Summarizing the content:

1. Summaries the main points in this module. You may include references to the learning objectives.
2. Introduction to Greedy Algorithms.
3. Greedy Scheduling Solution.
4. Huffman coding.
5. Minimum Spanning Tree.
6. Prim’s Algorithm.
7. Kruskal Algorithm.
8. How is this useful?

greedy algorithm a special type of algorithm used to solve optimization problems by deriving the maximum or minimum value for a specific instance. The algorithm selects the best possible result for the current scenario independent of the current result. Greedy algorithms are usually implemented for scenarios with specific conditions. This algorithm is used to solve optimization problems, maximization problems and minimization problems. And provide feasible or optimized solutions. Some problem scenarios that might be best suited for greedy algorithms, such as Huffman coding, minimum spanning tree graphs using Prim's or Kruskal's algorithms, and finding the shortest path between two vertices in a graph.

1. How do you plan to use this information?

Minimize the problem: As long as all conditions are met, the problem can be solved easily. However, when this problem requires a minimum requirement, it is called a minimization problem.

Maximization problem: A problem that requires a maximum outcome is called a maximization problem.

Optimization problem: When a problem requires a minimum or maximum result, the problem is called an optimization problem.

1. Provide summary of your reading list — external resources, websites, book chapters, code libraries, etc.
2. <https://en.wikipedia.org/wiki/Huffman_coding>
3. <https://www.programiz.com/dsa/greedy-algorithm>
4. <https://en.wikipedia.org/wiki/Minimum_spanning_tree>

**Reflecting on the content:**

1. What is the most important thing you learnt in this module?

At each step in the algorithm, make a choice.

* Hey, I can increase my activity set by one.
* And leave lots of room for future choices.
* Let’s do that and hope for the best!!!

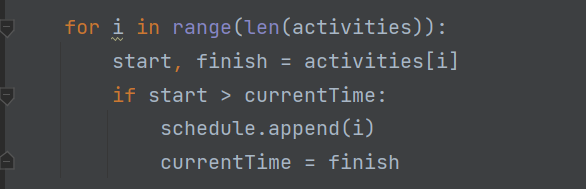
Hope that at the end of the day, this results in a globally optimal solution.

1. How does this relate to what you already know?

Greedy algorithms have many similarities with dynamic programming. In particular, the problem to which the greedy algorithm applies is also an optimal substructure. A significant difference between greedy algorithm and dynamic programming is that in greedy algorithm, the optimal substructure is used in a top-down manner. Instead of finding the optimal solution to the subproblem first, then making a choice, a greedy algorithm will first make a choice that appears to be the optimal choice at the time, and then solve a resulting subproblem.

Activity1:

1. Analyze the greedy algorithm for Activity Selection problem in this week's lab. Demonstrate your understanding of the code.



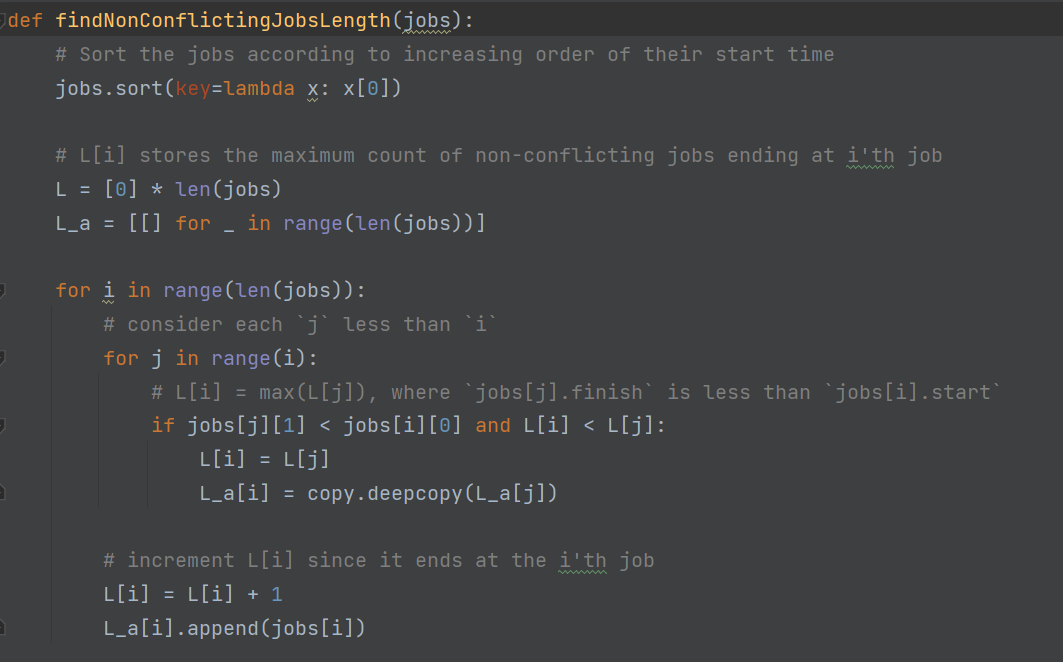
Iterates over all activities and if the start time is greater than or equal to the current time, do the activity.

