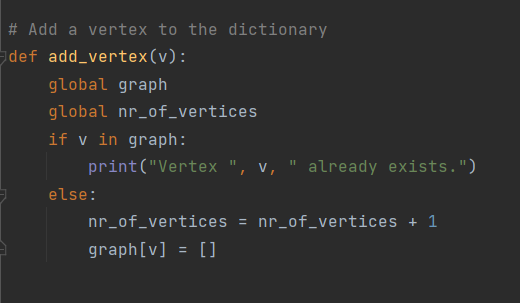
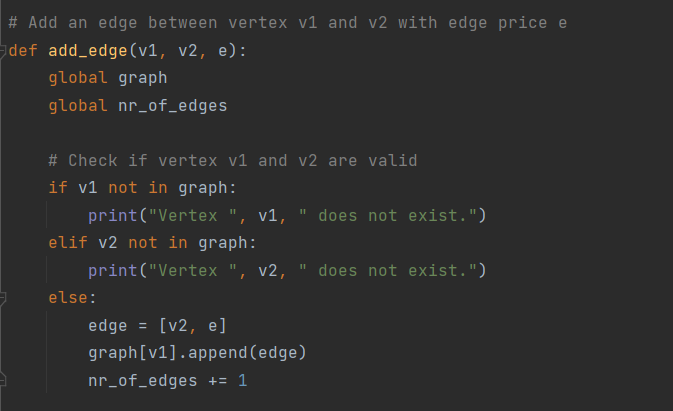
**Jorza Ionut Documentation**

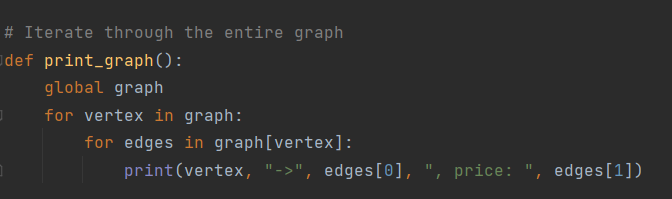
The program uses a ‘graph’ dictionary to store the vertices, edges and prices by the form: {0: [1, 20]} – which means there is an edge between the vertices 0 and 1 with the price of 20.



Here we have the add vertex function. We are adding a new vertex to the dictionary and incrementing the number of vertices by 1 every time a new vertex is added.



The add edge function is used so that we can add an edge between 2 vertices. We first check if those 2 vertices are valid, which means that they are part of the graph, if they are then we can add the edge to the graph.

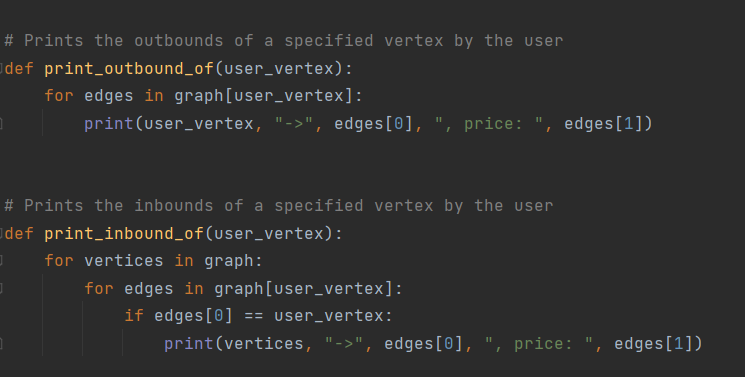


This function is used to print the entire graph by the form:

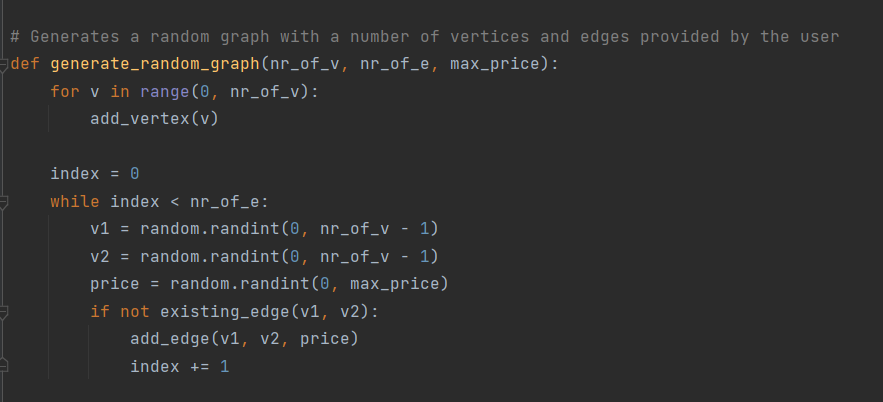
Ex: 0->1, price: 20

Which means there is edge between vertices 0 and 1 with a price of 20.

We go through each vertex and each of its edges and print them on the screen.

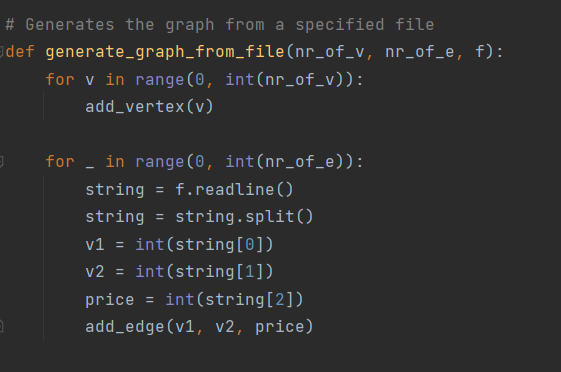


In order to print the outbounds and inbounds of a specified vertex, we use 2 functions that prints them. These go through each of the edges and checks if there exist an outbound or inbound with the specified vertex, if it exists, then we print it.



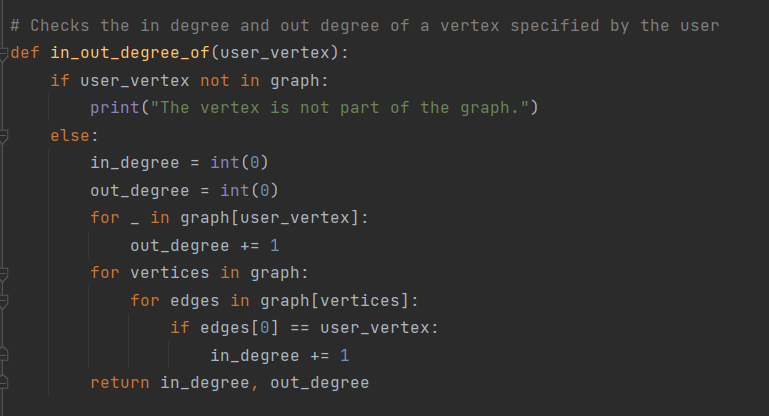
This function is used to generate a random graph, using the number of vertices, the number of edges and the max price an edge can have. All of these are provided by the user.

We use a random generator that generates edges until the number of edges is hit.



This function is used to generate a graph from a file.

Again, we use the number of vertices and the number of edges specified in the file, but this time we don’t add random edges, we add the specified edges from the file.



This function is used so that we determine the in degree and out degree of a specified vertex by the user. We only print the degree, not the actual edge. As you can see, we go through each edge and check if the user vertex is part of that edge by being in or out.

This is the program. We can create and modify vertices, edges, prices. The menu is well-written and you can run the program and understand it without even reading the code.