Lab 07 Notes

In this lab, you are given an input file that contains the map of a city (think 2D-array of 4 different characters + whitespaces) and the starting location of a blob that takes over the city. You will write a recursive program to show how the blob spreads through the city, by marking its path with 'B'.

Input File format:
<numRows> <numCols> // the size of the map
<Blob startRow> <Blob startCol> //Starting row/column index of blob
<map characters>

Following is the list of valid characters:

'S' – Street, replaced by 'B' when Blob moves through 'S'
'P' – same as 'S'; also increments private variable total_eaten
'#'—not affected by blob. Keep all '#' as '#' in the output file
'@'—Blob spreads in all directions surrounding a sewer. It will then move to the next sewer on the map, one at a time. You can model this using a nested for loop, every @ encountered on your 2D-map invokes a call to blobbify(). Keep all '@' as '@' in the output file.
' '—space, replace with 'B'.

Blob spreads from one index location to the next in the following order:
Top, right, bottom, left

In other words, assuming that the starting location is (x,y), blob moves in the following order:

```
(x – 1, y);
(x, y +1);
(x +1, y);
(x, y –1);
```

Let's look at our city.h file:

```
#ifndef CITY_H
#define CITY_H

#include <fstream>

class City
{
private:
  char** m_city; // city in which blob will spread, modified as
spreading occurs
  int m_rows, m_cols; // rows and cols of the city
  int m_start_x, m_start_y; // starting position of the blob
  bool m_is_sewers; // has the blob reached the sewers?
 Int total_eaten;
public:
  // Makes the city
  // @param inFile the file that inputs the city information
  City(std::ifstream& inFile);

  //@post deletes m_city
  ~City();

  // Covers all applicable tiles with blobs (B)
  // @post The city array will be modified, as all applicable tiles
will be B
  void blobbifyCity();

  // Blobbifies subsection of city, excluding sewers
  // @param startPos the Position which describes where the blob
starts to spread
  // @post Replaces all applicable tiles in subsection of city
array with Bs
  // @post Modifies m_is_sewers to true if the blob encounters a
sewer
  void blobbify(int start_x, int start_y);

  // Print the city
  // @post Prints the city
  void print();
};
#endif
```

In City.cpp, you would read the input file character-by-character and populate your 2D character array in the constructor.

Once file reading is complete, use the City::print() function to print out the city map:

```
void City::print()
{
  for (int i = 0; i < m_rows; i++)
    {
      for (int j = 0; j < m_cols; j++)
     {
       std::cout << m_city[i][j];
     }
      std::cout << '\n';
    }
  std::cout << '\n';
}
```

In the blobbifyCity() function, you would invoke the recursive blobbify() function and pass the starting position of the blob as argument. You set the m_is_sewers flag in the blobbify() function when @ is encountered.

```
void City::blobbifyCity()
{
  blobbify(m_start_x, m_start_y);
  if (m_is_sewers)
    {
      for (int i = 0; i < m_rows; i++)
     {
       for (int j = 0; j < m_cols; j++)
         {
           if (m_city[i][j] == '@')
          blobbify(i, j);
         }
     }
    }
}
```

Note that you would instantiate your city object in the main.cpp file, like so:

```
City c(std::ifstream& inFile);
```

Sample Input File:

```
8 8
0 0
       #
###### #
@      #
########
### ####
@   #  #
# ######
```

Sample Output File:

```
8 8
0 0


BBBBBBB#
######B#
@BBBBBB#
########
###B####
@BBB#  #
#B######
########
```

Files in your tarball:

```
main.cpp
city.cpp
city.h
Makefile
```