



AMAZE-US: PERFECT MAZE GENERATOR

using Iterative Depth-First
Search Algorithm

TODAY'S AGENDA

Highlights:

- App Updates/
Improvements

1

Project Proponent

2

Project Co-proponent & Project Proponent's
Department

3

Background Significance

4

Objectives

5

Limitations

6

App Improvements/Updates

7

Updated Plan for Dissemination of
Findings

PROJECT PROPONENT



David Estrella

Project Manager
& Developer



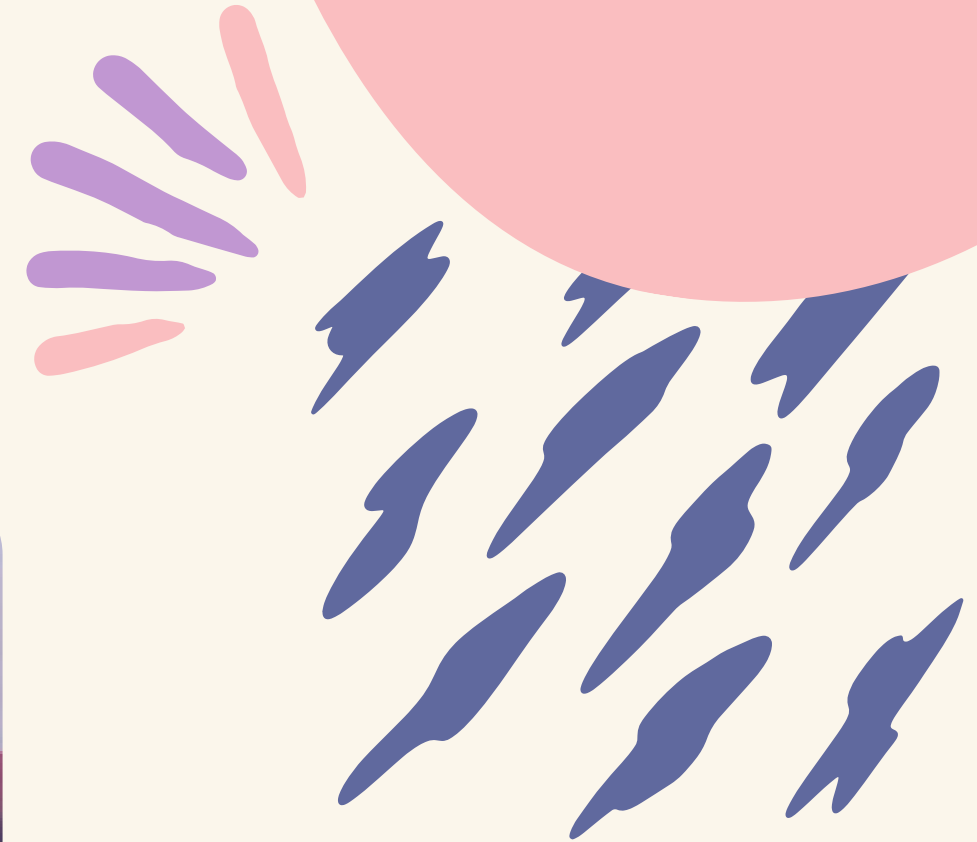
Jovielette Orlanda

Project Designer



Eden Inovejas

Business Analyst



Joshua Mateo

Technical Writer



Alleina Abad



System Analyst



Alijah Andres

UI/UX Designer





PROJECT CO-PROPONENT

Ms. Angie Payne, MSIT, MC

PROJECT PROPONENT'S DEPARTMENT

College of Computer & Information
Sciences (CCIS)



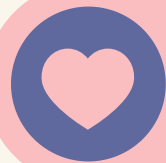


BACKGROUND SIGNIFICANCE

A maze is a tour puzzle consisting of a complex system of paths. Mazes can be found in the video game industry wherein there is a demand for random creation of interior and/or exterior environments. Mazes are also often used in psychological experiments to study spatial navigation and learning on rats or mice. As you can see, mazes have been applied more and more often within real life. Our research aims to use maze generation algorithms to automatically create a perfect maze. A Perfect maze is defined as a maze that satisfies these two conditions: all cells are part of a unique connected space and no cyclic path is allowed in its construction. In order to remove bias, the researchers also set the algorithm to choose the entry and exit points that have the longest paths.

