DMDD Final Project Simulation
Zixiao 20200328

Simulation

Content

Abstract

In this simulation, we didn't consider multithreading, which means we assume there is only one cumstomer is using the application.

So the main goal for this simulatio is to test if the database can store the data and interact with each other correctly.

The database for this simulation is as following:

This simulation include following things:

- 1. Test if the database can store right data into tables:
  - Map data simulation
    - `building`
    - `floorMap`
    - `floorMapPoints`
    - `locationToVisit`
    - `locationToChangeFloor`
    - `beacons`
  - User information simulation
    - `adminUser`
    - `customer`
- 2. Test if the connections between device and beacon are generated correctly
  - `device`
  - `deviceBeaconConnection`
- 3. Test if the positioning is updated correctly
  - `deviceCurrentPosition`
- 4. Test if the navigation is generated correctly
  - `navigationRequest`
- 5. Test if the paths are generated correctly
  - `singleFloorPath`
  - `singleFloorPathPoints`
  - `floorAlongtheWay`
  - `beginFloor`
  - `midFloor`
  - `endFloor`
  - `midFloorChangeCombination`
  - `pathCombination`
- 6. Test if the system is update correctly when navigation request is over
  - `navigationComplete`
  - `navigationRequest`

Connect to database

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
# import the magic code for using sql in jupyter notebook
%load_ext sql
```

```
# local database
# %sql mysql+pymysql://root:fjwwzx970814@localhost/mydb
# remote database
%sql mysql+pymysql://brickea_mac:fjwWZX970814@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
```

```
'Connected: brickea_mac@mydb'
```

```
%%sql
show tables
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
22 rows affected.
```

Tables_in_mydb
adminuser
beacons
beginfloor
building
customer
device
devicebeaconconnection
devicecurrentposition
endfloor
flooralongtheway
floormap
floormappoints
locationtochangefloor
locationtovisit
midfloor
midfloorchangecombination
navigationcomplete
navigationrequest
pathcombination
singlefloorpath
singlefloorpathpoints
view_buildings

