1) Stadt0 = Wiesbaden Wiesbaden -> Mainz -> Frankfurt -> Berlin -> München Strecke = [Wiesbaden, Mainz, Frankfurt, Berlin, München]

2) #Eingabedaten Städte = [Wiesbaden, Main, Frankfurt, Berlin, München] #Stadt ist eine Struktur die, die aktuelle Stadt enthält #die von der momentanenStadt aus erreichbaren Städte sowie #deren zugehörige Streckenlängen (um die kürzeste zu finden) #Diese Struktur muss für alle Städte aus der Liste "Städte" existieren Stadt = {momentaneStadt, Zielstädte, StreckenlängenZuZielstädte} Strecke = [] Stadt0 = Wiesbaden while Städte: #solange die Liste Städte Elemente enthält kürzesteStadt = kürzesteStadt(Stadt0) Strecke.append(kürzesteStadt) #kürzesteStadt zur Liste hinzufügen Städte.remove(kürzesteStadt) #kürzesteStadt aus "Städte" enfernen return Strecke

Aufgabe 4) Sieb des Erathostenes theoretische Aufgaben: 1) 1) 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 2) 2 3 — 5 — 7 — 9 —— 11 —— 13 —— 15 —— 17 3) 2 3 — 5 — 7 — — —— 11 —— 13 —— —— —— 17 2) Der Algorithmus terminiert, denn er fügt alle Primzahlen zu einer Liste hinzu und es gibt eine endliche Anzahl von Primzahlen.

Der Algorithmus ist determiniert, weil sich die Primzahlen nicht ändern.

Der Algorithmus ist deterministisch, denn alle Zwischenergebnisse sind gleichen Startbedigungen immer gleich.

Der Algorithmus ist korrekt, denn er ist terminierend und liefert das gewollte Ergebnis zurück, nämlich eine Liste mit Primzahlen bis m