COMP 201 – Fall 2023

Assignment 2: Heap Management

Zombie Apocalypse

Assigned 1 November 2023 18:00, Due: 22 November 2023 23:59

M. Burak Kizil (*mkizil19@ku.edu.tr*) is the lead person for this assignment.

The Last Stand of Humanity

In the year 2045, a mysterious virus known as the "Zeta Strain" began to spread across the globe. Originating from a remote research facility in the Arctic, the virus had the terrifying ability to reanimate the dead. Within months, major cities were overrun by hordes of the undead, now referred to as "Zombies". Humanity was on the brink of extinction.

However, not all hope was lost. A group of survivors, consisting of scientists, soldiers, and civilians, discovered a potential safe haven: a fortified city known as "Eden", located at the southernmost tip of the world. The city was rumored to be free of the Zeta Strain and was humanity's last hope for survival.

The journey to Eden was perilous. The survivors had to navigate through treacherous terrains, avoid or confront the undead, and most importantly, trust each other. Along the way, they discovered that the Zombies could be defeated if they were outnumbered. Using this knowledge, the survivors formed tight-knit groups, always ensuring they had strength in numbers.

But the Zeta Strain had one more trick up its sleeve. If a human was caught alone by a Zombie, they would be infected and turned into one of them. The journey to Eden became not just a quest for survival, but a test of trust, strategy, and human spirit.

As the survivors approach the gates of Eden, they know that every decision they make could mean the difference between life and death. The fate of humanity rests in their hands.

1 Introduction

In this homework, you will implement a zero-player simulation game following some rules and patterns.

Our goal is to read the given text files which contain the initial grid world with humans' and zombies' locations, simulate the game according to the rules described below, and announce the final statistics which will include three elements below.
1. Humans Survived: A human is said to have survived when s/he reaches the exit at the right-down corner of the grid world, i.e. grid[height-1][width-1].
2. Zombies Dead: When there are at least two people immediately near a zombie, (i.e., right, down, left, up) zombie gets exterminated by people and we clear that cell as empty.
3. Humans Infected: When a person gets caught alone by at least one zombie, s/he gets infected and the program turns that person into a zombie.

Logistics

This is an individual assignment, and all hand-ins are electronic via GitHub. Any clarification or correction will be announced on the Blackboard.

Handout Instructions

How to Start

- Accept the GitHub Classroom assignment using the link:
  `https://classroom.github.com/a/66N`$_6$$tjo$
- Clone the GitHub repository created for you to a Linux machine in which you plan to do your work (Using Linux servers [linuxpool.ku.edu.tr]. See **How to use linuxpool.ku.edu.tr linux servers** section for details.):

  ```
  $ git clone https://github.com/COMP201-Fall2023/assignment-3-USER.git
  ```

  (Replace USER with your GitHub username that you use to accept the assignment)
- **[IMPORTANT]** After cloning the repository, you are required to write the following honor code in a new file called "honor.txt" and commit & push it: "I hereby declare that I have completed this assignment individually, without support from anyone else." You can use the following command to create "honor.txt" file and write the honor code in it:
  `$ echo "I hereby declare that I have completed this assignment individually, without support from anyone else." > honor.txt`
- Then, start filling in the `main.c` file. There is a bare-bones template ready for you to use. If you do not want to use this template, it is fine. The only necessities are clean, not spaghetti code, and the correct outputs.
- You can compile your code with the following code. Use of `-std=gnu99` is **highly** suggested.

  ```
  $ gcc -std=gnu99 -g main.c -o main
  ```

Task

Fill in Your Information

Please fill the comment at the start of the `main.c` file as following:

```
// Write your full name: YOUR_NAME, write your KU ID: YOUR_ID
```

Grid system

1. There are several maps under the maps folder. For each map, first line gives the width (column number) and second line gives the height of the grid (row number).
2. You need to dynamically allocate memory for the grid.
3. Points are zero-indexed from the top left corner. Thus, top left corner is (0, 0), bottom right corner which is also exit corner is (Height-1, Width-1).
4. Dashes (-) stands for empty cells. H stands for humans, and Z for zombies. A cell can only be one of these three elements. During the simulation, you should make necessary adjustments to maintain this condition (i.e. checking for collisions).

Movements

Humans: Humans move first to the right then to the down. You can think it as follows, in the odd steps (1,3,5..) they will try to go right, and in the even steps they will try to go down. **They can only get to their destination if it is empty.**

Zombies: Similar to humans, but they have four movements with the following order: **Right, Down, Left, Up.** Again, they can only move if the destination is empty.

Rules

1. You should first move the humans, then zombies. These movements should be in a dynamically allocated temporary grid. After the movements, check and apply the necessary rules from below.
2. Interactions. **It is important to apply these rules in the following order so that we get the desired simulation.**
- **2.1.** If a zombie is surrounded by at least two people, it dies. Mark that cell as empty.
- **2.2.** If a human is surrounded by at least one zombie, s/he turns into a zombie.