```
import sqlalchemy as sqlManager
```

```
# Create connection with database
connection = sqlManager.create_engine('mysql+pymysql://brickea_mac:fjwWZX970814@rm-
0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb?charset=utf8')
```

Map and user data simulation and test

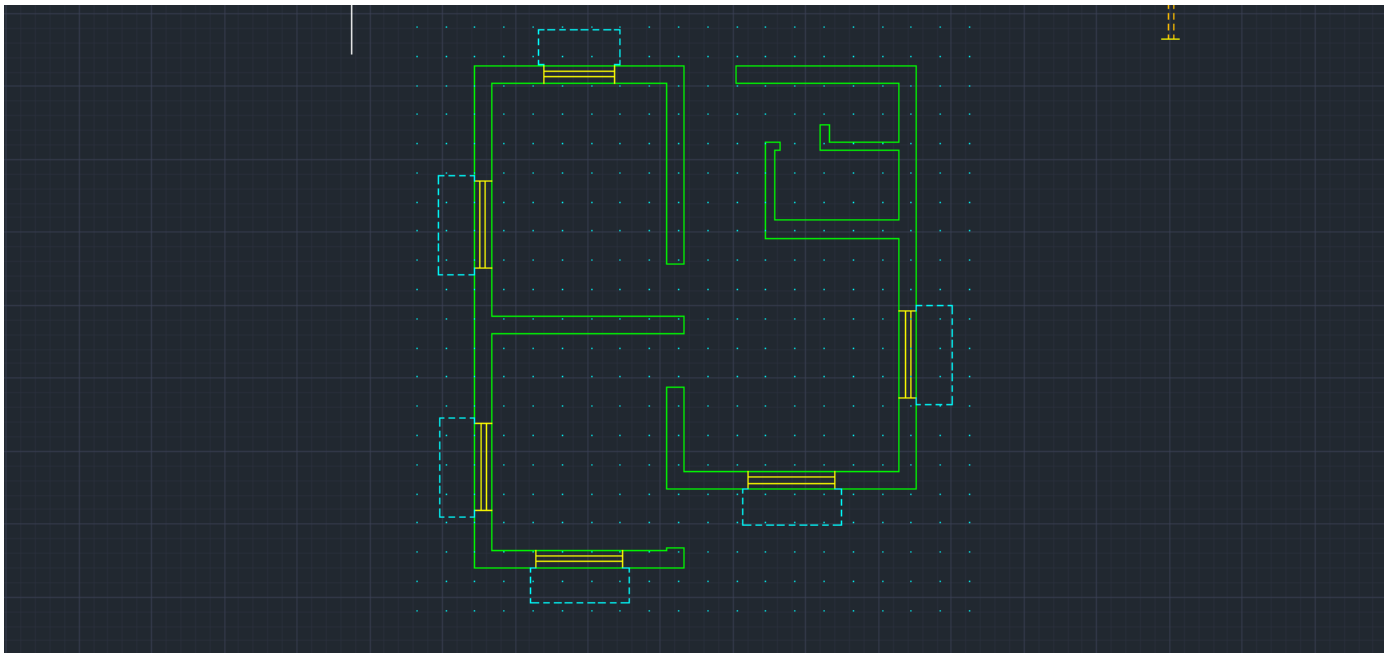
Map data simulation

Because I only got one map sample as following

So I will use it as a template to generate 5 different floor maps for 2 different buildings

And I assume the coordinate original in the map is at its top left

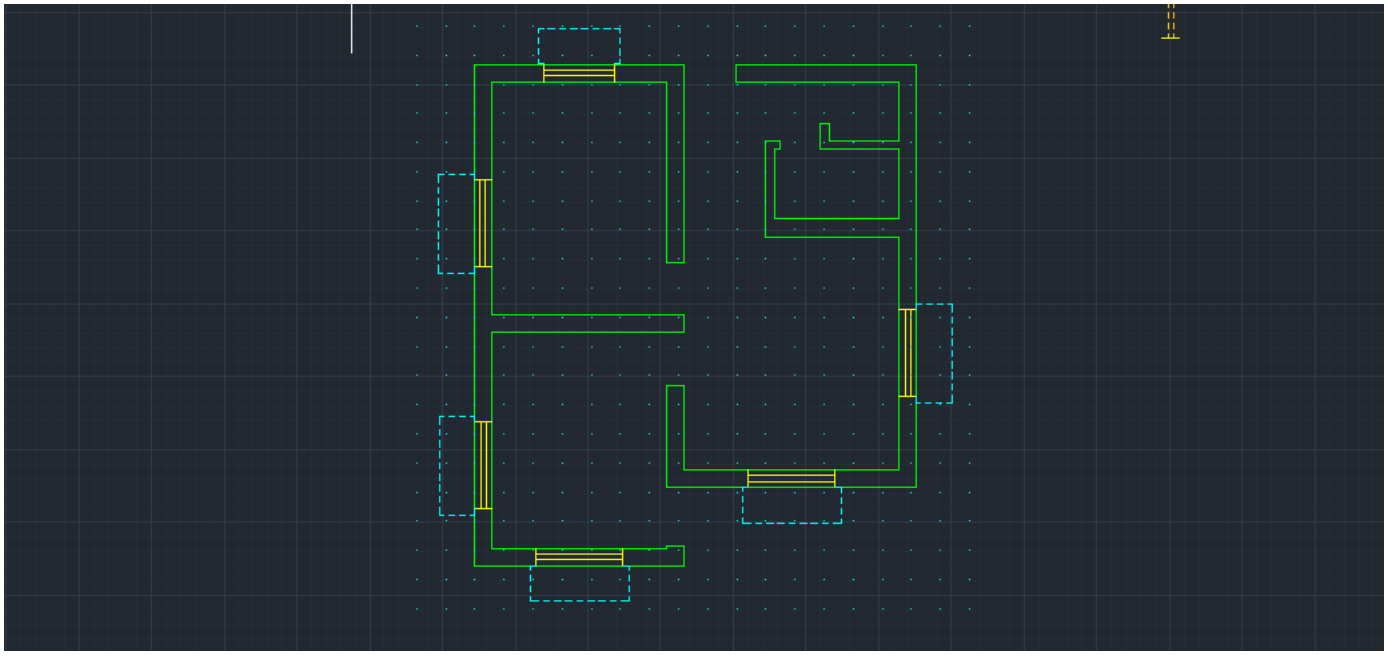
The map is 21 X 20



Here I assume

- points within green line is a wall
- points within red circle is an elevator
- points within purple circle is a stair
- elevator and stair are in the same position for every floor
- elevator and stair can help human move to any floor

building - table test



