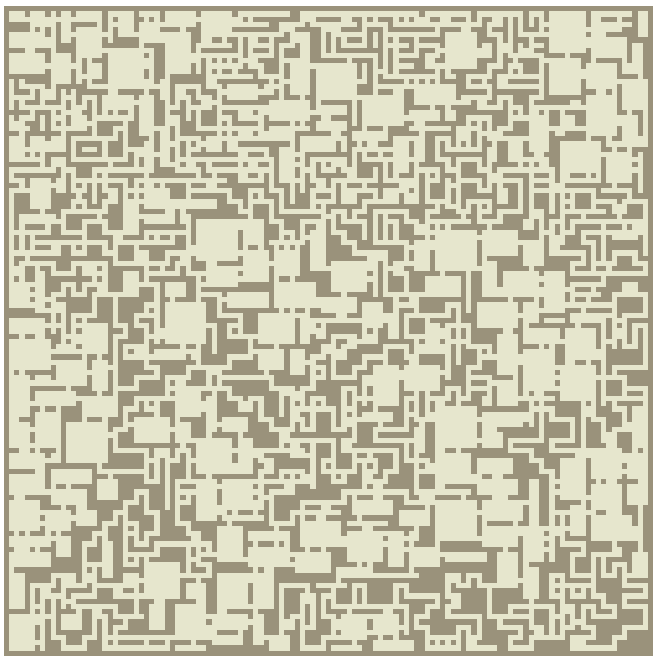
Advanced Higher Computing Coursework

⟨DUNDGEN⟩



Lewis Frampton
SQA Number: 063042323
Forrester High School, Edinburgh

Table Of Contents

1.0 - Project Proposal

1.0 Project Idea

This project originally was created with the intention of developing it into a game where the user would have to navigate a randomised maze. This idea was relatively simple, with the largest hurdle being the random generation of the maze. Once I figured out a way of creating it however, I realised that - apart from that one part of the program - it lacked any depth and was incredibly boring to work on from that point onwards. Not wanting to make the time spent on the maze game useless, I pivoted slightly towards creating a randomised dungeon generator for tabletop games, such as D&D or Pathfinder. This solution would still create a randomised maze but would also populate it with rooms and poke holes in the maze, creating a much more interesting end product.

Before any work had to be done on the new project, I have to distinguish the difference between a maze and a dungeon. A maze will only have long stretching corridors and a single path between any two points whilst a dungeon will have long corridors broken up by rooms and also have passages between separate corridors that allow for players to loop back on themselves. DundGen will allow for a user to customise both the size of the maze and the amount of gaps that are poked into it. It will also allow the user to choose if they want to export the maze as an image, a text file or both. The user will also have the option of quitting the program at any point, except for while the dungeon is being generated as once it starts it cannot be stopped.

1.1 Feasibility Study

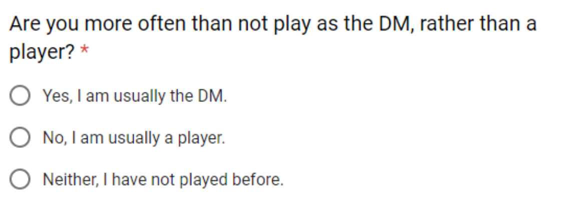
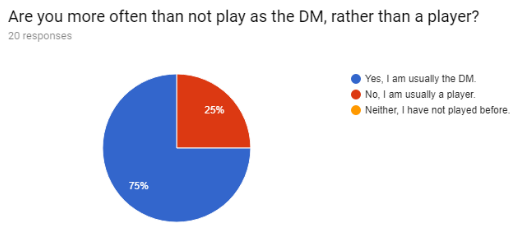
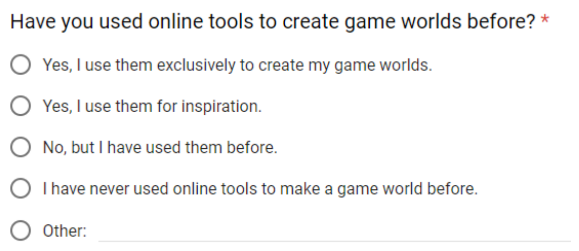
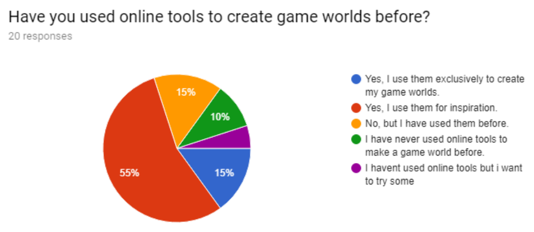
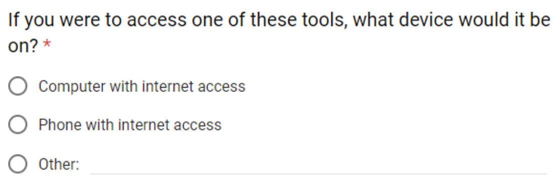
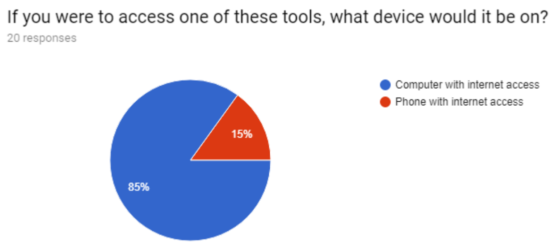
Projects like this have been very successful in the past, with websites such as ‘donjon’ or even the official Wizards of The Coast random dungeon generator.

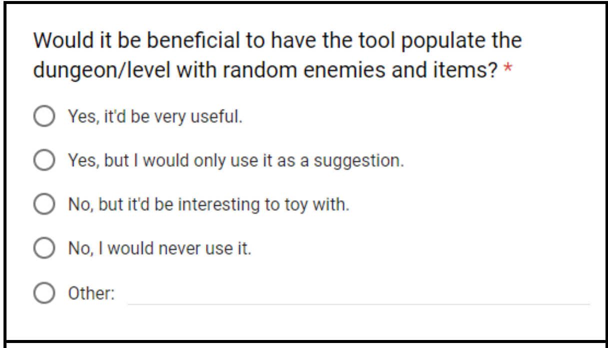
Technologically speaking, this is definitely feasible as it’s still a fairly basic program. There are no costs in creation and since I have no intention of selling the program, there are no legal issues. Any potential legal issues that I may run into by mention the name of the D&D system is cleared up by the openSRD that WoTC published to allow for free fan creation (https://open5e.com/license.html). This program also has plenty time to be created, meaning that it is feasible within the time limit.

1.2 User Survey

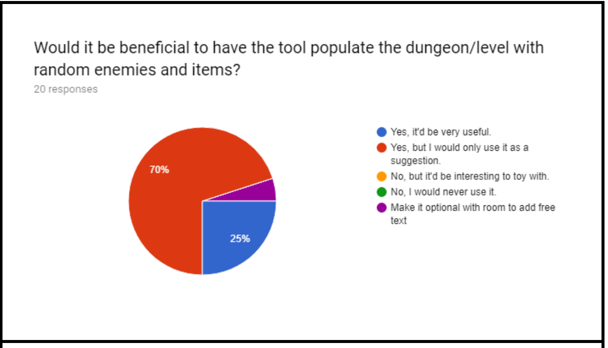
Before design could properly begin, it was important to understand who the target audience would be. As the games are generally rated to be 13+, the complexity of the program will be designed so that at least a teenager could understand how to use the program. Though age was now cleared up, I had to gather details on how individuals would interact with the program and how much experience with computers they had. To do this I created a survey (https://docs.google.com/forms/d/e/1FAIpQLSf6op4NxrYwxsaDtf7HxiD0cfm7blcqryvtl8B3vpjqkmeeYw/viewform) through Google Forms (https://www.google.co.uk/forms/about/) and then published the survey on the official D&D subreddit (https://www.reddit.com/r/DnD/). I chose this source as the subreddit is one of the largest collections of players (480,231 users as of the time this was written) that are readily available, knowledgeable on the game and are friendly to these types of surveys.

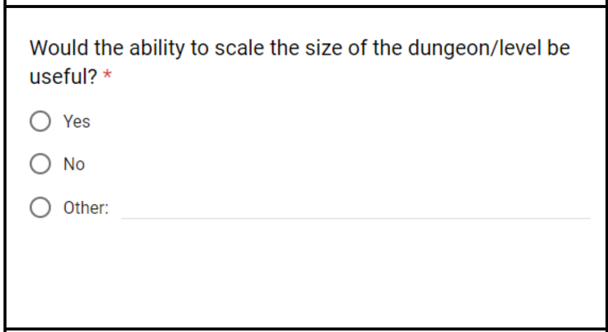
Survey:

Question	Responses
 Are you more often than not play as the DM, rather than a player? * ◯ Yes, I am usually the DM. ◯ No, I am usually a player. ◯ Neither, I have not played before.	 Are you more often than not play as the DM, rather than a player? 20 responses
 Have you used online tools to create game worlds before? * ◯ Yes, I use them exclusively to create my game worlds. ◯ Yes, I use them for inspiration. ◯ No, but I have used them before. ◯ I have never used online tools to make a game world before. ◯ Other:	 Have you used online tools to create game worlds before? 20 responses
 If you were to access one of these tools, what device would it be on? * ◯ Computer with internet access ◯ Phone with internet access ◯ Other:	 If you were to access one of these tools, what device would it be on? 20 responses

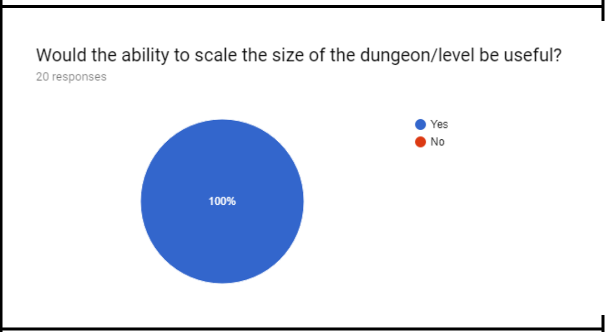

Would it be beneficial to have the tool populate the dungeon/level with random enemies and items? *

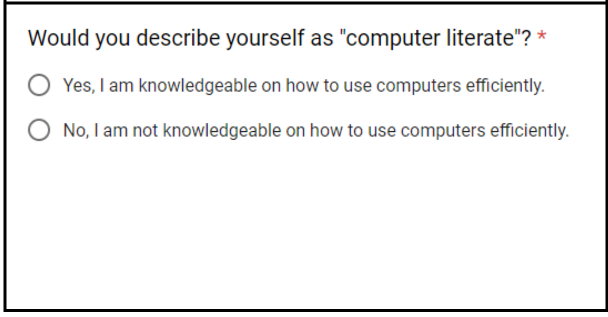
- ○ Yes, it'd be very useful.
- ○ Yes, but I would only use it as a suggestion.
- ○ No, but it'd be interesting to toy with.
- ○ No, I would never use it.
- ○ Other: ____



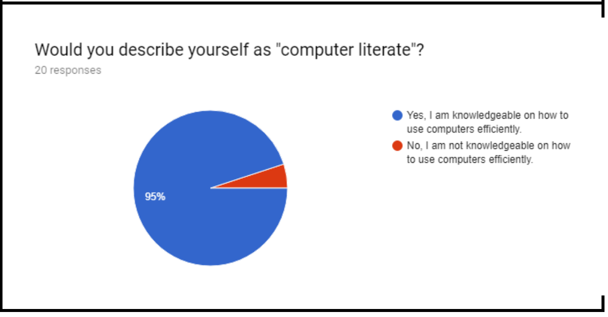

Would the ability to scale the size of the dungeon/level be useful? *

- ○ Yes
- ○ No
- ○ Other: ____




Would you describe yourself as "computer literate"? *

- ○ Yes, I am knowledgeable on how to use computers efficiently.
- ○ No, I am not knowledgeable on how to use computers efficiently.



