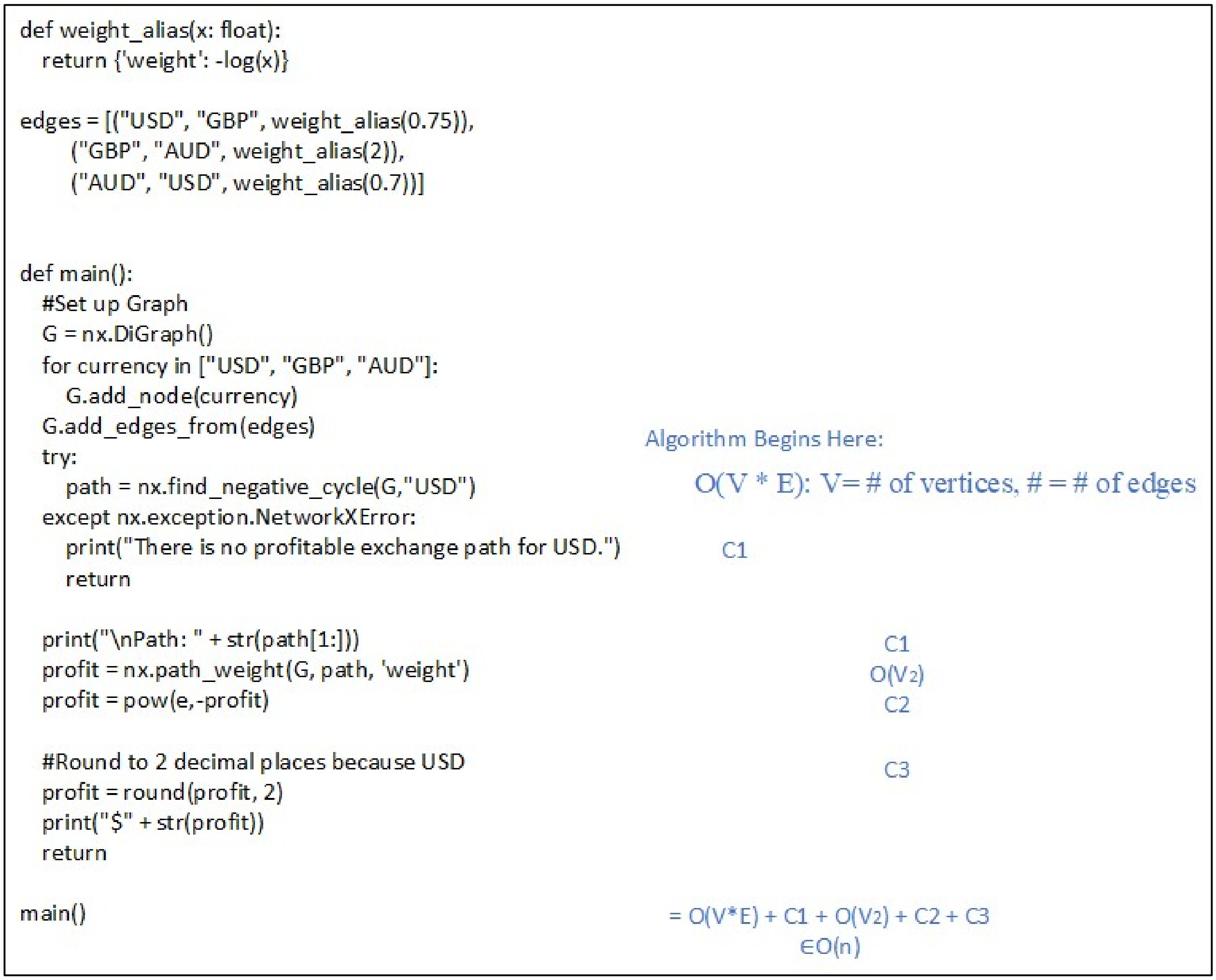
Hayden Coble
hcoble@my.athens.edu
00101613

Assignment 5

1. Arbitrage.py



```
def weight_alias(x: float):
   return {'weight': -log(x)}

edges = [("USD", "GBP", weight_alias(0.75)),
       ("GBP", "AUD", weight_alias(2)),
       ("AUD", "USD", weight_alias(0.7))]


def main():
   #Set up Graph
   G = nx.DiGraph()
   for currency in ["USD", "GBP", "AUD"]:
      G.add_node(currency)
   G.add_edges_from(edges)                        Algorithm Begins Here:
   try:
      path = nx.find_negative_cycle(G,"USD")         O(V * E): V= # of vertices, # = # of edges
   except nx.exception.NetworkXError:
      print("There is no profitable exchange path for USD.")     C1
      return

   print("\nPath: " + str(path[1:]))                           C1
   profit = nx.path_weight(G, path, 'weight')                  O(V2)
   profit = pow(e,-profit)                                     C2

   #Round to 2 decimal places because USD                      C3
   profit = round(profit, 2)
   print("$" + str(profit))
   return

main()                                      = O(V*E) + C1 + O(V2) + C2 + C3
                                                       ∈O(n)
```

2. Prim's algorithm visits nodes multiple times to get the minimum distance/lowest cost/lowest weight whereas Kruskal's algorithm traverses each node once. Because of this behavior, Prim's algorithm is ideal for graphs with many nodes with smaller weights (dense graphs) and Kruskal's is ideal for graphs with greater weights and potentially less nodes (sparse graphs).
3. See prim_kruskal.py
4. The algorithm for making change operates on the assumption that we have an infinite amount of all coins. If we have 0 nickels, dimes, or quarters (or any other coins of greater value), the algorithm can still work by giving enough pennies equal to the amount covered by the missing coin(s). However, if we have 0 pennies the algorithm will fail. This is because n (the value to obtain) is found by adding one coin (d_j) to the amount n – d_j until the condition n >= d_j fails.

Hayden Coble
hcoble@my.athens.edu
00101613

4.1. Suppose we need to make 9 cents but have no pennies. On the first iteration, we would add a nickel, and the remaining change is four cents. There is no coin available that would satisfy the condition 4 >= (coin value).