TITLE

AUTHOR
Version
CREATEDATE

Table of Contents

Table of contents

The Great Pumpkin Patch Problem

Input The input to this program will be a number of different gardens. The first line of the input for each garden will be the dimensions of the garden, r, the number of rows in the garden, and c, the number of columns, where 0 â‰¤ r â‰¤ 40 and 0 â‰¤ c â‰¤ 40. Following the dimensions will be r lines with c characters on each line. Each of these characters will be a lower case letter representing the type of gourd grown in the square. A lower case 'p' will represent pumpkins. A garden with 0 for the number of rows and/or columns indicates the end of input and should not be processed. Output For each garden, output the number of the garden (with the first input set being garden 1), the number of pumpkin patches in the garden, and the size of the pumpkin patches in order from smallest to largest. If there is more than one patch of a given size, print the size as many times as it occurs. Use the following format:Garden # 1: 4 patches, sizes: 1 4 8 10 Have a blank line between each line of output.

Author:

Tim Kwist

Version:

1.00

Date:

Wednesday, September 17, 2014

File Index

File List

Here is a list of all documented files with brief descriptions:

File Documentation

pumpkin.cpp File Reference

```
#include <algorithm>
#include <iostream>
#include <fstream>
```

Functions

- int **findTotalPumpkins** (char **pPatch)
  Function headers.
- bool **checkForPatches** (char **pPatch, int &row, int &col)
  Sequentially search through the 2D array of characters to see if there are any pumpkins (represented by 'p' s).
- int **findPatchSize** (char **pPatch, int row, int col, int from)
  Recursively check the size of a pumpkin patch.
- int **main** ()
  Main method implementation.

Variables

- int **patchRows** = 0
  Global Variables.
- int **patchCols** = 0

Function Documentation

bool checkForPatches (char ** *pPatch*, int & *row*, int & *col*)

Sequentially search through the 2D array of characters to see if there are any pumpkins (represented by 'p' s).

If there are, it will return true and the ints passed by reference will contain the coordinates of the pumpkin.

Parameters:

pumpkinPatch	2D array of characters that represents the 'pumpkin patch' to be searched though
row	Blank int passed by reference. If a pumpkin is found, this variable will contain the x coordinate (or row) the pumpkin is in
col	Blank int passed by reference. If a pumpkin is found, this variable will contain the y coordinate (or col) the pumpkin is in

Returns:

True if a 'p' is found in the array. Otherwise, false.

Precondition:

None

Postcondition:

The array will be unchanged. If there is a patch found, the value of row and col will be changed to the coordinates of that pumpkin

int findPatchSize (char ** *pPatch*, int *row*, int *col*, int *from*)

Recursively check the size of a pumpkin patch.

The original coordinate must contain a pumpkin ('p'). Once a pumpkin is found, the character will be changed to a 0 to avoid passing over the same pumpkin. If valid, check for pumpkins in each 4 main directions (north, south, east, west) for more pumpkins. If more are found, call this method with that pumpkin's location.

Parameters:

pumpkinPatch	A 2D array of chars that represents the pumpkin patch
row	The row at which a pumpkin exists
col	The column at which a pumpkin exists
from	The direction from which this method was called. 0: none, 1: South, 2: West, 3: North, 4: East

Returns:

The size(int) of 'p' s in a patch.

Precondition:

A pumpkin('p') exists at pumpkinPatch[row][col]

Postcondition:

All pumpkin's within the patch of the original pumpkin will be changed to 0s in the pumpkinPatch array

int findTotalPumpkins (char ** *pPatch*)

Function headers.

Method implementation.

Sequentially count the number of pumpkins in the given array

Parameters:

pumpkinPatch	2D array of characters that represents the 'pumpkin patch' to be searched though

Returns:

Number of 'p' s in the array

Precondition:

None

Postcondition:

The array will be unchanged.

int main ()

Main method implementation.

Get input from the keyboard and create pumpkin patch

Index

INDEX