1.25 Analysis Of Findings

1) This question, though having little impact on the end product, was interesting.It showed that around 75% of end users would be using the program to plan out and design game sessions whilst the other 25% would be using the program as more of a toy to mess around with. This helped me realise that a large majority of end users would probably want to export a version of the dungeon when it was finished being generated.
2) This question showed the most disparity. It showed that majority (55%) of end users don't use these programs and leave the finished products as they are. They commonly alter the map afterwards. 15% of the end users would of taken and used the map as it was without alterfication. 15% of the end users would not of used them at all, but have used similar programs before. 5% of the end users have never used similar programs. This was valuable, as it shows that 95% of my target audience have used similar tools before. This information allows me to make the program more complex and not have to 'lead the user by the hand' as much.
3) This question showed that around 85% of end users would use computers as the device to access these types of programs. This showed me that the program should be developed with computer users in mind, instead of aiming for a mobile market.

4) This question was to gauge if it was important for DunGen to include an option to randomly populate the dungeon with monsters or traps. 100% agreed on the option being useful, though 70% said they would only use it as a suggestion.
5) With a solid 100%, I can assume that all of my target audience would view the ability to alter the size of the dungeon to be vital.
6) From the responses, I can assume that 95% of my target audiance are computer literate. This allows me to use more complex options and lets me presume that a very large majority of users could easily navigate the basic program.

This survey proved to be very useful in the end. With 20 responses, I could get a fairly large sample size of users. In turn, this allowed for me to presume that the individuals that took the survey are a good representation of potential end users.
One factor that could of hindered the surveys validity was the fact that I only published it in one location. The fact that this location was already online means that it is unlikely that any non-computer-literate users would of seen this survey. As a result, responses may of been skewed towards computer literate users.

1.3 Gantt Chart 1.0 - First Draft - 11/09/17

18						March
17						March
16						March
15					February	
14					February	
13					February	
12				February		
11				January		
10			December			
9			December			
8			November			
7			November			
6			November			
5			November			
4		October				
4		October				

3		October				
2	September					
1	September					
Week	Analysis 2 Weeks	Design 3 Weeks	Implementation 5 Weeks	Testing 2 Weeks	Documentation 3 Weeks	Evaluation 3 weeks

1.31 Gantt Chart 1.1 - Second Draft - 12/10/17

Made during the decision to switch from a random maze game to a dungeon generator. More analysis had to be done to accommodate this shift. More development time had to be taken to include the extra features. To account for this time, design and documentation were shortened.

18						March
17						March
16						March
15					February	
14					February	
13				January		
12				January		
11			December			
10			December			
9			December			
8			December			
7			November			
6			November			
5			November			
4			November			
4		October				
3	October					

2	October					
1	October					
Week	Analysis 3 Weeks	Design 1 Weeks	Implementation 8 Weeks	Testing 2 Weeks	Documentation 2 Weeks	Evaluation 3 weeks

From this point onwards, this Gantt Chart proved to be fairly accurate. Of course on some weeks I worked on other parts of the project or sections worked over their slot by a few days. I felt as though these minor delays didn't need to be documented on a revised chart as they were insignificant in the long run. Since I changed what I planned on doing, I had to change the chart to show the extra weeks I'd now be working in october, november and december to compensate for the time I'd already spent.

1.4 Resources

To create this project, I require a computer that is capable of running python 3.0+, has an internet connection and has the python module Pillow (http://pillow.readthedocs.io/en/4.0.x/index.html) installed. This module is used to allow for imaging editing and in DunGen, is used to create and output the image file of a create dungeon. These requirements will prove to be an issue as the school computers which I work on either don't have python installed, have an anti-virus that interferes with python's file handling aspects, often have poor internet connections and often have restrictions on installing new modules. Luckily, I found a work-around for most of these issues. The website Repl.it has proven to be perfect for my needs. The online development environment has allowed me to bypass the anti-virus and has allowed me to program on any computer, even those without python. The cloud based storage system has also allowed me to quickly work on numerous computers without losing any work. More recently, the website has been updated to allow modules to be installed onto it, allowing for Pillow to be installed onto it. This solution luckily solved my issues with resources.

1.7 - Allocated Times

Task	**Time Allocated (Predicted time task will take to complete)**
Analysis	9 hours
Design	3 hours
Implementation	24 hours
Testing	6 hours
Documentation	6 hours
Evaluation	9 hours
Total Time:	54 hours

1.8 Refinement
The biggest change to the project plan was probably the decision to change the direction the project was heading. The idea to switch from a maze game to a random dungeon generator was lingering around for a while. While the maze was interesting and feasible, the dungeon generator was all of that but also fun to work on. It linked a hobby with the project, making it less of a school project and more of a passion project. The choice to switch paths was made on the 12/10/17.

2.0 - Requirements Specification

2.0 Scope
This program will allow a user to play through a randomly generated maze, making their way from one corner to the other. The objective of this is to provide entertainment in the form of a maze game. The program will use a menu system that will allow the user to begin the game or exit the game. At any point during the game the user may exit. At the end of the game, the user will be shown the minimum amount of moves it takes to solve the maze and the amount of moves it took them to solve the maze.

2.01 Revision 1, 12/10/17: - change from maze game to dungeon generator
This program will allow a user to randomly generate a dungeon. The objective of this is to allow for players of popular tabletop games to quickly create levels, allowing for more time spent on other preparation. The program will use a menu system that will allow the user to begin generation, alter generation options or exit the program. In the options the user will be able to alter the amount of gaps that are put into the dungeon, decide if they want to output the dungeon as an image and decide if they want to output the dungeon as a text file.

2.02 Revision 2, 15/12/17: - addition of dungeon size and population options
This program will allow a user to randomly generate a dungeon. The objective of this is to allow for players of popular tabletop games to quickly create levels, allowing for more time spent on other preparation. The program will use a menu system that will allow the user to begin generation, alter generation options or exit the program. In the options the user will be able to alter the size of the maze, alter the amount of monsters that populate the maze, alter the amount of gaps that are put into the dungeon, decide if they want to output the dungeon as an image and decide if they want to output the dungeon as a text file.

2.03 Final Revision, 05/01/17: - removal of population options due to time constraints
This program will allow a user to randomly generate a dungeon. The objective of this is to allow for players of popular tabletop games to quickly create levels, allowing for more time spent on other preparation. The program will use a menu system that will allow the user to begin generation, alter generation options or exit the program. In the options the user will be able to alter the size of the maze, alter the amount of gaps that are put into the dungeon, decide if they want to output the dungeon as an image and decide if they want to output the dungeon as a text file.

2.1 Boundaries

As seen above (**2.0**), I tend to have difficulty limiting my scope. This tends to conflict with limited time windows and with the project deadline always getting closer, I had to know my projects limits. The end-product that I end up developing will be able to do all features described within **2.03** and no more. One feature that could be added in later however, is the option to randomly populate the dungeon with monsters. This would of opened up more customizability with the dungeon, offering users to customise the difficulty of the monsters, types of monsters and frequency of monsters.

2.2 End Users

As discussed in **1.25**, the target audience that I'm aiming for are people interested in games such as 'Dungeons & Dragons' or Pathfinder. As these games are rated 13+, I'm aiming for a target audience of people within their teens or older. Through my survey (**1.2**), I found that a large majority of the end users would be computer literate. I also found that a large amount of them would be using the program to help with designing a game for players to go through.

To collect this information, we have the target audience being someone who is in their teens or later, interested in running a tabletop RPG game and is computer literate.

2.3 User Requirements

To use DundGen, it is strongly recommended to be running a computer with a version of Windows Operating system that is more recent that Windows 7. The reason for this, is the fact that I will only be able to test DundGen on computers that are running Windows. Though program may work on other operating systems, they will not be supported.

Requirements

- A computer that is capable of running with the Windows Operating System installed. Versions of windows that are supported include Windows 7 and any more recent versions of the operating system up until 22/05/18. Windows requirements are seen here: Link_To_Windows_7_Requirements
- The user must have at least a keyboard and some form of display to interact with the program.
- 1.52 MB of free disk space. This file size is calculated from the total file size of the program itself and the largest default image and text outputs. A larger amount of free disk space may be required if the user wishes to create a larger dungeon; Though this option is available, iit will not be officially supported.
- At least 14 MB of memory free. This was taken by running Task Manager - whilst the program was creating the largest supported dungeon - which showed a peak of 13.7 MB of memory being used.
- Python 3.6+ installed. Though other version of python may allow for the program to run, only versions more recent than 3.6.3 are supported.
- The module for Python, Pillow, must be installed on the computer. For support on Pillow, see: https://pillow.readthedocs.io/en/5.1.x/

If the user does not wish to have DundGen run locally, they require:

- A computer that is capable of running with the Windows Operating System installed. Versions of windows that are supported include Windows 7 and any more recent

versions of the operating system up until 22/05/18. Windows requirements are seen here: Link_To_Windows_7_Requirements
- The user must have at least a keyboard and some form of display to interact with the program.
- A computer that has access to the internet and in turn, access to the Website repl.it.
- A computer that is capable of running an internet browser. The only supported browser is google chrome (see www.chrome.com). The official system requirements for chrome are found at: https://support.google.com/chrome/a/answer/7100626?hl=en

The user must have at least a keyboard and some form of display to interact with the program.
For the user to run the program locally, they must have a computer that is capable of running Python 3.0 or a more recent version. Along with this, the user must also have the module Pillow (http://pillow.readthedocs.io/en/4.0.x/index.html) correctly installed. The users computer must also have at l
OR
The user must have a computer that can reliably access the internet which would allow user to access the program through Repl.It at this link: https://repl.it/@LewisFrampton/AH-Project-Dundgen

2.4 Inputs & Outputs
To interact with the program, I've designed a basic menu system. To interact with this, the user must input integers from their keyboard.
The program will always output to the monitor. It will showing menu options, instructions, generation text or the finished dungeon. If the user wants to, they can enable the program to output the finished dungeon as a text file or as an image file (.bmp).

