find_Route.py input.txt lubeck Bremen

Src = node = Lubeck
dest = graphnode = Bremen
graph = {one city with it's neighbouring city}

generated = 0
expanded = 0
fringe = POO

fringe = {(0, 'lubeck')}

visited = {'lubeck' : ("",0)}
Parsed = []
max_node = 0

ubeck' : ("", 0), 'Hamburg' : ("lubeck", 63)
Bremen' : ('Hamburg', 179), 'Hannover' : ('Hamburg', 246),
Berlin' : (Hamburg, 354),

i> while : not empty :

[redacted]

_ , node_count = 0 , Lubecks

expanded = 1

if Lubeck == Bremen :

No

if Lubeck in parsed :

No

parsed = ['lubeck']

for , in {'Hamburg': 63.0} :

generated = 1

fringe = {('63', Hamburg)}

in accepted :

while not empty:

_, nodecount = '63', Hamburg

expanded = 2

if Hamburg == Bermen:

No

if Hamburg in parsed:

No

parsed = ['luebeck', 'Hamburg']

for i in {'luebeck': 63, Bremen: 116, Hannover: 153, Berlin: 291},

generated = 2

fringe = {(126, luebeck)}

for it in {—; Bremen: 116, —}

generated = 3

fringe = {(126, luebeck), (179, Bremen)}

if Bremen not in visited:

visited[Bremen] = (Hamburg, 179)

for i in {—; "Hannover": 153; —}

generated = 3

fringe = {—, (216, Hamburg)}

Hannover not in visited:

yes

for i in {— "Bremen": 291}

generated = 4

while not empty:

[illegible]

—, node-count = 126, 'Luebeck')

expanded = 3

if —1—

no

if [illegible] :

yes.

[illegible]

while not empty:

[illegible]

, node count = 179
80, Bremen

expanded = 4

if Bremen = no

yes.

Break;

route: [Hamburg, Luebeck)

if Bremen is visited:

Yes.

distance = 0.0

node-count = Bremen

while Bremen != luebeck:

distance = 116

node-count = Hamburg

distance = 116 + 63 = 179

node-count = luebeck.