Solution:

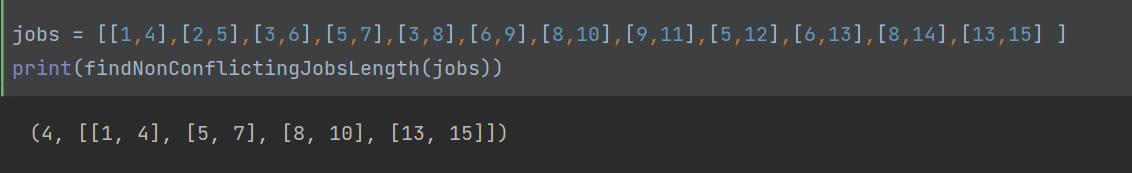


Activity2: Devise a dynamic programming solution to Activity Selection problem, and see if you get the same results as that of greedy strategy.

Code:



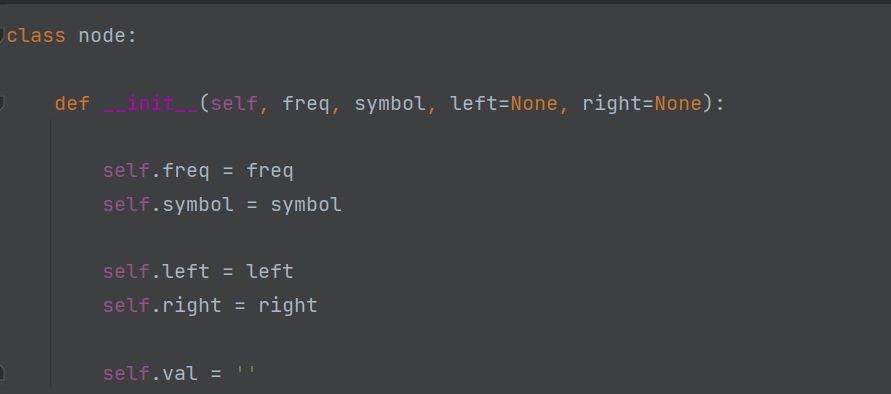
Result:



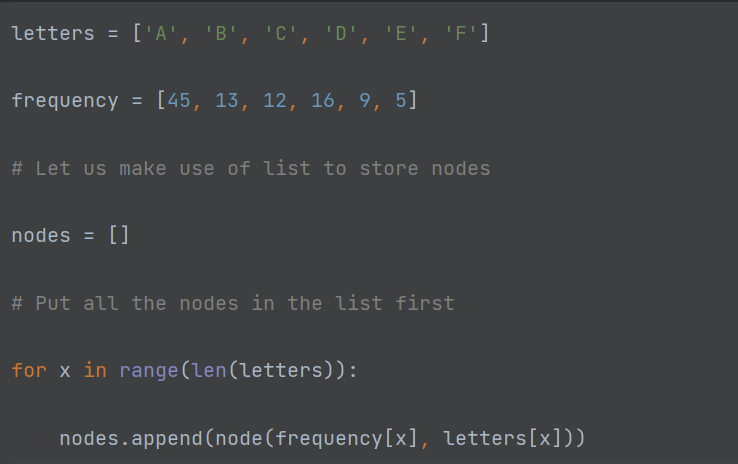
Compared with the greedy algorithm, the result is the same.