To summarise:
Inputs: - Menu choices (integer - keyboard)

Outputs: - Display Information (strings - display)
- Image File (.bmp - file)
- Text File(.txt - file)

2.5 - Functional Requirements

Requirement	Explanation
Computer	This will be required as without one, I could not create the program. A computer is also vital to allow users to actually interact with DundGen. The computer must also be capable of accepting

	inputs, outputting information and running a version of python 3. This will also be important as I will be using a computer to research.
Keyboard	This will be required to create the program and will also be required to interact with the program.
Mouse	This will be required to help me research
Wi-Fi	This will be used to help research and also used to interact with the online website ‘repl.it’.
Google Drive	This will be used to store my project and allow for easy transfer between computers
Google Survey	This is what I used to create the surveys and get responses.
Repl.it	This website will be vital to my project as it allows both an easy cloud storage system but also allows me to program on computers that don’t have python installed onto them.
Python 3.6	This will be the programming language that I will use to program DundGen with

2.6 - User Requirements

2.7 - Resources

The websites listed below are what I used through my project:

Website name	URL	Purpose
Stack Overflow	https://stackoverflow.com	To help with any problems I had while programming DundGen
Repl	Repl.it	To allow me to program python online. This allowed me to program on any computer with an internet connection
Google Drive	drive.google.com	To allow for me to collect the project together and store it online
Google Survey	survey.google.com	This is what I used to create the surveys and get responses.

3.0 - Test Plan

3.1 Test Plan

What I am testing	What I expect to happen	What actually happened	How I fixed the issue	Evidence
Does the program when ran, produce a randomly generated dungeon?	The program to create a randomly generated dungeon, filled with rooms and gaps in walls.			
Does altering the size of the dungeon fact result in a larger dungeon?	Default size is 55x55, so I'm going to see if creating a 125x125 maze creates a larger dungeon.			
Does altering the amount of gaps in the maze in fact result in a more open dungeon?	I expect to see a maze with many more gaps between corridors in it. The maze itself should look more open and have strangely connecting corridors.			
Does the program output an image file correctly?	For an image file, in the format .bmp and named after the size of the dungeon, to be exported and saved to the same directory the program is being ran from.			
Does the program output a text file correctly?	For an image file, in the format .txt and named 'dungeon', to be exported and saved to the same directory the program is being ran from.			
Does the dungeon have interconnected rooms?	Run a near-default dungeon generation then look at the finished product to see if anything isn't connected. Gaps in Dungeon will be set to 1 to best test			

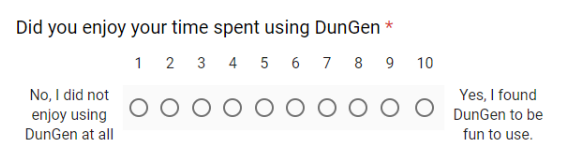
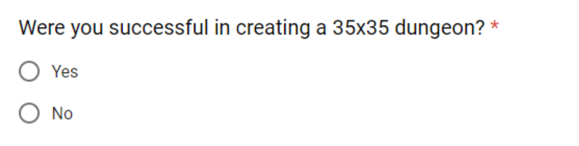
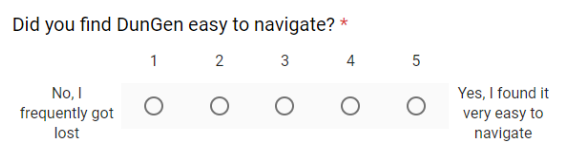
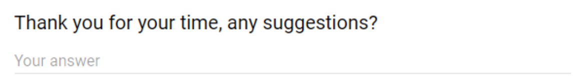
	the rooms own ability to connect to nearby corridors/rooms.			
Does the main menu have working input validation? This will be done by using extreme testing(using ‘0’ and ‘4’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘3’).	In the main menu the program should only accept inputs of the integers 1, 2 or 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the valid input is entered, the program should exit safely.			
Does the option menu have working input validation? This will be done by using extreme testing(using ‘0’ and ‘6’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘5’).	In the option menu the program should only accept inputs of the integers 1, 2 or 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the valid input is entered, the option menu should exit back to the main menu.			
Does the maze size option have input validation? This will be done by using extreme testing(using ‘0’ and ‘7’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘6’).	In the maze size option menu, the program should only accept integers from 1 to 6. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (6), the program will exit back to the option menu			
Does the gaps in maze option have input validation? This will be done by using extreme testing(using ‘0’ and ‘4’ as inputs), exceptional	In the gaps in maze option menu, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid			

testing(using 'abc' as an input) and normal('3').	input. When the correct value is entered (3), the program will exit back to the option menu			
Does the Output Image File option have input validation? This will be done by using extreme testing(using '0' and '3' as inputs), exceptional testing(using 'abc' as an input) and normal('3').	In the Output Image File option, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (3), the program will exit back to the option menu			
Does the Output Text File option have input validation? This will be done by using extreme testing(using '0' and '3' as inputs), exceptional testing(using 'abc' as an input) and normal('3').	In the Output Text File option, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (3), the program will exit back to the option menu			

3.1 End User Survey Plan

For this, I will hopefully contact a few of those that had completed my original survey and ask them to use the program for 5 minutes with the objective of creating a 35x35 dungeon. This choice was chosen as if successful, it shows that the users find the program intuitive. Once they have used the program for this long, I will ask them to complete another user survey:

Question	**Response**

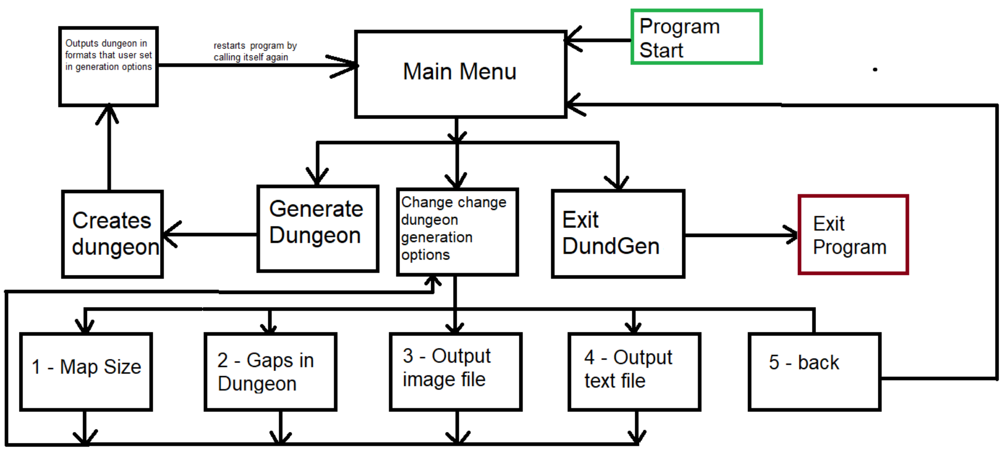
 Did you enjoy your time spent using DunGen * 1 2 3 4 5 6 7 8 9 10 No, I did not enjoy using DunGen at all ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Yes, I found DunGen to be fun to use.	
 Were you successful in creating a 35x35 dungeon? * ○ Yes ○ No	
 Did you find DunGen easy to navigate? * 1 2 3 4 5 No, I frequently got lost ○ ○ ○ ○ ○ Yes, I found it very easy to navigate	
 Thank you for your time, any suggestions? Your answer	

Note: No refinement had to be done in this section as they largest change(from maze to dungeon) was made during the analysis phase.

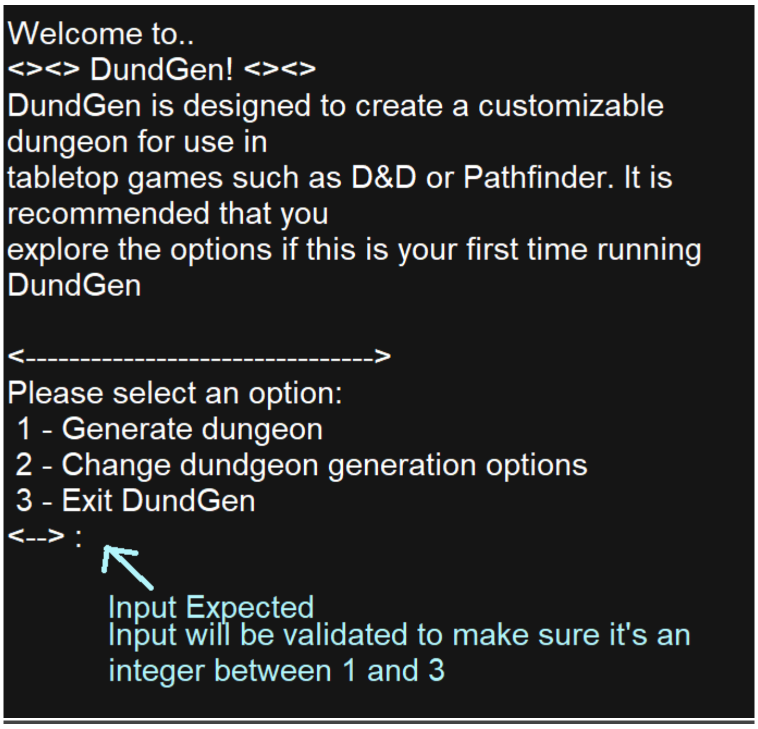
4.0 - Interface Design

4.0 Flowchart

The program will be a basic number-based menu system due to time constraints. Due to my findings in the survey, I could make the programs layout more complex due to around 95% of my target audience being computer literate. To help with my menu design, I created a flowchart that shows how a user would navigate the interface:



4.1 Main Menu



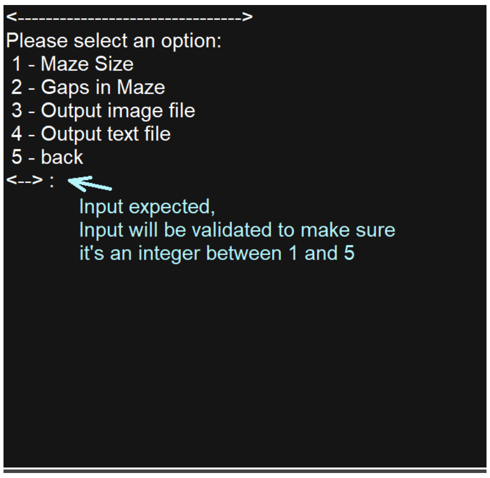
```
Welcome to..
<><> DundGen! <><>
DundGen is designed to create a customizable dungeon for use in
tabletop games such as D&D or Pathfinder. It is recommended that you
explore the options if this is your first time running DundGen

<-------------------------------->
Please select an option:
 1 - Generate dungeon
 2 - Change dundgeon generation options
 3 - Exit DundGen
<--> :
```

Input Expected
Input will be validated to make sure it's an integer between 1 and 3

Brief introduction will only be shown when the program initially starts. The choice of a number based system was made as my target audience are computer literate and chances are, can navigate this fairly simple menu.