```
# Generate building data
building = pd.DataFrame([
    [1,'building_1','address_1','description'],
    [2,'building_2','address_2','description'],
],columns=['idbuilding','name','address','description'])
building
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idbuilding	name	address	description
0	1	building_1	address_1	description
1	2	building_2	address_2	description

```
# Insert data into database
pd.io.sql.to_sql(building,'building',connection,schema='mydb',if_exists='append',index=False)
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pymysql/cursors.py:170: Warning: (3719,
"'utf8' is currently an alias for the character set UTF8MB3, but will be an alias for UTF8MB4 in a
future release. Please consider using UTF8MB4 in order to be unambiguous.")
  result = self._query(query)
```

```
%%sql
select * from building
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
2 rows affected.
```

idbuilding	name	address	description
1	building_1	address_1	description
2	building_2	address_2	description

floorMap - table test

```
def generate_floor_info(floor_number=[]):
    result = []
    floor_id = 0
    for building in range(len(floor_number)):
        for floor in range(floor_number[building]):
            result.append([floor_id+1,building+1,floor+1,
('building_'+str(building+1)+'_floor_'+str(floor+1)),True])
            floor_id+=1
    return result
```

```
floor_data = generate_floor_info(floor_number=[3,2])
floor_data
```

```
[[1, 1, 1, 'building_1_floor_1', True],
 [2, 1, 2, 'building_1_floor_2', True],
 [3, 1, 3, 'building_1_floor_3', True],
 [4, 2, 1, 'building_2_floor_1', True],
 [5, 2, 2, 'building_2_floor_2', True]]
```

```
floor_columns = ['idfloorMap','idbuilding','floorNumber','excelMap','ifupdate']
```

```
# Generate floor data for each building
floor = pd.DataFrame(data=floor_data,columns=floor_columns)
floor
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idfloorMap	idbuilding	floorNumber	excelMap	ifupdate
0	1	1	1	building_1_floor_1	True
1	2	1	2	building_1_floor_2	True
2	3	1	3	building_1_floor_3	True
3	4	2	1	building_2_floor_1	True
4	5	2	2	building_2_floor_2	True

```
# Insert data into database
pd.io.sql.to_sql(floor,'floorMap',connection,schema='mydb',if_exists='append',index=False)
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pandas/io/sql.py:1336: UserWarning: The
provided table name 'floorMap' is not found exactly as such in the database after writing the table,
possibly due to case sensitivity issues. Consider using lower case table names.
  warnings.warn(msg, UserWarning)
```

```
%%sql
select * from floorMap
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
5 rows affected.
```

idfloorMap	idbuilding	floorNumber	excelMap	ifupdate
1	1	1	b'building_1_floor_1'	1
2	1	2	b'building_1_floor_2'	1
3	1	3	b'building_1_floor_3'	1
4	2	1	b'building_2_floor_1'	1
5	2	2	b'building_2_floor_2'	1

floorMapPoints - table test

Here I assume bottom left is the coordinate original point