3. After applying these rules, you need to move temporary grid to original grid at each step.

Example Input

```
4
6
H-Z-
--Z-
Z-H-
-H--
H-Z-
----
```

Example Output

```
Humans survived: 2
Humans infected: 3
Zombies died: 0
Final Grid:
- - Z Z
- - - Z
- Z - Z
- - - -
- - - -
- - - -
```

Example

Inspecting this example simulation will help you to understand better of rules. You should call the simulation using following command:

```
./output maps/map_n.txt
```

```
Initial Grid:
H - Z -
- - Z -
Z - H -
- H - -
0
- H - Z
- - - Z
- Z - Z
- - H -
1
- - - Z
- H - Z
- - - -
- Z Z Z
2
- - Z -
- - Z Z
- - - -
Z Z Z -
Humans survived: 0
Humans infected: 3
Zombies died: 0
Final Grid:
- - Z -
- - Z Z
- - - -
Z Z Z -
```

```
Initial Grid:
H - - - -
- - - - -
Z - - - -
- - - - -
- - - - -
0
- H - - -
- - - - -
- Z - - -
- - - - -
- - - - -
1
- - - - -
- H - - -
- - - - -
- Z - - -
- - - - -
2
- - - - -
- - H - -
- - - - -
Z - - - -
- - - - -
3
- - - - -
- - - - -
Z - H - -
- - - - -
- - - - -
4
- - - - -
- - - - -
- Z - H -
- - - - -
- - - - -
5
- - - - -
- - - - -
- - - - -
- Z - H -
- - - - -
6
- - - - -
- - - - -
- - - - -
Z - - - H
- - - - -
7
- - - - -
- - - - -
Z - - - -
- - - - -
- - - - -
Humans survived: 1
Humans infected: 0
Zombies died: 0
Final Grid:
- - - - -
- - - - -
Z - - - -
- - - - -
- - - - -
```

```
Initial Grid:
- - - - -
- H - - -
- Z H - -
- - H - -
- - - - -
0
- - - - -
- - H - -
- - - H -
- - - H -
- - - - -
1
- - - - -
- - - - -
- - H H -
- - - - -
- - - H -
2
- - - - -
- - - - -
- - H - H
- - - - -
- - - - -
3
- - - - -
- - - - -
- - - - -
- - H - H
- - - - -
4
- - - - -
- - - - -
- - - - -
- - - H H
- - - - -
5
- - - - -
- - - - -
- - - - -
- - - - -
- - - H -
6
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Humans survived: 3
Humans infected: 0
Zombies died: 1
Final Grid:
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
```

Evaluation

You are not allowed to use static structs. Your implementation must utilize dynamic memory allocation and structs.

Your score will be computed out of a maximum of 100 points based on the following distribution:

- **80** Correctness points
- **10** Effective use of version control points
- **10** Style points

Correctness points: Your code will be evaluated based on a set of given input parameters and should result in correct numbers for:

1. Grid world after each simulation step
2. Infected and survived human numbers with number of dead zombies

You should pay extra attention to get the same output as examples, especially iteration number and number of blank lines. Due to auto-grader you may lose significant points.

Effective use of version control points: You are required to push your changes to the repository frequently. If you only push the final version, even if it is implemented 100% correctly, you will lose a fraction of the grade because you are expected to learn to use Version Control Systems effectively. You do not have to push every small piece of change to GitHub, but every meaningful change should be pushed. For example, each of the functions coded and tested can be one commit. **For each function, there should be at least one commit (with proper commit message) that includes just modifications on that function.**

Style points: Finally, we've reserved 10 points for a subjective evaluation of the style of your solutions and your commenting. Your solutions should be as clean and straightforward as possible. Your comments should be informative, but they need not be extensive.

Handin Intructions

Same as previous assignments, we use GitHub for the submissions as follows. Note that we want you to get used to using a version management system (Git) in terms of writing good commit messages and frequently committing your work so that you can get the most out of Git.

- Commit all the changes you make:

```
$ git commit -a -m "commit message"
(Note: please use meaningful commit messages.)
```

- Push your work to GitHub servers:

```
$ git push origin main
```

Advice

- You will find many hints in the template code. While none of them is required, they are strongly advised since they both will make the assignment easier, and will make easier to get you partial points.
- Keep your GitHub repository by frequently committing the changes.
- Don't forget GDB and Valgrind since they can save a lot of time in debugging.
- Use linuxpool.ku.edu.tr Linux servers to test your code in order to avoid compatibility issues.
- You can write a makefile for your own project that helps you compile and run your code faster.

How to use linuxpool.ku.edu.tr linux servers

- Connect to KU VPN (If you are connected to the KU network, you can skip this step.) See for details:
- Connect to linuxpool.ku.edu.tr server using SSH (Replace USER with your Koc University username):

```
$ ssh USER@linuxpool.ku.edu.tr
(It will ask your password, type your Koc University password.)
```

- When you are finished with your work, you can disconnect by typing:

```
$ exit
```

Your connection to the server may drop sometimes. In that case, you need to reconnect.

We advise you to watch the following video about the usage of SSH, which is used to connect remote servers, and SCP, which is used to transfer files between remote servers and your local machine:

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss the given assignments with your classmates, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the web material as everything on the web has been written by someone else. See Koç University - Student Code of Conduct.

Late Submission Policy

You may use up to 7 grace days (in total) over the course of the semester for the assignments. That is, you can submit your solutions without any penalty if you have free grace days left. Any additional unapproved late submission will be punished (1 day late: 20 % off, 2 days late: 40 % off) and **no submission after 2 days will be accepted.**

Regulations

- **Blackboard:** This is an individual assignment, and all hand-ins are electronic. Any clarification or correction will be announced on the Blackboard.
- **Cheating:** We use automated plagiarism detection to compare your assignment submission with others and also the code repositories on GitHub and similar sites. Moreover, we plan to ask randomly selected 10% of students to explain their code verbally after the assignments are graded. And one may lose full credit if he or she fails from this oral part.