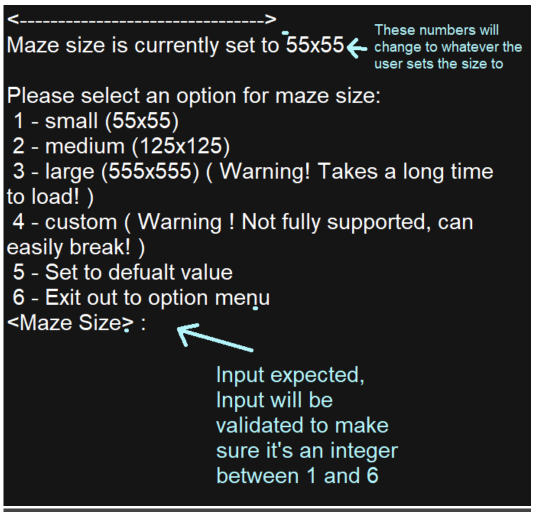
4.2 Option Menu (Change dungeon generation options)



```
<-------------------------------->
Please select an option:
 1 - Maze Size
 2 - Gaps in Maze
 3 - Output image file
 4 - Output text file
 5 - back
<--> :
```

Input expected,
Input will be validated to make sure it's an integer between 1 and 5

4.3 Maze Size Option



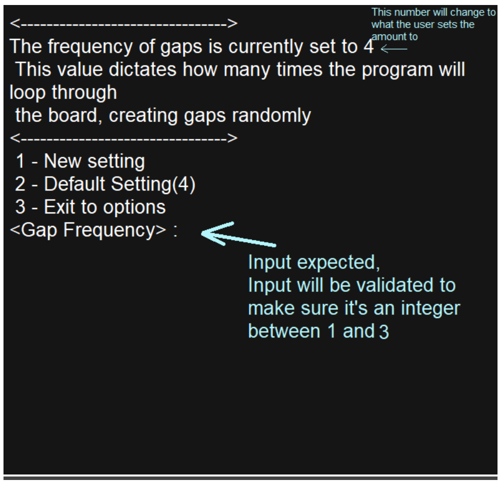
```
<---------------------------------->
Maze size is currently set to 55x55

Please select an option for maze size:
 1 - small (55x55)
 2 - medium (125x125)
 3 - large (555x555) ( Warning! Takes a long time
to load! )
 4 - custom ( Warning ! Not fully supported, can
easily break! )
 5 - Set to defualt value
 6 - Exit out to option menu
<Maze Size> :
```

These numbers will change to whatever the user sets the size to

Input expected, Input will be validated to make sure it's an integer between 1 and 6

4.4 Gaps in Maze Option

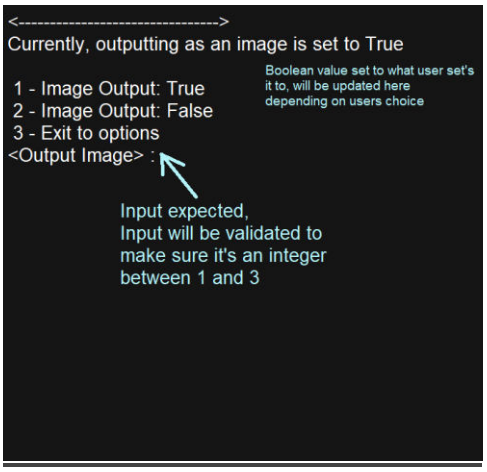


```
<-------------------------------->
The frequency of gaps is currently set to 4
 This value dictates how many times the program will
loop through
 the board, creating gaps randomly
<-------------------------------->
 1 - New setting
 2 - Default Setting(4)
 3 - Exit to options
<Gap Frequency> :
```

This number will change to what the user sets the amount to

Input expected, Input will be validated to make sure it's an integer between 1 and 3

4.5 Output Image File Option



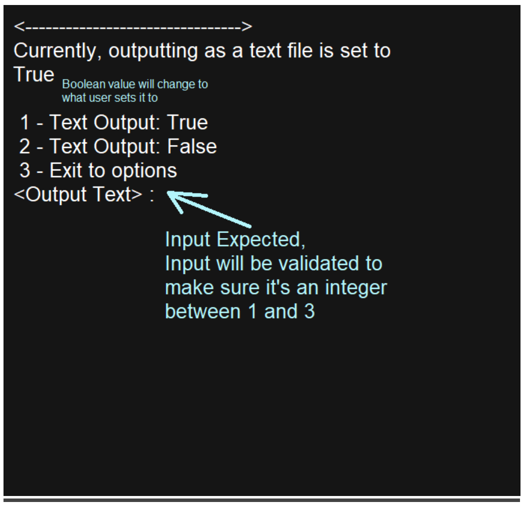
```
<-------------------------------->
Currently, outputting as an image is set to True

 1 - Image Output: True
 2 - Image Output: False
 3 - Exit to options
<Output Image> :
```

Boolean value set to what user set's it to, will be updated here depending on users choice

Input expected, Input will be validated to make sure it's an integer between 1 and 3

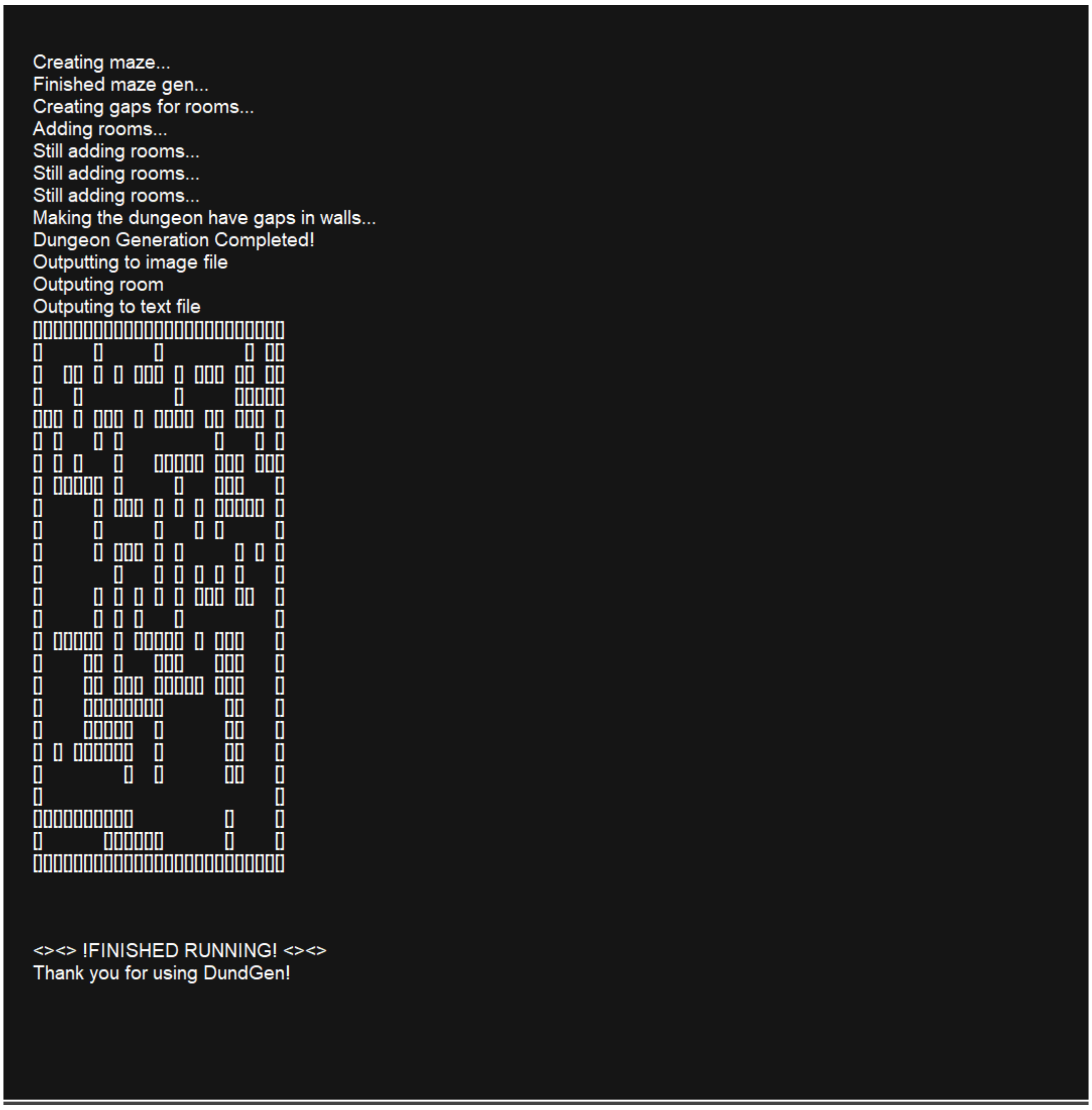
4.6 Output Text File Option



```
<-------------------------------->
Currently, outputting as a text file is set to
True
 1 - Text Output: True
 2 - Text Output: False
 3 - Exit to options
<Output Text> :
```

Boolean value will change to what user sets it to
Input Expected, Input will be validated to make sure it's an integer between 1 and 3