```
np.zeros((3,3))
```

```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
def generate_floor_basic_points(x_len,y_len):
    return np.zeros((x_len,y_len))
```

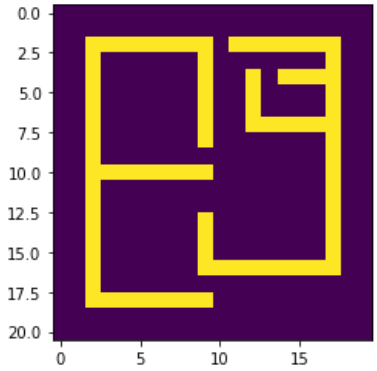
```
# The map is 20 X 21 (unit:meter)
floor_map_data = generate_floor_basic_points(21,20)
```

```
def generate_wall(floor_map,wall=[]):
    # wall should input coordinate of start point and end point
    # wall = [(start_x,start_y),(end_x,end_y)]
    start = wall[0]
    end = wall[1]
    for list_x in range(start[1],end[1]+1):
        for list_y in range(start[0],end[0]+1):
            floor_map[list_x][list_y] = 1
    return floor_map
```

```
# Generate wall
floor_map_data = generate_wall(floor_map_data,wall = [(2,2),(9,2)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,2),(9,8)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,2),(2,18)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,18),(9,18)])
floor_map_data = generate_wall(floor_map_data,wall = [(2,10),(9,10)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,13),(9,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(9,16),(17,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(11,2),(17,2)])
floor_map_data = generate_wall(floor_map_data,wall = [(17,2),(17,16)])
floor_map_data = generate_wall(floor_map_data,wall = [(14,4),(17,4)])
floor_map_data = generate_wall(floor_map_data,wall = [(12,4),(12,7)])
floor_map_data = generate_wall(floor_map_data,wall = [(12,7),(17,7)])
```

```
plt.imshow(floor_map_data)
```

```
<matplotlib.image.AxesImage at 0x1258f5210>
```




```
def generate_floor_map_point(floor_map,floor_id_list):
    result = []
    id_coordinate = 1
    for floor_id in floor_id_list:
        for y, y_values in enumerate(floor_map):
            for x, value in enumerate(y_values):
                passable = True if value == 0 else False
                result.append([id_coordinate,floor_id,x,21-1-y,passable])
                id_coordinate+=1
    return result
```

```
floor_map_points_data = generate_floor_map_point(floor_map_data,[1,2,3,4,5])
floor_map_points_columns = ['coordinatePoint','idfloorMap','x','y','ifpassable']
```

```
len(floor_map_points_data)
```

```
2100
```

```
20*21*5
```

```
2100
```

```
floor_map_points = pd.DataFrame(data = floor_map_points_data,columns=floor_map_points_columns)
```

```
# Insert data into database
pd.io.sql.to_sql(floor_map_points,'floorMapPoints',connection,schema='mydb',if_exists='append',index
=False)
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pymysql/cursors.py:170: Warning: (3719,
"'utf8' is currently an alias for the character set UTF8MB3, but will be an alias for UTF8MB4 in a
future release. Please consider using UTF8MB4 in order to be unambiguous.")
  result = self._query(query)
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pandas/io/sql.py:1336: UserWarning: The
provided table name 'floorMapPoints' is not found exactly as such in the database after writing the
table, possibly due to case sensitivity issues. Consider using lower case table names.
  warnings.warn(msg, UserWarning)
```