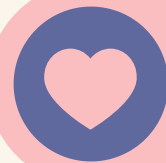
OBJECTIVES

Use the Iterative Depth-first Search (DFS) algorithm to repeatedly generate a perfect maze. A perfect maze satisfies these three conditions:



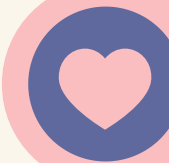
1

All cells are part
of a unique
connected space



2

No cyclic path is
allowed in its
construction



3

The chosen entry and
exit is the longest
path in the maze



SAFETY AND MONITORING PLAN



This section outlines and documents monitoring activities designed to protect the subjects' safety and the data's validity. – Not Applicable

ETHICAL CONSIDERATION

A. Informed Consent (Applies to studies using human subjects)

NOT APPLICABLE 

B. Risks and Side Effects (Applies to studies using human subjects)

NOT APPLICABLE 

C. Benefits to Subjects (Applies to studies using human subjects)

NOT APPLICABLE 

D. Costs to Subject (Applies to studies using human subjects)

NOT APPLICABLE 

E. Compensation to Subject (Applies to studies using human subjects)

NOT APPLICABLE 

F. Provisions for vulnerable subjects (Applies to studies using human subjects)

NOT APPLICABLE 

G. Subject Privacy and Data Confidentiality (Applies to studies using human subjects)

NOT APPLICABLE 

DISSEMINATION OF FINDINGS

In the case of our maze generator, the findings could include the efficiency and effectiveness of the algorithm that we used to generate the mazes, as well as any potential applications or improvements to the algorithm. We can also share our findings with others in the field through online forums or by sharing our code and data on websites such as GitHub. Overall, our aim is to disseminate the findings and share the results with others and contribute to the broader body of knowledge in the field.

PREVIOUS LIMITATIONS

A Maze Us Generator app can only randomly generate mazes but it **cannot solve it**. The app also recommends a **minimum input of 2** rows and columns and a **maximum of 100** rows and 100 columns for **optimal performance**. Larger grid sizes would make the maze cells too miniscule to be visible to the eye. This is because the app screen size is fixed at **700x700 pixels**, it is **not resizable or responsive**. Since the app generates a new randomized maze for every run, it does not store previously generated mazes. The maze also does not have entry & exit points. Also since the program has li, it executes slowly for very large grid sizes.