4.7 Generate Dungeon



```
Creating maze...
Finished maze gen...
Creating gaps for rooms...
Adding rooms...
Still adding rooms...
Still adding rooms...
Still adding rooms...
Making the dungeon have gaps in walls...
Dungeon Generation Completed!
Outputting to image file
Outputing room
Outputing to text file
▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
▯     ▯     ▯        ▯ ▯▯
▯  ▯▯ ▯ ▯ ▯▯▯ ▯ ▯▯▯ ▯▯ ▯▯
▯   ▯         ▯     ▯▯▯▯▯
▯▯▯ ▯ ▯▯▯ ▯ ▯▯▯▯ ▯▯ ▯▯▯ ▯
▯ ▯   ▯ ▯         ▯   ▯ ▯
▯ ▯ ▯   ▯   ▯▯▯▯▯ ▯▯▯ ▯▯▯
▯ ▯▯▯▯▯ ▯     ▯   ▯▯▯   ▯
▯     ▯ ▯▯▯ ▯ ▯ ▯ ▯▯▯▯▯ ▯
▯     ▯     ▯   ▯ ▯     ▯
▯     ▯ ▯▯▯ ▯ ▯     ▯ ▯ ▯
▯       ▯   ▯ ▯ ▯ ▯ ▯   ▯
▯     ▯ ▯ ▯ ▯ ▯ ▯▯▯ ▯▯  ▯
▯     ▯ ▯ ▯   ▯         ▯
▯ ▯▯▯▯▯ ▯ ▯▯▯▯▯ ▯ ▯▯▯   ▯
▯    ▯▯ ▯   ▯▯▯   ▯▯▯   ▯
▯    ▯▯ ▯▯▯ ▯▯▯▯▯ ▯▯▯   ▯
▯    ▯▯▯▯▯▯▯▯      ▯▯   ▯
▯    ▯▯▯▯▯  ▯      ▯▯   ▯
▯ ▯ ▯▯▯▯▯▯  ▯      ▯▯   ▯
▯        ▯  ▯      ▯▯   ▯
▯                       ▯
▯▯▯▯▯▯▯▯▯▯         ▯    ▯
▯      ▯▯▯▯▯▯      ▯    ▯
▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯



<><> !FINISHED RUNNING! <><>
Thank you for using DundGen!
```

The first line shows that the program is beginning the generation by starting to create the maze. When this section is done, the second line will be displayed. The third signifies that the maze will begin to be changed into a dungeon by filling in dead ends. The 4th line shows that the program is now beginning to fill the dungeon with rooms. For every 50 attempts, the line 'Still adding rooms…' is printed. This is to show that the program is still running as with larger mazes, this section can take a while. The 8th line shows that the rooms have all been placed and now walls in the dungeon are randomly being removed to give it more of a randomised look. The 9th line shows that the main chunk of the program is finished and that the dungeon has been successfully created! The 10th - 13th lines are only printed if the user wants to output the dungeon to a file. The dungeon is then printed in its final form and the user is shown a small thank you message. At this point the program will restart by taking the user back to the main menu.

4.8 - Reflection

This section has been changed significantly. Originally I planned for the program to ask the user to select a size for the maze out of a predefined list though as the project grew I decided that I wanted to include more options and interactivity. The best choice that would fit this criteria would be the addition of a menu system and the inclusion of an option menu.

This choice was made on the 8/01/18. Though it was out of the time period I had allocated for implementation and long after the design, it didn't take too long to implement.

5.0 - Program Design

5.0 - Initial Plan

The main chunk of the program is to create the maze. At this stage I figure it won't be too hard to port the maze into a game by using PyGame. Though I haven't ever worked with PyGame, I figure it can't hard to learn the basics of movement. The largest hurdle I presume will be to create the randomly generated maze. After researching various methods, I settled on a method called Depth-First Search[1]. This method was chosen due to the fact it is fairly simple to understand and fills the requirement of having a stack.
I decided to split the program into 3 seperate files at this as this will both clean up the design and make the implementation easier. One file would be the main one, which will connect the other two and allow for user interaction. One of the other ones would be responsible for the maze generation while the last one will be responsible for the game aspect.

5.1 - Initial Maze Generation Design

This code was the initial design for the maze creation:

```
INTEGER x = INTEGER INPUT
INTEGER y = INTEGER INPUT
ARRAY board = [[create array with values [create array with values “1” with length y] with length x]
ARRAY cursor = []
Cursor.append(STRING(random(0,x-1))+”,”+STRING(random(0,y-1)))

FUNCTION begin(){
  STRING t = ""
  ARRAY i = t.split(",")
  SET board[INTEGER(i[0]][INTEGER(i[0]]) TO 0
}

FUNCTION validateMovement(Xc, Yc){
  SET Xc TO INTEGER
  SET Yc TO INTEGER
  ARRAY possibleMoves = [FALSE, FALSE, FALSE, FALSE]
  IF Yc <= y - 3 AND board[Xc][Yc+1] == 1 AND board[Xc][Yc+2] == 1{
      SET possibleMoves[0] TO True
  }
  IF Xc <= x-3 AND board[Xc+1][Yc] == 1 AND board[Xc+2][Yc] == 1{
      SET possibleMoves[1] TO True
  }
  IF Yc >= 2 AND board[Xc][Yc-1] == 1 AND board[Xc][Yc-2] == 1{
      SET possibleMoves[2] TO True
  }
  IF Xc >= 2 AND board[Xc-1][Yc] == 1 AND board[Xc-2][Yc] == 1{
      SET possibleMoves[3] TO True
  }
  RETURN possibleMoves
}

FUNCTION move(n){
  IF n == 0{
    t = STRING()
```

[1] http://www.migapro.com/depth-first-search/

```
  }ELSE{
    t = cursor.pop(-1)
  }
  SET n TO 0
  ARRAY i = t.strip().split(",")
  possibleMoves = validateMovement(i[0], i[[1])
  IF STRING(possibleMoves) IS NOT '[False, False, False, False]'{
    INTEGER d = random(3)
    SET i[0] TO INTEGER
    SET i[1] TO INTEGER
    IF d == 0 AND possibleMoves[0] == True{
      SET board[i[0]][i[1]+1], board[i[0]][i[1]+2] TO 0, 0
      cursor.append(str(i[0]) + "," + str(i[1]+2))
    }ELSEIF d == 1 AND possibleMoves[1] == True{
      SET board[i[0]+1][i[1]], board[i[0]+2][i[1]] TO 0, 0
      cursor.append(str(i[0]+2) + "," + str(i[1]))
    }ELSEIF d == 2 AND possibleMoves[2] == True{
      SET board[i[0]][i[1]-1], board[i[0]][i[1]-2] TO 0, 0
      cursor.append(str(i[0]) + "," + str(i[1]-2))
    }ELSEIF d == 3 AND possibleMoves[3] == True{
      SET board[i[0]-1][i[1]], board[i[0]-2][i[1]] TO 0, 0
      cursor.append(str(i[0]-2) + "," + str(i[1]))
    move(0)
  }ELSEIF LENGTH(cursor) IS NOT == 0{
    move(1)
  }
  }
}

FUNCTION main(){
  begin()
  move(0)
}
```

Revision
Obviously this design has changed significantly as the project has shifted focus. The change was decided soon after this design was created, so there is no design of the original game. Fortunately, my new project also needs a maze to be randomly generated which meant this code wasnt for waste. However, in the final product, the maze generation is significantly different as this version proved to have some glaring issues.

One of those issues was the fact that ‘FUNCTION move(n)’, was recursive and called itself until the maze was completed. This resulted the program crashing when the size of the maze grew too large due to the fact that through each loop of move(n), no information was being deleted. Each time it looped through it would create a new set of local variables, eventually resulting in a memory leak.

5.2 - DundGen structure and design
As I did in the original plan, I decided to split DundGen up into multiple python files to allow for an easier implementation and simpler design:

File Name	Function
main.py	This will be the centre of the program.This file will be the location of all global variables, the main menu, input validation and where the other files will be called from.
mazeGen.py	This will take a 2d array (in which there are x number of arrays which have the length x,

	with x being a user chosen value) and turn it into a maze using a randomised depth-first search algorithm.
DungeonGen.py	This will take the finished maze (once mazeGen.py has finished) and turn it into a dungeon by filling in dead ends, adding random holes between walls and adding rooms.
output.py	This is where the code for the outputs are kept. If the user wishes they can output a text file of the finished dungeon, a .bmp file of the finished dungeon or both.