```
%%sql
select * from floorMapPoints
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
2100 rows affected.
```

coordinatePoint	idfloorMap	x	y	ifpassable
1	1	0	20	1
2	1	1	20	1
3	1	2	20	1
4	1	3	20	1
5	1	4	20	1
6	1	5	20	1
7	1	6	20	1
8	1	7	20	1
9	1	8	20	1
10	1	9	20	1
11	1	10	20	1
12	1	11	20	1
13	1	12	20	1
14	1	13	20	1
15	1	14	20	1
16	1	15	20	1
17	1	16	20	1
18	1	17	20	1
19	1	18	20	1
20	1	19	20	1
21	1	0	19	1
22	1	1	19	1
23	1	2	19	1
24	1	3	19	1
25	1	4	19	1
26	1	5	19	1
27	1	6	19	1
28	1	7	19	1
29	1	8	19	1
30	1	9	19	1

31	1	10	19	1
32	1	11	19	1
33	1	12	19	1
34	1	13	19	1
35	1	14	19	1
36	1	15	19	1
37	1	16	19	1
38	1	17	19	1
39	1	18	19	1
40	1	19	19	1
41	1	0	18	1
42	1	1	18	1
43	1	2	18	0
44	1	3	18	0
45	1	4	18	0
46	1	5	18	0
47	1	6	18	0
48	1	7	18	0
49	1	8	18	0
50	1	9	18	0
51	1	10	18	1
52	1	11	18	0
53	1	12	18	0
54	1	13	18	0
55	1	14	18	0
56	1	15	18	0
57	1	16	18	0
58	1	17	18	0
59	1	18	18	1
60	1	19	18	1
61	1	0	17	1
62	1	1	17	1
63	1	2	17	0
64	1	3	17	1
65	1	4	17	1
66	1	5	17	1
67	1	6	17	1
68	1	7	17	1
69	1	8	17	1
70	1	9	17	0

71	1	10	17	1
72	1	11	17	1
73	1	12	17	1
74	1	13	17	1
75	1	14	17	1
76	1	15	17	1
77	1	16	17	1
78	1	17	17	0
79	1	18	17	1
80	1	19	17	1
81	1	0	16	1
82	1	1	16	1
83	1	2	16	0
84	1	3	16	1
85	1	4	16	1
86	1	5	16	1
87	1	6	16	1
88	1	7	16	1
89	1	8	16	1
90	1	9	16	0
91	1	10	16	1
92	1	11	16	1
93	1	12	16	0
94	1	13	16	1
95	1	14	16	0
96	1	15	16	0
97	1	16	16	0
98	1	17	16	0
99	1	18	16	1
100	1	19	16	1
101	1	0	15	1
102	1	1	15	1
103	1	2	15	0
104	1	3	15	1
105	1	4	15	1
106	1	5	15	1
107	1	6	15	1
108	1	7	15	1
109	1	8	15	1
110	1	9	15	0

111	1	10	15	1
112	1	11	15	1
113	1	12	15	0
114	1	13	15	1
115	1	14	15	1
116	1	15	15	1
117	1	16	15	1
118	1	17	15	0
119	1	18	15	1
120	1	19	15	1
121	1	0	14	1
122	1	1	14	1
123	1	2	14	0
124	1	3	14	1
125	1	4	14	1
126	1	5	14	1
127	1	6	14	1
128	1	7	14	1
129	1	8	14	1
130	1	9	14	0
131	1	10	14	1
132	1	11	14	1
133	1	12	14	0
134	1	13	14	1
135	1	14	14	1
136	1	15	14	1
137	1	16	14	1
138	1	17	14	0
139	1	18	14	1
140	1	19	14	1
141	1	0	13	1
142	1	1	13	1
143	1	2	13	0
144	1	3	13	1
145	1	4	13	1
146	1	5	13	1
147	1	6	13	1
148	1	7	13	1
149	1	8	13	1
150	1	9	13	0

151	1	10	13	1