Activity3: Have a look at the associated ipython notebook and run the code for building Prefix Code Trees. Demonstrate your understanding of the code by (doing some of the following activities):

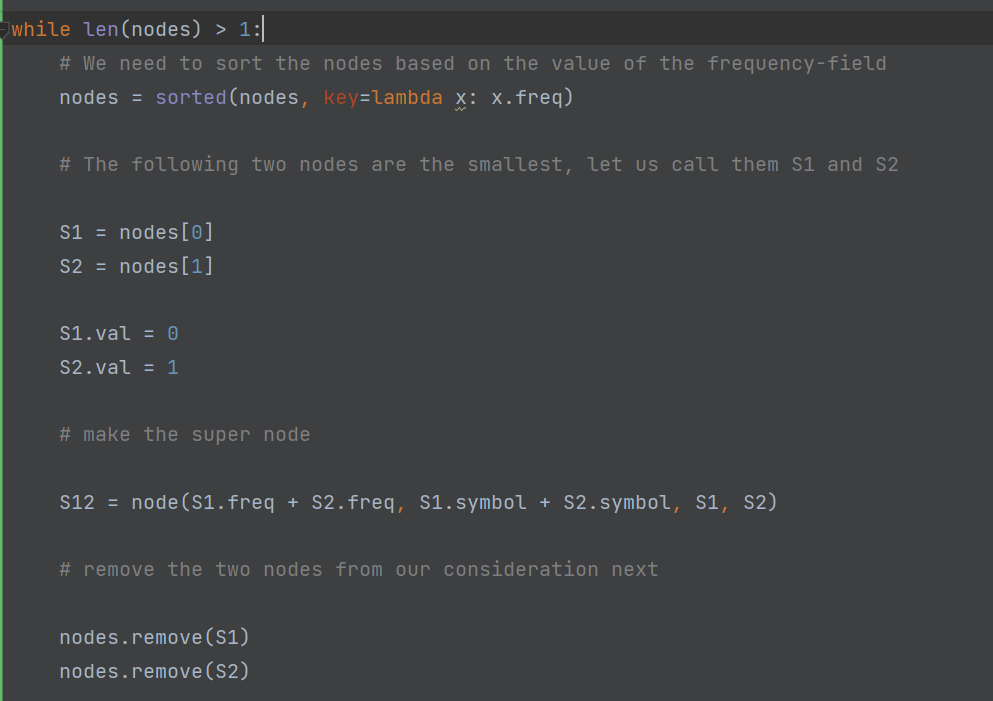
Define tree nodes:



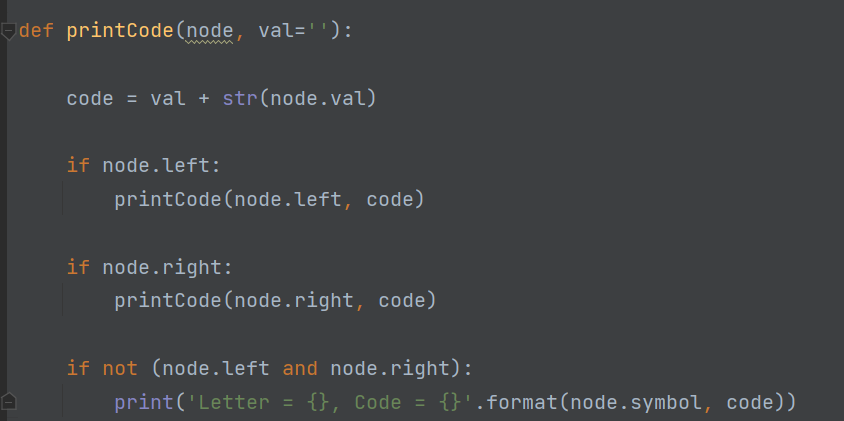
letter and frequency giving definition:



Merge the two least frequent nodes and generate a new root node:

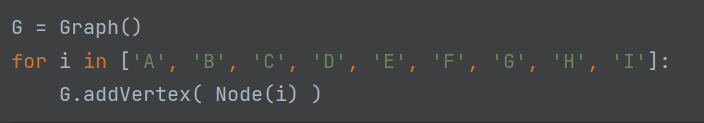


Print the code of each letter:

