Estrutura de dados

A principal estruturá de dados utilizada para solução do problema foram os arrays. Esta estrutura foi escolhida pois é a que eu possuo maior familiaridade.

Algoritmos utilizados para resolver o problema

A minha solução do problema foi inspirada no algoritmo Breadth-First Search (BFS), que utiliza árvores e camadas para resolver o problema. Porém, ao invés de utilizar árvores para guardar as camadas, eu utilizei um array. Abaixo uma imagem que representa como a árvore foi representada por um array no meu código.

Pseudo-código

Este pseudo código corresponde ao código que realiza o preenchimento das camadas:

Para cada PV inicial cria-se um array que contém um array de inteiros, onde o PV inicial, [1], fica na camada 0.

A partir do PV inicial descobrimos os próximos PVs possíveis, [4, 5], que serão inseridos na camada 1.

Para cada próximo PV, descobrimos mais PVs alcançáveis, [6,7,8,9], que serão inseridos na camada 2.

Repetimos este processo até alcançar o limite de temperatura ou até não existir um próximo PV acessível a partir da última camada.

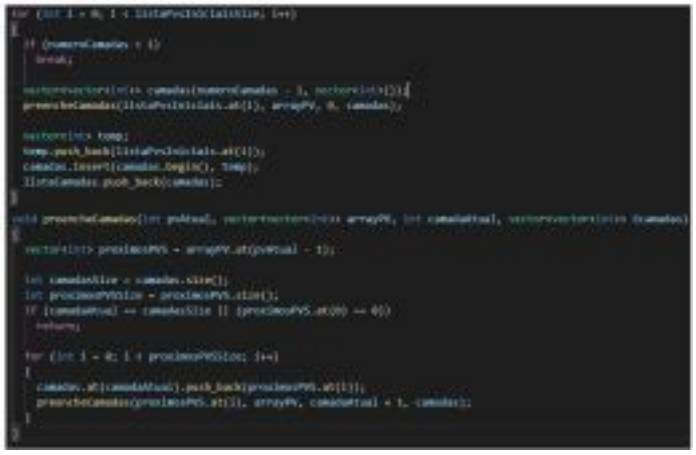
Análise de complexidade assintótica

Acredito que o meu código possui duas partes importantes em que faz sentido analisar a complexidade: a **função que preenche as camadas** e a **função que preenche o resultado**.

- **função que preenche as camadas:**

Aqui temos um for que executa m vezes, onde m é o número de PVs imediatamente acessíveis a partir dos CVs. Este for chama a função preencheCamadas, que é uma função recursiva.

A função preencheCamadas é recursiva e irá executar **n** vezes no pior caso, (quando chegaremos no limite de temperatura em todas as camadas), onde **n** é o número de camadas que teremos. Porém, temos que para cada **m** haverá uma execução separada da função preencheCamadas, logo temos uma complexidade de **O(n*m)**.



- **função que preenche o resultado:**

Como eu utilizei uma estrutura de dados com três arrays, eu preciso de 3 for's para percorrer a variável arrayCamadas, que será transformada no resultado que é impresso após o fim da execução do código. Como temos 3 for's aninhados, fica claro que temos uma complexidade $O(n^3)$.