152	1	11	13	1
153	1	12	13	0
154	1	13	13	0
155	1	14	13	0
156	1	15	13	0
157	1	16	13	0
158	1	17	13	0
159	1	18	13	1
160	1	19	13	1
161	1	0	12	1
162	1	1	12	1
163	1	2	12	0
164	1	3	12	1
165	1	4	12	1
166	1	5	12	1
167	1	6	12	1
168	1	7	12	1
169	1	8	12	1
170	1	9	12	0
171	1	10	12	1
172	1	11	12	1
173	1	12	12	1
174	1	13	12	1
175	1	14	12	1
176	1	15	12	1
177	1	16	12	1
178	1	17	12	0
179	1	18	12	1
180	1	19	12	1
181	1	0	11	1
182	1	1	11	1
183	1	2	11	0
184	1	3	11	1
185	1	4	11	1
186	1	5	11	1
187	1	6	11	1
188	1	7	11	1
189	1	8	11	1
190	1	9	11	1

191	1	10	11	1
192	1	11	11	1
193	1	12	11	1
194	1	13	11	1
195	1	14	11	1
196	1	15	11	1
197	1	16	11	1
198	1	17	11	0
199	1	18	11	1
200	1	19	11	1
201	1	0	10	1
202	1	1	10	1
203	1	2	10	0
204	1	3	10	0
205	1	4	10	0
206	1	5	10	0
207	1	6	10	0
208	1	7	10	0
209	1	8	10	0
210	1	9	10	0
211	1	10	10	1
212	1	11	10	1
213	1	12	10	1
214	1	13	10	1
215	1	14	10	1
216	1	15	10	1
217	1	16	10	1
218	1	17	10	0
219	1	18	10	1
220	1	19	10	1
221	1	0	9	1
222	1	1	9	1
223	1	2	9	0
224	1	3	9	1
225	1	4	9	1
226	1	5	9	1
227	1	6	9	1
228	1	7	9	1
229	1	8	9	1
230	1	9	9	1

231	1	10	9	1
232	1	11	9	1
233	1	12	9	1
234	1	13	9	1
235	1	14	9	1
236	1	15	9	1
237	1	16	9	1
238	1	17	9	0
239	1	18	9	1
240	1	19	9	1
241	1	0	8	1
242	1	1	8	1
243	1	2	8	0
244	1	3	8	1
245	1	4	8	1
246	1	5	8	1
247	1	6	8	1
248	1	7	8	1
249	1	8	8	1
250	1	9	8	1
251	1	10	8	1
252	1	11	8	1
253	1	12	8	1
254	1	13	8	1
255	1	14	8	1
256	1	15	8	1
257	1	16	8	1
258	1	17	8	0
259	1	18	8	1
260	1	19	8	1
261	1	0	7	1
262	1	1	7	1
263	1	2	7	0
264	1	3	7	1
265	1	4	7	1
266	1	5	7	1
267	1	6	7	1
268	1	7	7	1
269	1	8	7	1
270	1	9	7	0

271	1	10	7	1
272	1	11	7	1
273	1	12	7	1
274	1	13	7	1
275	1	14	7	1
276	1	15	7	1
277	1	16	7	1
278	1	17	7	0
279	1	18	7	1
280	1	19	7	1
281	1	0	6	1
282	1	1	6	1
283	1	2	6	0
284	1	3	6	1
285	1	4	6	1
286	1	5	6	1
287	1	6	6	1
288	1	7	6	1
289	1	8	6	1
290	1	9	6	0
291	1	10	6	1
292	1	11	6	1
293	1	12	6	1
294	1	13	6	1
295	1	14	6	1
296	1	15	6	1
297	1	16	6	1
298	1	17	6	0
299	1	18	6	1
300	1	19	6	1
301	1	0	5	1
302	1	1	5	1
303	1	2	5	0
304	1	3	5	1
305	1	4	5	1
306	1	5	5	1
307	1	6	5	1
308	1	7	5	1
309	1	8	5	1
310	1	9	5	0

311	1	10	5	1
312	1	11	5	1