PREVIOUS VERSION

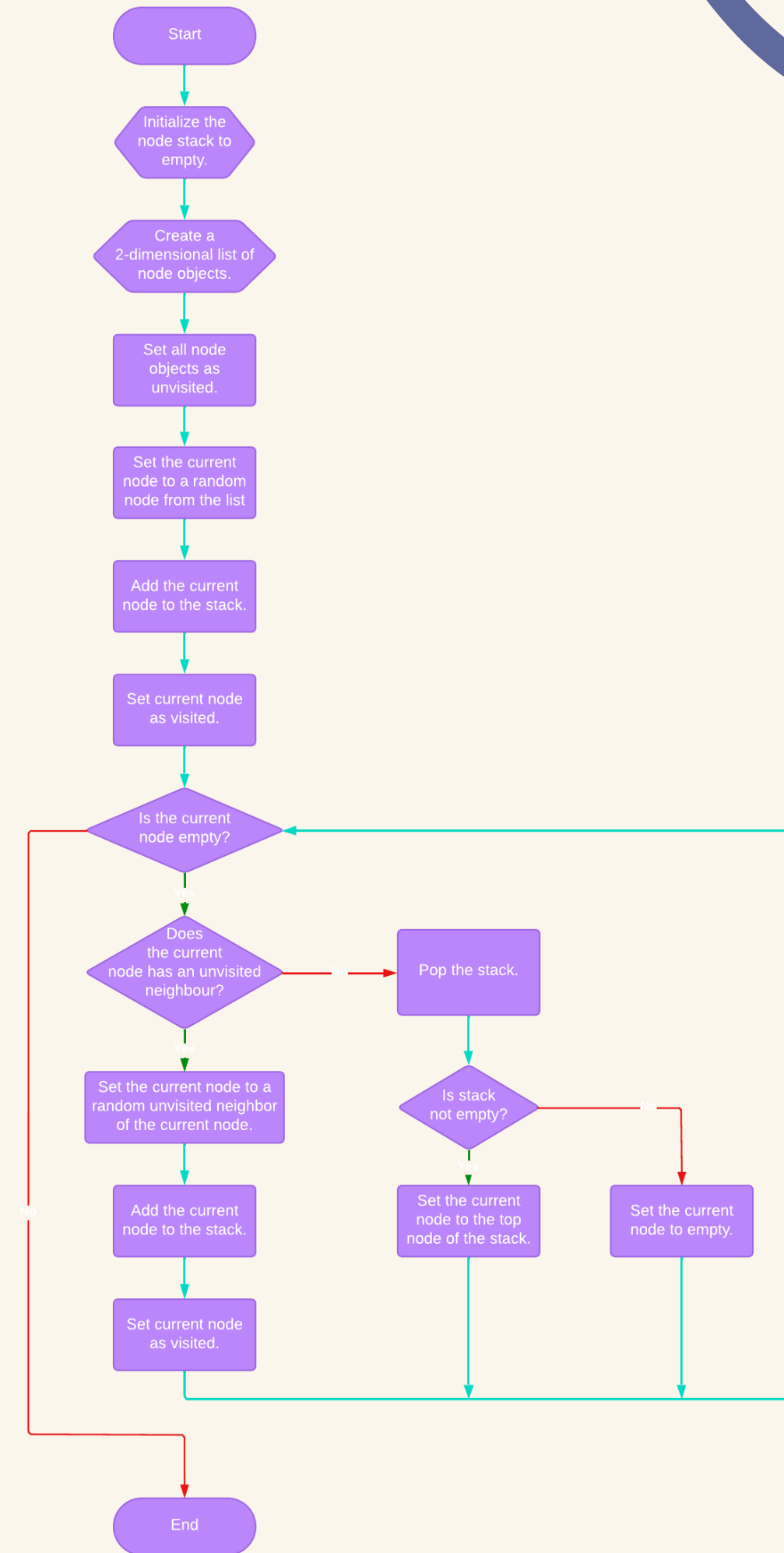
```
ador ~$ ffmpeg -video_size 1366x768 -framerate 25 -f x11grab -i :0.0 ds2.mp4
ersion 4.4.3 Copyright (c) 2000-2022 the FFmpeg developers
with gcc 10.2.1 (GCC) 20201203
ration: --prefix=/usr --disable-debug --enable-gpl --enable-gnutls --disable-stri
nable-libcdio --enable-version3 --enable-runtime-cpudetect --enable-libmp3lame --
ovorbis --enable-libxvid --enable-libx264 --enable-libvpx --enable-libtheora --en
ed --enable-static --enable-libxcb --enable-libpulse --enable-libfreetype --enabl
lug --enable-libspeex --enable-libcelt --enable-libass --enable-libopus --enable-
-enable-libjack --disable-libopencore_amrnb --disable-libopencore_amrwb --disable
peg --enable-postproc --enable-opencl --enable-libx265 --enable-libv4l2 --enable-
enable-vaapi --enable-vdpau --enable-libbs2b --enable-avresample --enable-libvids
ble-libdav1d --disable-libzimg --enable-libwebp --disable-libmysofa --enable-vulk
le-libdrm --enable-libsvtav1 --enable-libsrt --enable-librist --enable-nvenc --en
c
il      56. 70.100 / 56. 70.100
dec     58.134.100 / 58.134.100
rmat    58. 76.100 / 58. 76.100
vice    58. 13.100 / 58. 13.100
lter     7.110.100 /  7.110.100
sample  4.  0.  0 /  4.  0.  0
ale      5.  9.100 /  5.  9.100
sample  3.  9.100 /  3.  9.100
proc    55.  9.100 / 55.  9.100
```

```
nokto@explorador ~$ cd programming/python/mid/
nokto@explorador mid$ python game.py
pygame 2.1.3.dev8 (SDL 2.0.22, Python 3.11.0)
Hello from the pygame community. https://www.pygame.org/contribute.html
nokto@explorador mid$
```