5.21 - main.py Design

```
IMPORT mazeGen.py, dungeonGen.py, output.py

ARRAY size = [55, 125, 555]
INTEGER sizeChoice = 0
INTEGER amountOfGaps = 4
BOOLEAN imageOutput = TRUE
BOOLEAN textOutput = TRUE

SEND 'placeholder welcome message' TO DISPLAY

FUNCTION validateInput(userInput, maxChoice, text){
  IF userInput IS AN INTEGER{
    IF 0 < userInput <= maxChoice{
      RETURN TRUE
    }
  SEND text TO DISPLAY
  RETURN FALSE
  }
}

FUNCTION beginGen(){
  USE GLOBAL size, sizeChoice, amountOfGaps, imageOutput, textOutput
  INTEGER frequencyOfRooms = size[sizeChoice] * 5
  SEND 'Creating Maze...' TO DISPLAY
  ARRAY board = [create array with values [CREATE ARRAY WITH VALUES “1” WITH LENGTH size[sizeChoice]] WITH
LENGTH size[sizeChoice]]
  SET board TO mazeGen.createMaze(board, size[sizeChoice], size[sizeChoice])
  SEND 'Creating gaps for rooms...' TO DISPLAY
  LOOP WITH i IN RANGE size[sizeChoice]{
    SET board TO dungeonGen.createGaps(board, size[sizeChoice])
  }
  SEND 'Adding rooms...' TO DISPLAY
  LOOP WITH i IN RANGE(frequencyOfRooms){
    OBJECT newRoom = dungeonGen.room()
    ARRAY pos = newRoom.findValidPlacement(board, size[sizeChoice])
    IF pos IS NOT FALSE{
      SET board TO newRoom.placeRoom(pos, board)
      SET board TO newRoom.connectRoom(pos, board)
    }
  }
  SEND 'Making the dungeon have gaps in walls...' TO DISPLAY
  LOOP WITH i IN RANGE amountOfGaps{
    SET board TO dungeonGen.imperfectify(board, size[sizeChoice])
  }
  SEND 'Dungeon Generation Completed!' TO DISPLAY
  IF imageOutput IS TRUE{
    SEND 'Outputting to image file' TO DISPLAY
    output.createImage(board, size[sizeChoice])
    }
  if textOutput IS TRUE{
    SEND 'Outputting to text file' TO DISPLAY
    output.createText(board, size[sizeChoice])
    }
  mazeGen.printBoard(board)
  mainMenu(TRUE)
}

FUNCTION optionMenu(){
  STRING choice = ''
  WHILE validateInput(choice, 5, "\n<-------------------------------->\nPlease select an option:\n 1 - Maze
Size \n 2 - Gaps in Dungeon \n 3 - Output image file \n 4 - Output text file \n 5 - back") IS NOT TRUE{
    SET choice TO USER_INPUT
  }
  IF choice == "1"{
    USE GLOBAL sizeChoice
    SEND '"\n<-------------------------------->\nMaze size is currently set to " +
STRING(size[sizeChoice])+"x"+STRING(size[sizeChoice])' TO DISPLAY
    STRING optionChoice = ''
    WHILE validateInput(optionChoice, 6, "\nPlease select an option for maze size:\n 1 - small (55x55)\n 2 -
medium (125x125)\n 3 - large (555x555) ( Warning! Takes a long time to load! ) \n 4 - custom ( Warning ! Not
fully supported, can easily break! ) \n 5 - Set to defualt value \n 6 - Exit out to option menu") IS NOT TRUE{
      SET optionChoice TO USER_INPUT
    }
    IF optionChoice == '1'{
      SEND 'Setting Maze Size to Small' TO DISPLAY
      sizeChoice = 0
    }ELSEIF optionChoice == '2'{
      SEND 'SEtting Maze Size to medium' TO DISPLAY
      sizeChoice = 1
    }ELSEIF optionChoice == '3'{
      SEND 'Setting Maze Size to Large' TO DISPLAY
      sizeChoice = 2
    }ELSEIF optionChoice == '4'{
      SEND 'Setting Maze Size to custom value\n!Warning! Not supported fully! Can easily break! Please enter
an odd integer for the program to work, otherwise this program will crash!' TO DISPLAY
      size.append(int(input()))
      sizeChoice = LENGTH(size) - 1
    }ELSEIF optionChoice == '5'{
      SEND 'Setting Maze Size to default' TO DISPLAY
      sizeChoice = 0
    }ELSEIF optionChoice == '6'{
      optionMenu()
    }


  }
  ELSEIF choice == "2"{
    USE GLOBAL amountOfGaps
    SEND '\n<-------------------------------->\nThe frequency of gaps is currently set to " +
STRING(amountOfGaps) + "\n This value dictates how many times the program will loop through\n the board,
creating gaps randomly' TO DISPLAY
    STRING optionChoice = ''
    WHILE validateInput(optionChoice, 3, "<-------------------------------->\n 1 - New setting \n 2 - Default
Setting(4) \n 3 - Exit to options") IS NOT TRUE{
      SET optionChoice TO USER_INPUT
    }
    IF optionChoice == '1'{
      INTEGER newValue = USER_INPUT
      WHILE validateInput(newValue, 9999999999, "Please enter an integer value") IS NOT TRUE{
        newValue = USER_INPUT
      }
      SET amountOfGaps TO INTEGER newValue
    }
    ELSEIF choice == '2'{
      SET amountOfGaps TO 4
    }
    ELSE{
      optionMenu()
    }

  }
  ELSEIF choice == "3"{
    USE GLOBAL imageOutput
    SEND ‘\n<-------------------------------->\nCurrently, outputting as an image is set to " + 
STRING(imageOutput)’ TO DISPLAY
    STRING optionChoice = ''
    WHILE validateInput(optionChoice, 3, "\n 1 - Image Output: True\n 2 - Image Output: False\n 3 - Exit to 
options") IS NOT TRUE{
      SET optionChoice TO USER_INPUT
    }
    IF optionChoice == '1'{
      SEND 'Output as image set to True!' TO DISPLAY
      SET imageOutput TO TRUE
    }ELSEIF optionChoice == '2'{
      SEND 'Output as image set to False!' TO DISPLAY
      SET imageOutput TO FALSE
    }ELSE{
      optionMenu()
    }
  }
  ELSEIF choice == "4"{
    USE GLOBAL imageOutput
    SEND ‘\n<-------------------------------->\nCurrently, outputting as a text file is set to " + 
STRING(textOutput)’ TO DISPLAY
    STRING optionChoice = ''
    WHILE validateInput(optionChoice, 3, "\n 1 - Text Output: True\n 2 - Text Output: False\n 3 - Exit to 
options") IS NOT TRUE{
      SET optionChoice TO USER_INPUT
    }
    IF optionChoice == '1'{
      SEND 'Output as text set to True!' TO DISPLAY
      SET textOutput TO TRUE
    }ELSEIF optionChoice == '2'{
      SEND 'Output as text set to False!' TO DISPLAY
      SET textOutput TO FALSE
    }ELSE{
      optionMenu()
    }
  }
  ELSEIF choice == "5"{
    mainMenu(False)
  }

}

FUNCTION mainMenu(completed){
  IF completed == TRUE{
    SEND '\n\n<><> !FINISHED RUNNING! <><>\nThank you for using DundGen!\n\n' TO DISPLAY
  }
  STRING choice = ''
  WHILE validateInput(choice, 3, "\n<-------------------------------->\nPlease select an option:\n 1 - 
Generate dungeon \n 2 - Change dungeon generation options \n 3 - Exit DundGen"): IS NOT TRUE{
    SET choice TO USER_INPUT
  }
  IF choice == '1'{
    beginGen()
  }ELSEIF choice == '2'{
    optionMenu()
  }ELSEIF choice == '3'{
    SEND 'Thanks for using DundGen!' TO DISPLAY
    EXIT()
  }
}

mainMenu(False)
```

5.22 - mazeGen.py Design

```
IMPORT MODULE random

ARRAY yMovementAmount = [2, 0, -2, 0]
ARRAY xMovementAmount = [0, 2, 0, -2]

FUNCTION printBoard(board){
  STRING text = ''
  LOOP THROUGH board WITH line{
    LOOP WITH i IN RANGE LENGTH(line){
      IF line[i] == "-"{
        SET text TO text + "[]"
      }ELSE{
        SET text TO text + "  "
    }
    ADD NEW LINE TO text
  }
}
}

FUNCTION createMaze(board, x, y){
  ARRAY start = [-1,-1]
  SET start[0] to random_integer(1, y-1, 2)
  SET start[1] to random_integer(1, x-1, 2)
  SET board[start[0][start[1]] TO ' '
  STACK pathOfCursor = []
  PUSH start ONTO pathOfCursor
  WHILE TRUE == TRUE{
    IF LENGTH(pathOfCursor) == 0{
      SEND 'Finished Maze Gen' TO DISPLAY
      RETURN board
    }
    ARRAY validMovements = [False, False, False, False]
    SET validMovements TO validateDirections(pathOfCursor[LENGTH(pathOfCursor)-1], board)
    IF NOT ANY validMovements{
      POP pathOfCursor
    }
    ELSE{
      INTEGER chosenDir = random_integer(4)
      WHILE validMovements[chosenDir] = FALSE{
        chosenDir = random_integer(4)
      }
      SET board[pathOfCursor[len(pathOfCursor)-1][0] +
yMovementAmount[chosenDir]][pathOfCursor[len(pathOfCursor)-1][1] + xMovementAmount[chosenDir]] TO " "
      SET board[pathOfCursor[len(pathOfCursor)-1][0] + yMovementAmount[chosenDir] // 2
][pathOfCursor[len(pathOfCursor)-1][1] + xMovementAmount[chosenDir] // 2 ] TO " "
      PUSH ([pathOfCursor[len(pathOfCursor)-1][0] + yMovementAmount[chosenDir],
pathOfCursor[len(pathOfCursor)-1][1] + xMovementAmount[chosenDir]]) ONTO pathOfCursor
    }
  }
}

FUNCTION validMovements(curPos, board){
  validMovements = [False, False, False, False]
  LOOP WITH i IN RANGE(4){
    TRY{
      IF board[curPos[0] + yMovementAmount[i]][curPos[1] + xMovementAmount[i]] == "-" AND curPos[0] +
yMovementAmount[i] > 0 AND curPos[1] + xMovementAmount[i] > 0{
        SET validMovements[i] TO TRUE
      }
    }
    EXCEPT{
      pass
    }
  }
  RETURN validMovements
}
```

5.23 - dungeonGen.py

```
IMPORT MODULE random

FUNCTION createGaps(board, size){
  ARRAY deltaX = [0, 1, 0, -1]
  ARRAY deltaY = [1, 0, -1, 0]
  LOOP WITH i IN RANGE(size-1){
    LOOP WITH n IN RANGE(size-1){
      INTEGER amountOfPaths = 0
      IF board[i][n] == " "{
        LOOP WITH k IN RANGE(4){
          TRY{
            IF board[i+deltaY[k]][n+deltaX[k]] == " "{
              amountOfPaths++
            }
          }
          EXCEPT INDEX_ERROR{
            PASS
          }
        }}
      IF amountOfPaths == "1"{
        SET board[i][n] TO "-"
      }
    }}
  RETURN board
}

FUNCTION imperfectify(board, size){
  INTEGER percentage = 5
  LOOP WITH i IN RANGE(size-1){
    LOOP WITH n IN RANGE(size-1){
      IF board[i][n] == "-"{
        ARRAY validDirections = [False, False]
        TRY{
          IF board[i+1][n] == " " AND board[i-1][n] == " "{
            SET validDirections[0] TO TRUE
          }
          IF board[i][n+1] == " " AND board[i][n-1] == " "{
            SET validDirections[1] TO TRUE
          }
        }
        EXCEPT INDEX_ERROR{
          PASS
        }
        IF (validDirections[0] == TRUE AND validDirections[1] == FALSE) OR (validDirections[0] == FALSE AND
validDirections[1] == TRUE){
          IF RANDOM_INTEGER(100) <= percentage{
            SET board[i][n] TO " "
          }
        }
      }
    }
  }
  RETURN board
}

CLASS room{
  CONSTRUCTOR([INTEGER][minSize], [INTEGER][maxSize], [INTEGER][width], [INTEGER][height], [INTEGER][area])
    SET minSize TO 2
    SET maxSize TO 10
    SET width to RANDOM_INTEGER(minSize, maxSize)
    SET height to RANDOM_INTEGER(minSize, maxSize)
  END CONSTRUCTOR

  FUNCTION findValidPlacement(board, size){
    LOOP WITH k IN RANGE(size){
      LOOP WITH z IN RANGE(size){
        IF board[k][z] == "-"{
          INTEGER counter = 0
          LOOP WITH q IN range(height){
            LOOP WITH w IN range(width){
              TRY{
                IF board[k+q][z+w] == "-"{
                  counter++
                }ELSE{
                  BREAK
                }
              }EXCEPT INDEX_ERROR{
                BREAK
              }
            }
          }
          IF counter == area{
            ARRAY roomData = []
            SET roomData TO [k, z, width, height]
            RETURN roomData
          }
        }
      }
    }
    RETURN FALSE
  }

  FUNCTION placeRoom(pos, board){
    LOOP WITH i IN RANGE(self.height-2){
      LOOP WITH n IN RANGE(self.width-2){
        SET board[pos[0] + i + 1][pos[1] + n + 1] TO " "
      }
    }
    RETURN board
  }

  FUNCTION connectRoom(pos, board){
    ARRAY possibleConnections = [[],[],[],[]]
    LOOP WITH i IN RANGE(self.width-2){
      IF board[pos[0] - 1][pos[1] + i + 1] == " " AND board[pos[0] + 1][pos[1] + i + 1] == " "{
        APPEND (STRING(pos[0])+','+STRING(pos[1] + i +1) TO possibleConnections[0] )
      }
      IF board[pos[0] + self.height - 1][pos[1] + i + 1] == " " AND board[pos[0] + self.height + 1][pos[1] + i
+ 1] == " "{
        APPEND (STRING(pos[0] + self.height) + ',' + STRING(pos[1] + i + 1)) TO possibleConnections[1]
      }
    }
    LOOP WITH i IN RANGE(self.height-2){
      IF board[pos[0] + i + 1][pos[1] - 1] == " " AND board[pos[0] + i + 1][pos[1] + 1] == " "{
        APPEND (STRING(pos[0] + i + 1)+','+STRING(pos[1])) TO possibleConnections[2]
      }
      IF board[pos[0] + i + 1][pos[1] - 1 + self.width] == ' ' AND board[pos[0] + i + 1][pos[1] + 1 +
self.width] == ' '{
        APPEND (STRING(pos[0] + i + 1)+','+STRING(pos[1]+self.width)) TO possibleConnections[3]
      }
    }
    LOOP WITH i IN RANGE(4){
      TRY{
        SET DoorCoord TO possibleConnections[i][randint(0,len(possibleConnections[i]))].split(',')
        SET board[int(DoorCoord[0])][int(DoorCoord[1])] TO " "
      }EXCEPT INDEX_ERROR{
        PASS
      }
    RETURN board
    }
  }
```