313	1	12	5	1
314	1	13	5	1
315	1	14	5	1
316	1	15	5	1
317	1	16	5	1
318	1	17	5	0
319	1	18	5	1
320	1	19	5	1
321	1	0	4	1
322	1	1	4	1
323	1	2	4	0
324	1	3	4	1
325	1	4	4	1
326	1	5	4	1
327	1	6	4	1
328	1	7	4	1
329	1	8	4	1
330	1	9	4	0
331	1	10	4	0
332	1	11	4	0
333	1	12	4	0
334	1	13	4	0
335	1	14	4	0
336	1	15	4	0
337	1	16	4	0
338	1	17	4	0
339	1	18	4	1
340	1	19	4	1
341	1	0	3	1
342	1	1	3	1
343	1	2	3	0
344	1	3	3	1
345	1	4	3	1
346	1	5	3	1
347	1	6	3	1
348	1	7	3	1
349	1	8	3	1
350	1	9	3	1

351	1	10	3	1
352	1	11	3	1
353	1	12	3	1
354	1	13	3	1
355	1	14	3	1
356	1	15	3	1
357	1	16	3	1
358	1	17	3	1
359	1	18	3	1
360	1	19	3	1
361	1	0	2	1
362	1	1	2	1
363	1	2	2	0
364	1	3	2	0
365	1	4	2	0
366	1	5	2	0
367	1	6	2	0
368	1	7	2	0
369	1	8	2	0
370	1	9	2	0
371	1	10	2	1
372	1	11	2	1
373	1	12	2	1
374	1	13	2	1
375	1	14	2	1
376	1	15	2	1
377	1	16	2	1
378	1	17	2	1
379	1	18	2	1
380	1	19	2	1
381	1	0	1	1
382	1	1	1	1
383	1	2	1	1
384	1	3	1	1
385	1	4	1	1
386	1	5	1	1
387	1	6	1	1
388	1	7	1	1
389	1	8	1	1
390	1	9	1	1

391	1	10	1	1
392	1	11	1	1
393	1	12	1	1
394	1	13	1	1
395	1	14	1	1
396	1	15	1	1
397	1	16	1	1
398	1	17	1	1
399	1	18	1	1
400	1	19	1	1
401	1	0	0	1
402	1	1	0	1
403	1	2	0	1
404	1	3	0	1
405	1	4	0	1
406	1	5	0	1
407	1	6	0	1
408	1	7	0	1
409	1	8	0	1
410	1	9	0	1
411	1	10	0	1
412	1	11	0	1
413	1	12	0	1
414	1	13	0	1
415	1	14	0	1
416	1	15	0	1
417	1	16	0	1
418	1	17	0	1
419	1	18	0	1
420	1	19	0	1
421	2	0	20	1
422	2	1	20	1
423	2	2	20	1
424	2	3	20	1
425	2	4	20	1
426	2	5	20	1
427	2	6	20	1
428	2	7	20	1
429	2	8	20	1
430	2	9	20	1

431	2	10	20	1
432	2	11	20	1
433	2	12	20	1
434	2	13	20	1
435	2	14	20	1
436	2	15	20	1
437	2	16	20	1
438	2	17	20	1
439	2	18	20	1
440	2	19	20	1
441	2	0	19	1
442	2	1	19	1
443	2	2	19	1
444	2	3	19	1
445	2	4	19	1
446	2	5	19	1
447	2	6	19	1
448	2	7	19	1
449	2	8	19	1
450	2	9	19	1
451	2	10	19	1
452	2	11	19	1
453	2	12	19	1
454	2	13	19	1
455	2	14	19	1
456	2	15	19	1
457	2	16	19	1
458	2	17	19	1
459	2	18	19	1
460	2	19	19	1
461	2	0	18	1
462	2	1	18	1
463	2	2	18	0
464	2	3	18	0
465	2	4	18	0
466	2	5	18	0
467	2	6	18	0
468	2	7	18	0
469	2	8	18	0
470	2	9	18	0

471	2	10	18	1
472	2	11	18	0
473	2	12	18	0
474	2	13	18	0
475	2	14	18	0