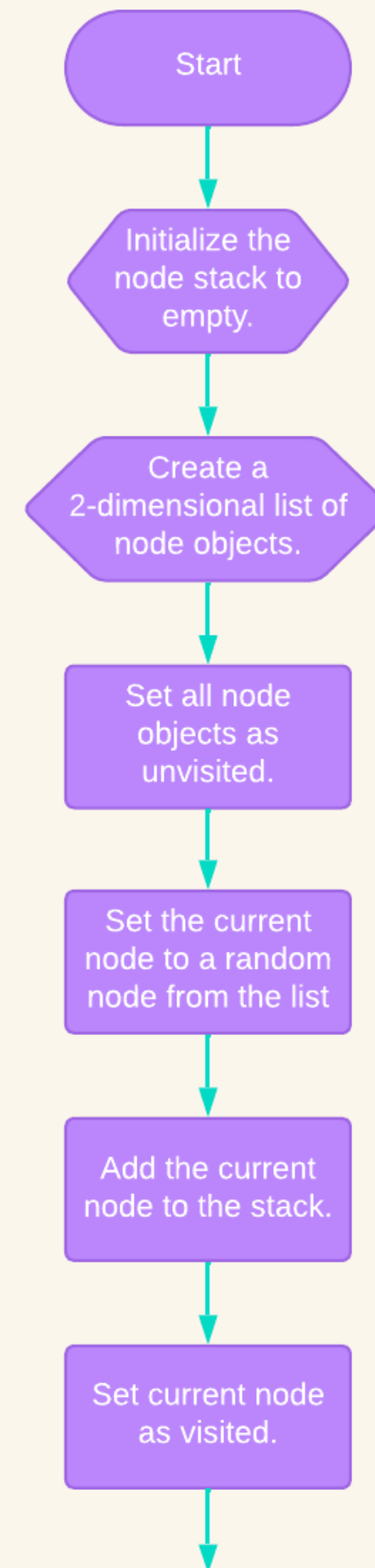
IMPROVEMENTS

- **Fixed minor graphical glitches.**
- **Generated mazes can now be exported as a PNG file.**
- **Faster maze generation**
- **Added start and finish nodes.**

