**CS 10 - Tinker Percolation**

**Collaboration Policy**

We encourage collaboration on various activities such as lab, codelab, and textbook exercises. However, **no collaboration between students is allowed on the programming assignments**. Please be sure to read and understand our full policy at: [Full Collaboration Policy](https://docs.google.com/document/d/1WyzL3qvKLrC1UCRf178b_wYWQmEZlhDObFNFb79U63I/edit?usp=sharing)

**Submission Instructions**

Submit to [R’Sub](https://galah.cs.ucr.edu) testing, feedback and grading.

**Assignment Specifications**

For this assignment you will simulate the process of percolation. Percolation is used within various science fields to measure the movement or filtering of fluids through a porous material. We will simulate a simplified version of this movement process in a percolation program. Significant pieces of the program have already been written for you!

**You must start by creating a file and copying the provided skeleton code into the file.** Either download it from the Google Drive folder or execute ‘git pull’ in your workspace to grab it from the updated repository. If using git pull, you should not create your assignment in the starter C++ file, you should create your own file.

**Your Assignment**

Your task is to implement the three remaining functions to allow this program to function correctly. You must properly complete the printGrid, percComplete and perc functions. We recommend completing the functions in the order they are listed. **Do not** edit any existing code within the source file. Within this functions you should **utilize the global constants** covered in the documentation and found near the top of the provided source code file.

**Additional Documentation**

We have provided a PDF that contains descriptions of all the functions and some steps to get you started on the implementation of each of the functions you are required to complete. The documentation also contains information about the other functions so that you know how to utilize them if warranted.

**Simulation Examples**

**The simulation is designed to only do one step at a time.** This allows us to see the percolation (if we choose to) as it occurs rather than it all taking place in a single step.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***GOOD*** *(what we want):*  Initial Grid:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  |   Iteration 1:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  |   Iteration 2:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | | Initial Grid:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  |   Iteration 1:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  | | Iteration 2:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  |   Iteration 3:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  | |



***BAD*** *(incorrect implementation, caused by coloring entire path or not using temporary state):*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Initial Grid:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  |   Iteration 1:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | | Initial Grid:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  | | Iteration 1:   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  | |

**Additional Thought**

*Question*: **Why do we need to utilize a transitional state,** TO\_BE\_COLORED**?**

*Understand*: What does the TO\_BE\_COLORED state prevent? (go through algorithm by hand)

*Try it*: Instead of utilizing TO\_BE\_COLORED, use the COLOR state in part one of the function.

**Solution**

**Acquiring the Solution**

*You may have already set this up if you followed the steps when they were released within the Intro to Cloud 9 document. If you have then you can skip to the getting new solutions section.*

We have created a way for you to grab the executable solution programs so that you may run them and experience the results if you have a question as to what should happen if run with a certain set of inputs.

1. Go to Cloud 9:<https://c9.io>
2. Log in to and view your dashboard
3. Click "Create New Workspace"
4. Choose "Clone from URL"
5. Paste following URL in box:<https://github.com/ucrcs010/cs010_programs.git>
6. Click "Create"

A workspace named cs010\_programs should be created. Once inside it, you should be able to navigate to the directory with the program name. Proceed to the execute your program section for instructions on how to execute.

**Getting New Solutions**

If you had previously set up a workspace to execute solution programs in, to acquire new files and executables, once in the generated *cs010\_programs* workspace select the terminal and use the following command:

git pull

**Compiling and Executing**

To compile your C++ file we have to also tell the compiler where the provided functions are**:**

$ g++ percolate.cpp perc\_functions.o -o percolate.out

To execute the program we must provide a grid via input redirection. We have provided three examples for you to utilize but you can easily build several of your own and we recommend doing so to help test your program.

$ run percolate.out 60 < grid\_file\_name.txt

**Frequent Errors**

**What should I do if I have an infinite loop or a program that does not finish?**

After selecting the terminal, we utilize ctrl+cto terminate an active program.

**The color is screwed up on my terminal, what can I do?**

Just reset the terminal by typing the command: reset

**What should I do about my Out of Range issues?**

Make sure you are properly checking your bounds BEFORE any vector accesses.