5.24 - output.py design

```
FROM module PIL IMPORT Image

FUNCTION createImage(board, size){
  CREATE OBJECT img WITH PIL
  img CREATED WITH (RGB, (size, size), (154, 146, 123)) AS PARAMETERS
  LOOP WITH i IN RANGE(size){
    LOOP WITH n IN RANGE(size){
      IF board[n][i] == ' '{
        PLACE PIXEL WITH COLOUR (230,230,205) AT COORDANITES (i,n)
      }
    }
  }
  SAVE IMAGE AS 'STRING(SIZE)' IN FORMAT .bmp
}

FUNCTION createFile(board):
  OPEN FILE ('dungeon.txt', WRITABLE)
  STRING text = ''
  LOOP THROUGH board WITH line{
    LOOP WITH i IN RANGE LENGTH(line){
      IF line[i] == "-"{
        SET text TO text + "[]"
      }ELSE{
        SET text TO text + "
      }
    ADD NEW LINE TO text
    }
  }
```

6.0 - Implementation

See file called 'DundGen - Python Files'
OR
See 'https://repl.it/@LewisFrampton/AH-Project-Dundgen' if you do not wish to run the program locally.

7.0 Testing

7.1 - Final Testing
In all evidence files, the images show the path that I took through the problem and encountered errors. The text and image file are outputs of program if no error occurred, though will only be included if relevant to test.

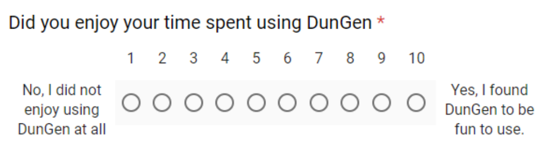
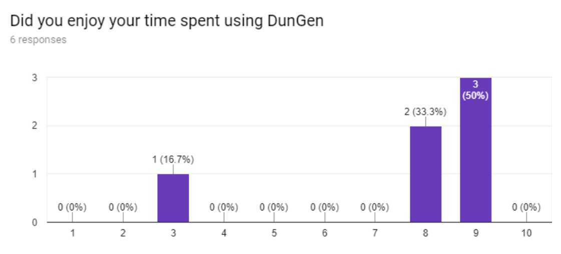
What I am testing	What I expect to happen	What actually happened	How I fixed the issue	Evidence
Does the program when ran, produce a randomly generated dungeon?	The program to create a randomly generated dungeon, filled with rooms and gaps in walls.	Worked as expected	n/a	See test_1_evidence
Does altering the size of the dungeon fact result in a larger dungeon?	Default size is 55x55, so I'm going to see if creating a 125x125 maze creates a larger dungeon.	It did create a larger dungeon.	n/a	See test_2_evidence
Does altering the amount of gaps in the maze in fact result in a more	I expect to see a maze with many more gaps between corridors in it. The maze itself should	Worked as expected. Comparing the final piece of	n/a	See test_3_evidence

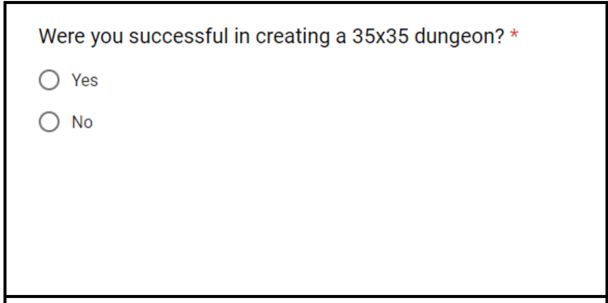
open dungeon?	look more open and have strangely connecting corridors. I will compare a 55x55 dungeon where 'Gaps in dungeon' is set to 1 with a 55x55 dungeon where 'Gaps in dungeon' is set to 15.	evidence in both files located within 'test_3_evidence' shows that when the value was set to 15, it resulted in a much more open dungeon than when the value was set to 1.		
Does the program output an image file correctly?	For an image file, in the format .bmp and named after the size of the dungeon, to be exported. I will test this with default generation settings	Worked as expected.	n/a	See test_4_evidence
Does the program output a text file correctly?	For an image file, in the format .txt and named 'dungeon', to be exported. I will test this with default generation settings	Worked as expected, used the same testing from test_4 as the program did both at the same time.	n/a	See test_5_evidence
Does the dungeon have interconnected rooms?	Run a near-default dungeon generation then look at the finished product to see if anything isn't connected. Gaps in Dungeon will be set to 1 to best test the rooms own ability to connect to nearby corridors/rooms.	Found a room that was not connected to any other rooms.	Error was fixed, turns out I copied and pasted the code for connections on the top and bottom of a room for the sides of a room. Fixed easily by looking at design.	See test_6_evidence
Does the dungeon have interconnected rooms?	Run a near-default dungeon generation then look at the finished product to see if anything isn't connected. Gaps in Dungeon will be set to 1 to best test the rooms own ability to connect to nearby	Worked as expected	n/a	See test_6.1_evidence

	corridors/rooms.			
Does the main menu have working input validation? This will be done by using extreme testing(using ‘0’ and ‘4’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘3’).	In the main menu the program should only accept inputs of the integers 1, 2 or 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the valid input is entered, the program should exit safely.	Worked as expected Though when doing normal testing, I saw that there was a spelling error in the exit message.	Change the exit message from “Thank you for using Dudngen!” to “Thank you for using DunGen!”	See test_7_evidence
Does the option menu have working input validation? This will be done by using extreme testing(using ‘0’ and ‘6’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘5’).	In the option menu the program should only accept inputs of the integers 1, 2, 3, 4, 5 or 6. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the valid input is entered, the option menu should exit back to the main menu.	Worked as expected apart from exceptional testing. Forgot to include validation against string values in this section.	Took the users input and called a validation function. Easy fix as I had already written the code, just had forgotten to include it here.	See test_8_evidence
Does the maze size option have input validation? This will be done by using extreme testing(using ‘0’ and ‘7’ as inputs), exceptional testing(using ‘abc’ as an input) and normal(‘6’).	In the maze size option menu, the program should only accept integers from 1 to 6. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (6), the program will exit back to the option menu	Worked as expected apart from extreme. Entered 7 and it took me back to the option page when it was supposed to simply ask for another answer.	On line 88 of main.py, the second parameter of validateInput was set to 7 when it was meant to be set to 6. Easy fix.	See test_9_evidence
Does the gaps in maze option have input validation? This will be done by using extreme testing(using ‘0’ and ‘4’ as inputs), exceptional testing(using ‘abc’ as an input) and	In the gaps in maze option menu, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (3), the	Worked as expected		See test_10_evidence

normal('3').	program will exit back to the option menu			
Does the Output Image File option have input validation? This will be done by using extreme testing(using '0' and '3' as inputs), exceptional testing(using 'abc' as an input) and normal('3').	In the Output Image File option, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (3), the program will exit back to the option menu	Worked as expected		See test_11_evidence
Does the Output Text File option have input validation? This will be done by using extreme testing(using '0' and '3' as inputs), exceptional testing(using 'abc' as an input) and normal('3').	In the Output Text File option, the program should only accept integers from 1 to 3. When an invalid input is entered, the program will ask the user to re-enter a valid input. When the correct value is entered (3), the program will exit back to the option menu	Worked as expected.		See test_12_evidence

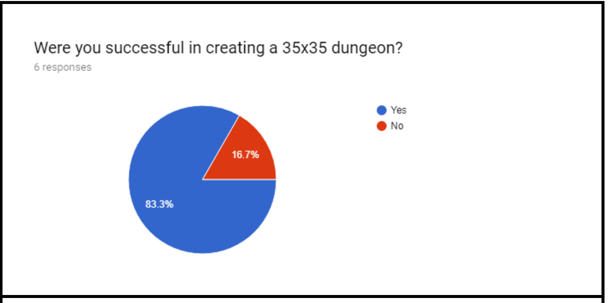
7.2 - End User Survey - Responses

Question	Responses (6 responses in total)
 Did you enjoy your time spent using DunGen * 1 2 3 4 5 6 7 8 9 10 No, I did not enjoy using DunGen at all ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Yes, I found DunGen to be fun to use.	

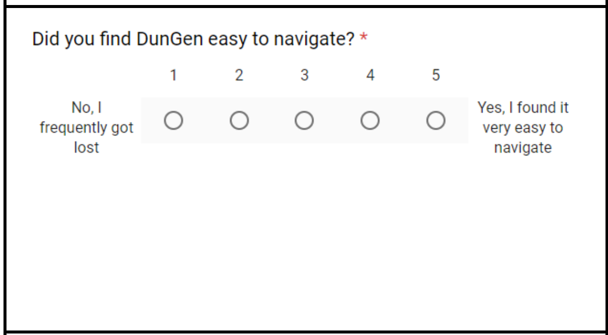

Were you successful in creating a 35x35 dungeon? *

◯ Yes

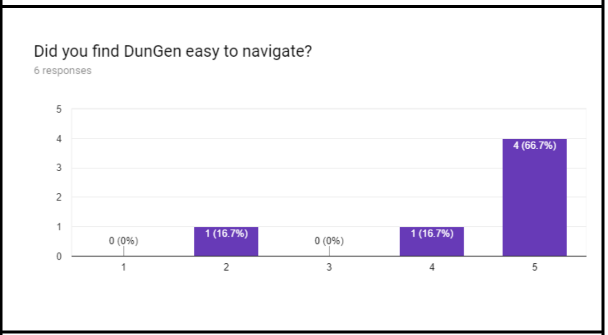
◯ No


Were you successful in creating a 35x35 dungeon?