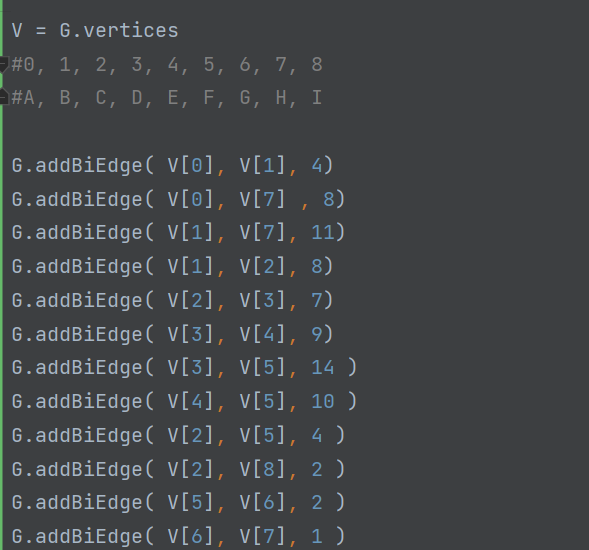


Activity4: Have a look at the associated ipython notebook and run the code for slow and optimized versions of Prim's Algorithm. Demonstrate your understanding of the code by (doing some of the following activities):

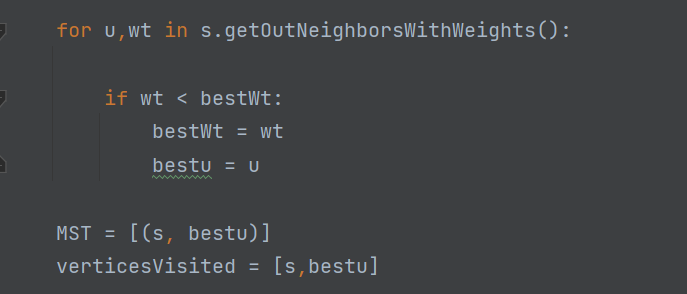
Define vertex:



Define edge:



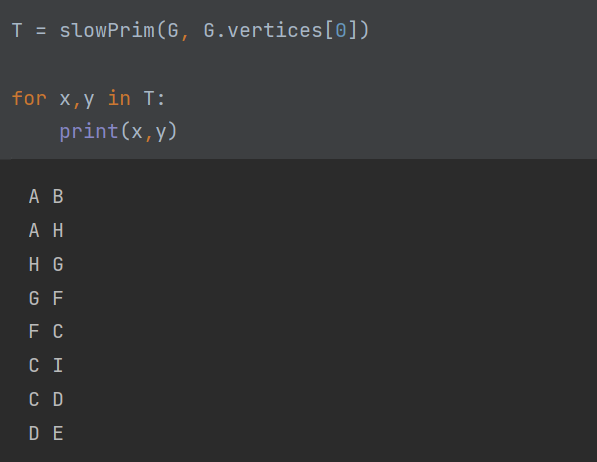
First, find the lightest edge leaving s.



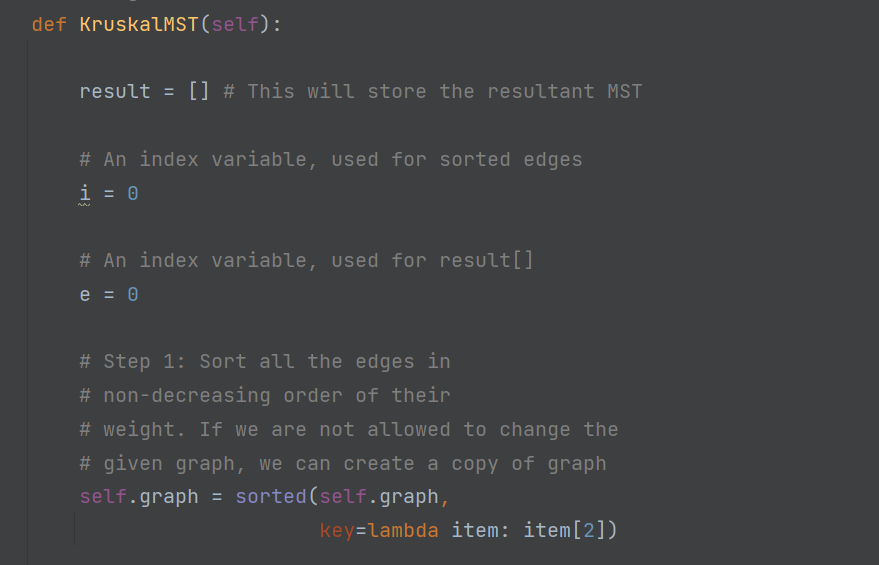
Find the lighter edge from the searched nodes:



The result of running the code:



Code:



Result:

