500611960

Section 1

1)

- Gives us confidence that our program will perform correctly.
- Future-proofs our programs. Regression testing ensures new changes do not cause problems in the existing program.
- Makes developing software easier, we can raise informative exceptions to provide insight into where our code has gone wrong.

2)

Mocking is used primarily for unit testing. When a unit being tested relies on other complex units, we can create a 'mock' unit, which can simulate these other complex units in controlled ways.

Mocking allows faster testing, as we don't have to rely on connections with external services. Mocking allows expectations to be set. This means that the mock objects know how they are supposed to be called by the test object and how to respond to the test object.

It can take time to learn and set up mocks, especially if they each require specific expectations to be set. Mocks are a replacement for real use scenarios and may not account for all possible situations.

3)

The Therac-25 radiation therapy machine was an update to the Therac-20 which relied heavily on hardware based safety mechanisms. Some of the software from the Therac-20 was reused in the Therac-25. A combination of overconfident engineers and unresolved software bugs caused six patients to receive fatal overdoses of radiation [4].

Section 2

1)

BFS finds the shortest path, but generally uses more memory than DFS as it must hold in memory all the child nodes at each level of the graph [2, 3]. If the destination is deep within the search tree, BFS may take too much time.

BFS is useful when we need to find the single shortest path. BFS is best suited for applications where the root is close to the destination vertices we are searching for [1].

2)

DFS suits applications where the source may be far from the destination [1]. DFS generally requires less memory, as it doesn't require storing all the child nodes at each level of the tree [3]. If we want to find

the shortest path, a DFS algorithm may take a much longer time and the solution found may not be the shortest. [2]

DFS is often used within games or puzzles, where scenarios may require exploring all the paths created as a result of a decision made by the user, terminating if a win state is encountered [1].

3)

If the ending cell is close to the starting cell, it is more likely that BFS will be faster, as it traverses all the closest cells first. If a deep path is taken before reaching the destination cell, DFS can take a long time to reach a close solution.

4)

We may be required to revisit (or 'wait' at) a teleporter in order to successfully finish the game (e.g. Locked in a room test-case). Depending how the board is designed, we may be required to traverse a cell in order to get to a water bucket, but then need to backtrack via this same cell in order to proceed through fire to solve the maze.

References

[1] (BFS)?, W., Störr, H., Vaity, Y., Sugumar, K. and Johnson, N., 2020. *When Is It Practical To Use Depth-First Search (DFS) Vs Breadth-First Search (BFS)?*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/3332947/when-is-it-practical-to-use-depth-first-search-dfs-vs-breadth-first-search-bf> [Accessed 28 May 2020].

[2] GeeksforGeeks. 2020. *Difference Between BFS And DFS - Geeksforgeeks*. [online] Available at: <https://www.geeksforgeeks.org/difference-between-bfs-and-dfs/> [Accessed 28 May 2020].

[3] Jiang, B., 1991. Traversing graphs in a paging environment, BFS or DFS?. *Information Processing Letters*, 37(3), pp.143-147.

[4] Leveson, N. and Turner, C., 1993. An investigation of the Therac-25 accidents. *Computer*, 26(7), pp.18-41.