colored balloons. The player selects a balloon to pop; if there is at least one other balloon of the same color "connected" to the selected balloon, the selected balloon does indeed pop along with all other balloons of the same color that are "connected" to it. We will call such a collection a "cluster".

The balloons are filled with helium, so if a balloon pops, all balloons *below* it rise up (there is a "ceiling" at the top of the grid).

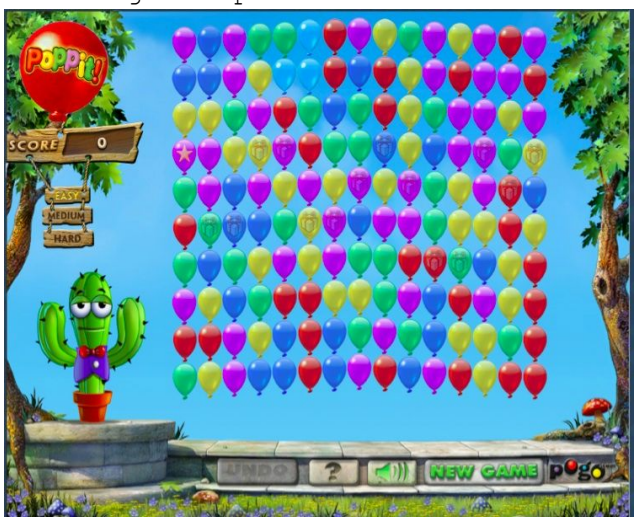
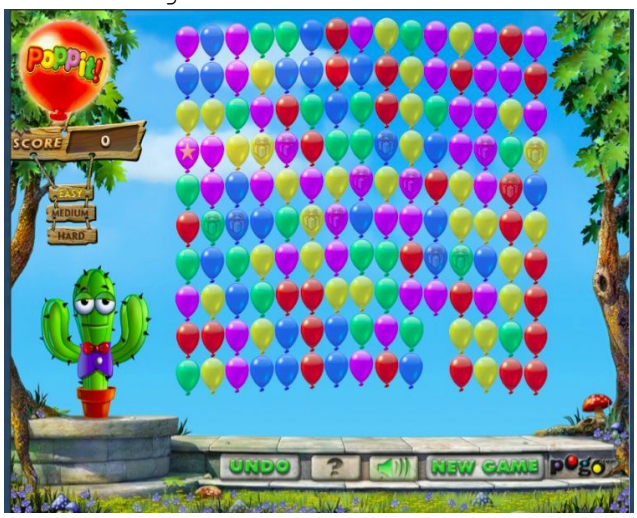
You can play a version of the game here:

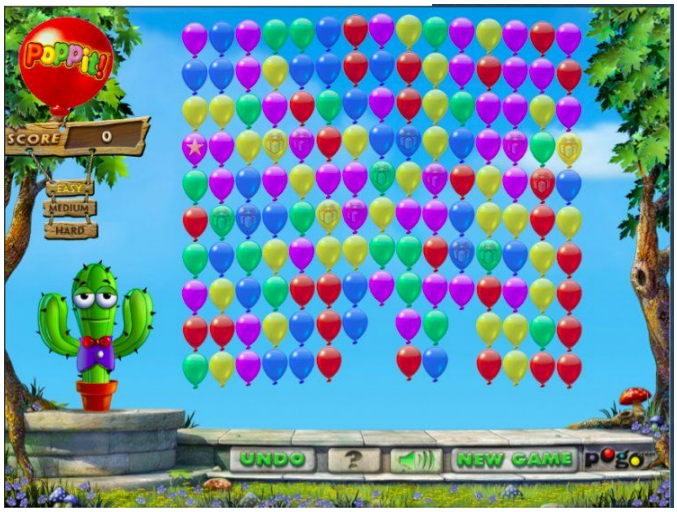
http://poppit.pogo.com/hd/PoppitHD.html

The static screen shots below show a couple of moves.

Scoring: for our version, the player is rewarded for popping larger clusters: if a move results in a cluster of size `n` being popped, the player gets `n(n-1)` added to their score.

The game ends when no clusters remain (the end board may or may not have any remaining balloons).

`select green pair  X` 	`select 3 green  XX` 



Your implementation of the game will be accomplished mainly through completing the implementation of an ADT for which I have given you the specifications (in the form of a header file).

Tasks

1. Complete and test the `BBoard` ADT. This means:
    a. Study the function specifications in `bboard.h`
    b. Study them again.
    c. Decide how you want to represent a board.
    d. Design `struct bboard_struct`. Make sure you understand where it belongs and what code has access to the data elements -- use the `stack` ADT from Part 1 as a model.
    e. Start working on the functions. Adopt the following approach:
        i. Write a function
        ii. Test and debug that function
        iii. Move on to next function