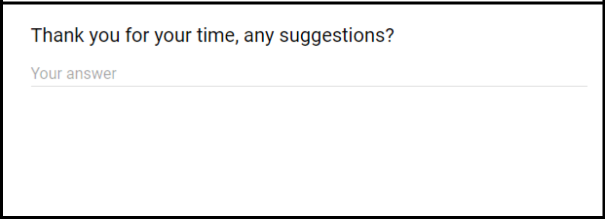
6 responses


Did you find DunGen easy to navigate? *

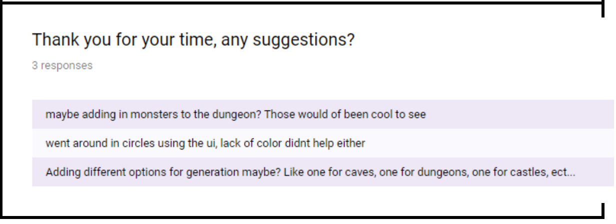
	1	2	3	4	5	
No, I frequently got lost	◯	◯	◯	◯	◯	Yes, I found it very easy to navigate


Did you find DunGen easy to navigate?

6 responses


Thank you for your time, any suggestions?

Your answer


Thank you for your time, any suggestions?

3 responses

maybe adding in monsters to the dungeon? Those would of been cool to see

went around in circles using the ui, lack of color didnt help either

Adding different options for generation maybe? Like one for caves, one for dungeons, one for castles, ect...

8.0 - Evaluation

Now nearing the end of my advanced higher computing coursework, I believe that my project has successfully met most of the predetermined criteria. Overall, I am happy with the work I have produced and have enjoyed working on it. Obviously, I do have some regrets and while happy with my project, it is far from perfect.

8.1 Did it meet the specifications?

I was tasked with designing, creating and maintaining a project of suitibile complexity all while documenting the process. I believe I have achieved this fairly well, which in turns means that I do believe my project meets the specifications.

I'll be the first to admit that my design was not perfect. This was largely due to the fact that when I decided to switch from the maze game into a randomised dungeon program, much of what I had planned for the design had to be scrapped. At this point, I had already wasted numerous weeks on ideas that were now useless which resulted in me rushing the design portion of the new idea. I believe that this hurt my end product as it encouraged a shabby menu system that was quickly thrown together. However, whilst this change in direction may of hurt this aspect of the project I believe it greatly improved other aspects, such as the program itself:

I believe that DundGen is of a suitable complexity for advanced higher. The program contains numerous data structures that are covered in the course as well as a complex algorithm (depth-first search which is used to assist with maze generation: info here). From the numerous 2d arrays that are used throughout the program, the stack which is vital the maze generation and the class which is used to create randomly sized rooms in the dungeon, the program contains more than enough complexity to meet that specification.

The program meets my own specifications as well, with DundGen having all features that I had planned to be in the final build.

8.2 End-User Testing Evaluation

Whilst only managing to contact 6 of the original 20 who answered the survey, I still managed to get valuable information.
I asked each of the 6 who answered to spend 5 minutes of their time with DundGen, tasked with getting it to create a dungeon with the dimensions 35x35. This was chosen as the user would have to navigate the menu system fairly well to change the dungeon size to a custom value. Once those 5 minutes had passed, I asked the user to complete a small survey (seen at **7.2**).

Question 1, “Did you enjoy your time spent using DundGen”?
I was happy to hear that ⅚ users had a positive experience with the project. This question was asked to help me gauge how well I actually did with DundGen, since if nobody even enjoyed using the program then I’d of counted the project as a failure, though a very educational one.

Question 2, “Were you successful in creating a 35x35 dungeon?”?
Again, this question was very useful to help me gauge how well I had done with the project. The fact that ⅚ users were able to successfully complete the task again shows that the project is able to be used fairly well by end-users. The fact that these users were also my target audience again showed that the project could be called a success.

Question 3, “Did you find DundGen easy to navigate?”
Again, ⅚ users had a positive experience with the UI. To me, this means that the UI has served its purpose of being fairly simple which allowed me to focus more on the program itself. At this point though, it’s fairly obvious to spot the pattern with the user who did not have good experience. Though the minority, I consider their feedback the most important as it encourages the most improvement with the project.

Question 4, “Thank you for your time, any suggestions?”
Answer 1: “maybe adding in monsters to the dungeon? Those would have been cool to see”
This suggestion actually was fairly close to becoming a part of the program. If I had more time, I would have loved to of added this to DundGen as it would of added a significant layer of depth to the program and vastly increased the usefulness of the program. However, due to time restraints this had to be removed from the scope.

Answer 2:”went around in circles using the ui, lack of color didnt help either”
It’s fairly safe to assume that this answer came from the user who had a negative experience with DundGen. I asked for more clarification on how I could improve the UI but the user is still to get back to me. I believe that if I had more time, a better interface could of vastly improved the overall user experience.

Answer 3:”Adding different options for generation maybe? Like one for caves, one for dungeons, one for castles, ect…”
This was a brilliant idea which like the others, I wish I could have added to DundGen if I only had more time to work on it.

In conclusion, users seemed to have an overall positive experience with DundGen. They believed that the program could use updates to improve the interface and add new features and I agree; DundGen could of use significant improvements which would only add to the user experience with the program. Again, if only I had more time to work on the project then I could of vastly improved it.

8.3 - Final Testing

Through my final testing, I know my project works and I believe that it is robust. As far as I know, I have caught all bugs that had to do with input validation or bugs that could cause the program to crash. However, I know that I have likely missed something. Working as a single-person team, it becomes very hard to fully test your program as you start to only test for bugs that you’d expect. Especially with the randomised aspect of DundGen, bugs could arise that I just never encountered due to the generation pattern. As a result of those factors, I can only hope that I’ve caught the larger bugs in the program through my testing.

8.4 - Further Developments

As mentioned in **8.1**, there are many things I wish I could have added to the project if only I had enough time. However, there’s no point in saying never while they could be added at some point in the future. The ideas for further development include:

- Improving the User Interface. This aspect seemed to be a large issue with the end-users, with one user stating that they got lost in it. Perhaps I could streamline the interface or create a new one altogether.

- Populating the dungeon with random monsters. The fact that I ran out of time and had to scrap this aspect really frustrates me. This addition could have added so much more depth to the program whilst also being perfectly in theme. This feature allows for so much more to be built on top of it as well, for example if a creature is only found in a cave, the dungeon should reflect that. It’d be the same if a creature is only found in the sea, or forest, or desert, or ruin, or ect… This leads us onto:

- More generation options! Allow the user to set a theme and have the dungeon reflect that! For example, if the user wanted an old castle then the dungeon should have a good mix of small rooms and very large rooms with straight passages between them. If the user wanted a cave instead, widen some of the passages and make rooms rounded, creating a sort of cavern look to them.

- Improve code efficiency! I’ll happily admit that the code that currently makes up DundGen is horribly inefficient; For example, the section of the code which find valid room placement takes a very long time if you slightly increase the dungeon size. I could spend time trying to figure out a more efficient system that would hopefully not only make the program more robust but also improve the user experience as they would be able to create much larger dungeons with a fair amount of ease.

- Clean up code! While I’m happy with the program I’ve created, I know parts of it are very weakly held together. I’ve read that using the statement ‘except: pass’ tends to be bad practice since instead of fixing an issue with the program, you just tell it to ignore the issue and keep running. Unfortunately, I use this 5 times in the program when I expect an Index_Error to be encountered. If I had more time, I could look at what is causing these errors and try to fix that instead of just telling the program to ignore them completely. Though this wouldn’t improve the user experience as the program works fine as it is, it would make future developments easier and would be good programming practice.

8.5 - Retrospective

Whilst on the topic of having spaghetti code, the development of this program was a maze of bad decisions. A clear example of this are the movement variables in dungeonGen.py (line 7 and line 8) and in mazeGen.py(line 7 and line 8). These variables allow me to loop through an IF statement 4 times, changing the applied direction each time. The reason for this very slight optimization? I couldn’t be bothered writing out 4 if statements each time I needed to validate directions. This in turn left me working on this ‘optimization’ for a good few hours whilst I could of avoided it by just writing some if statements. The result of this was much more work, harder to read code and even more semi-useless variables.
However, I hope I’ve learned from my mistakes. The situation above was an example of both laziness and useless optimization. If I was to work on a project of similar scale in the future, I would try to ensure that I would try to focus on the bigger picture and get more work done instead of constantly spending valuable time working on complex solutions for very simple problems.

Not only did this project expose me to some issues that are commonly encountered in the development cycle, it also taught me a great deal:

- **Focus on the bigger picture.** I often tried to work on section of the program that would never be used or worse, had no impact on the end product anyways. This took away valuable time that could of been spent implementing features (such as those within **8.4**), which resulted in hurting the project
- **How to manage time when working over the course of several months.** This was an issue for me. I often found it so much more tempting just to ignore the design and documentation and instead work on the implementation aspect. As a result, though I’m happy with the program side of the project, I feel as though the rest of it is lacking. If I was to work on a similar scale project at some point in the future, I would ensure that all parts of the program got the same attention as the implementation did as focusing too much one aspect of a large multi-part project ends up severely hurting the end product.
- **How to use many new data types and a new algorithm.** When I began work on the project, I had no idea what a Stack was, let alone knowing how to use one. By the end of the project, I can safely say that I am confident that creating DundGen has taught me how to use Stacks, 2d Arrays and classes with ease. I also was introduced to the algorithm of a Depth-First Search. Whilst I am not entirely confident using a the algorithm, I understand how it works in relation to DundGen.

- **That my code has improved over the course of the project.** As I've gained more experience with programming and whilst I was being introduced to more complex programming concepts, I definitely feel as though I have improved as a programmer.
- **How to limit your projects scope.** This was another large issue for me. Originally I had ideas of populating the dungeon with randomised creatures; allowing for more complex generation methods; adding colour and images to the outputs and possibly allowing the user to play through the dungeon a game. However, these all had to be scrapped and removed from my scope due to time constraints. This resulted in me wasting the time spent planning and researching these ideas, which took time away from the more important aspects of the program. In future products, I'll try to come with more grounded expectations on what I can achieve within time constraints.

8.6

To conclude, to me DundGen has been a success. Not only has it been very educational on the development cycle, it has improved my skills as a programmer, allowed me to experience working on a large project independently, taught me valuable time-management skills and been enjoyable to work on.

Though my documentation could of been improved, I feel as though it does a good job in conveying the project to the reader. My design was accurate to the end product and according the user surveys, was useable to most. I'm happy with the work done on the program itself, though my code could be improved in some areas I feel as though it is mostly of a high quality. My testing showed that this code was fairly robust and helped to find some simple bugs, which were easily caught and fixed. The end user survey showed that a majority of users enjoyed using the program, which I believe to show that DundGen has been a success.