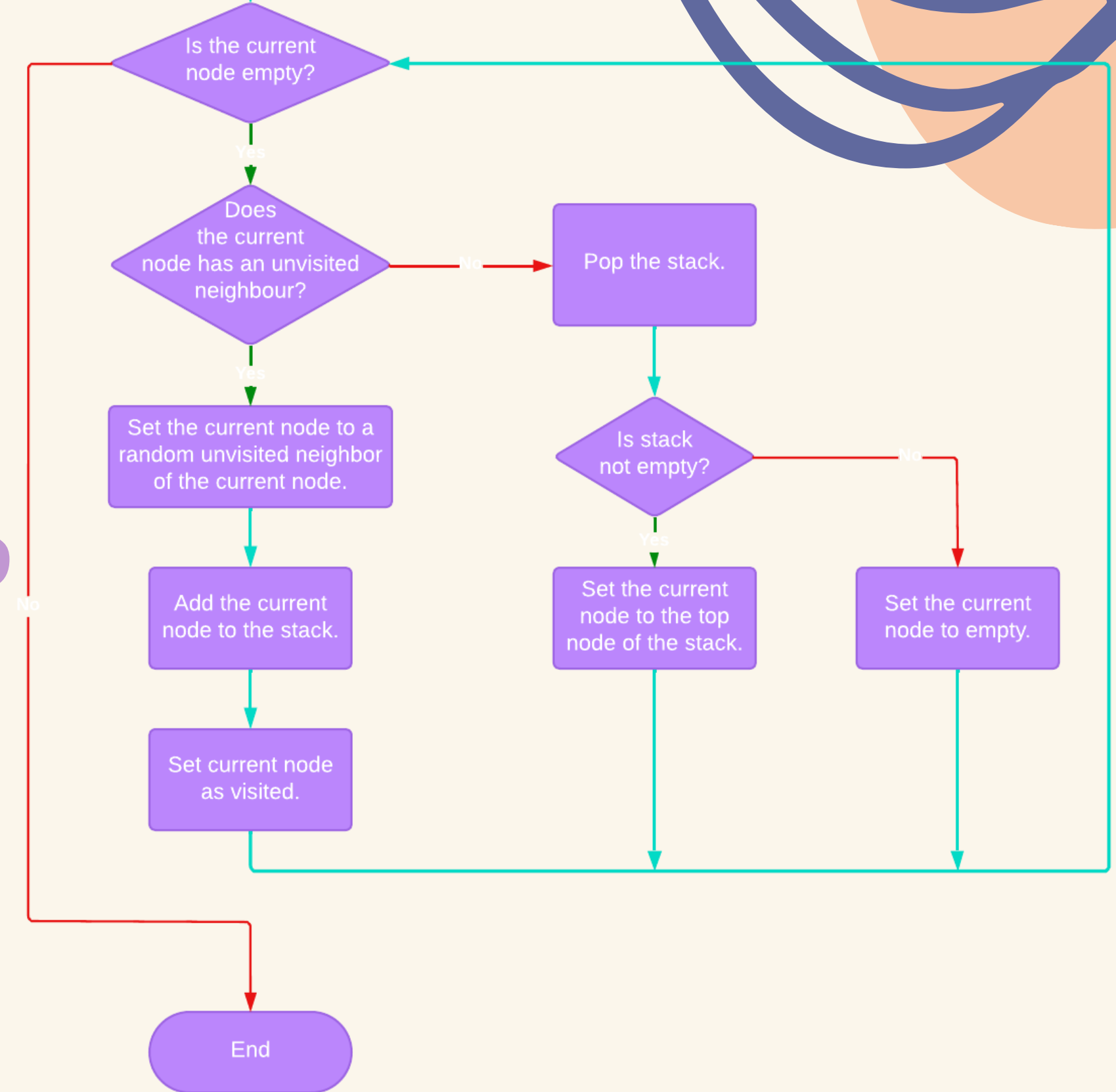
FLOWCHART



FLOWCHART



FLOWCHART



PSEUDOCODE

PSEUDOCODE

START

Initialize the node stack \leftarrow NULL

Create a 2-Dimensional list of node objects.

Set all node objects as unvisited

Set current node to a random node from the list

Add the current node to the stack

Set current node as visited

While !current node == NULL

If current cell has unvisited neighbor

Set current node to a random unvisited neighbor of the current node

Push stack \leftarrow current node

Mark current node as visited

Repeat loop

End if

Pop the stack

If stack \neq NULL

Set the current node to the top node of the stack

Repeat loop

End if

Set current node \leftarrow NULL

End While

END

REFERENCES

- V. Bellot, M. Cautrès, J-M. Favreau, M. Gonzalez-Thauvin, P. Lafourcade, K. Le Cornec, B. Mosnier, S. Rivière-Wekstein, How to generate perfect mazes, Information Sciences, Volume 572, 2021, Pages 444-459, ISSN 0020-0255, <https://doi.org/10.1016/j.ins.2021.03.022>. (<https://www.sciencedirect.com/science/article/pii/S0020025521002656>)
- Hui-Lung Lee, Chia-Feng Lee, Ling-Hwei Chen, A perfect maze based steganographic method, Journal of Systems and Software, Volume 83, Issue 12, 2010, Pages 2528-2535, ISSN 0164-1212, <https://doi.org/10.1016/j.jss.2010.07.054>. (<https://www.sciencedirect.com/science/article/pii/S0164121210002104>)
- Data and Safety Monitoring Plan Writing Guidance. (n.d.). National Institute of Mental Health (NIMH). <https://www.nimh.nih.gov/funding/clinical-research/data-and-safety-monitoring-plan-writing-guidance>
- - H.Urna, H. (2021, December 12). Maze generations: Algorithms and Visualizations. - Analytics Vidhya. Medium. <https://medium.com/analytics-vidhya/maze-generations-algorithms-and-visualizations-9f5e88a3ae37>
- Dissemination of research. (n.d.). Home. <https://www.unisa.edu.au/research/integrity/responsible-research-practice/dissemination-of-research/>



THANK YOU FOR
LISTENING!