476	2	15	18	0
477	2	16	18	0
478	2	17	18	0
479	2	18	18	1
480	2	19	18	1
481	2	0	17	1
482	2	1	17	1
483	2	2	17	0
484	2	3	17	1
485	2	4	17	1
486	2	5	17	1
487	2	6	17	1
488	2	7	17	1
489	2	8	17	1
490	2	9	17	0
491	2	10	17	1
492	2	11	17	1
493	2	12	17	1
494	2	13	17	1
495	2	14	17	1
496	2	15	17	1
497	2	16	17	1
498	2	17	17	0
499	2	18	17	1
500	2	19	17	1
501	2	0	16	1
502	2	1	16	1
503	2	2	16	0
504	2	3	16	1
505	2	4	16	1
506	2	5	16	1
507	2	6	16	1
508	2	7	16	1
509	2	8	16	1
510	2	9	16	0

511	2	10	16	1
512	2	11	16	1
513	2	12	16	0
514	2	13	16	1
515	2	14	16	0
516	2	15	16	0
517	2	16	16	0
518	2	17	16	0
519	2	18	16	1
520	2	19	16	1
521	2	0	15	1
522	2	1	15	1
523	2	2	15	0
524	2	3	15	1
525	2	4	15	1
526	2	5	15	1
527	2	6	15	1
528	2	7	15	1
529	2	8	15	1
530	2	9	15	0
531	2	10	15	1
532	2	11	15	1
533	2	12	15	0
534	2	13	15	1
535	2	14	15	1
536	2	15	15	1
537	2	16	15	1
538	2	17	15	0
539	2	18	15	1
540	2	19	15	1
541	2	0	14	1
542	2	1	14	1
543	2	2	14	0
544	2	3	14	1
545	2	4	14	1
546	2	5	14	1
547	2	6	14	1
548	2	7	14	1
549	2	8	14	1
550	2	9	14	0

551	2	10	14	1
552	2	11	14	1
553	2	12	14	0
554	2	13	14	1
555	2	14	14	1
556	2	15	14	1
557	2	16	14	1
558	2	17	14	0
559	2	18	14	1
560	2	19	14	1
561	2	0	13	1
562	2	1	13	1
563	2	2	13	0
564	2	3	13	1
565	2	4	13	1
566	2	5	13	1
567	2	6	13	1
568	2	7	13	1
569	2	8	13	1
570	2	9	13	0
571	2	10	13	1
572	2	11	13	1
573	2	12	13	0
574	2	13	13	0
575	2	14	13	0
576	2	15	13	0
577	2	16	13	0
578	2	17	13	0
579	2	18	13	1
580	2	19	13	1
581	2	0	12	1
582	2	1	12	1
583	2	2	12	0
584	2	3	12	1
585	2	4	12	1
586	2	5	12	1
587	2	6	12	1
588	2	7	12	1
589	2	8	12	1
590	2	9	12	0

591	2	10	12	1
592	2	11	12	1
593	2	12	12	1
594	2	13	12	1
595	2	14	12	1
596	2	15	12	1
597	2	16	12	1
598	2	17	12	0
599	2	18	12	1
600	2	19	12	1
601	2	0	11	1
602	2	1	11	1
603	2	2	11	0
604	2	3	11	1
605	2	4	11	1
606	2	5	11	1
607	2	6	11	1
608	2	7	11	1
609	2	8	11	1
610	2	9	11	1
611	2	10	11	1
612	2	11	11	1
613	2	12	11	1
614	2	13	11	1
615	2	14	11	1
616	2	15	11	1
617	2	16	11	1
618	2	17	11	0
619	2	18	11	1
620	2	19	11	1
621	2	0	10	1
622	2	1	10	1
623	2	2	10	0
624	2	3	10	0
625	2	4	10	0
626	2	5	10	0
627	2	6	10	0
628	2	7	10	0
629	2	8	10	0
630	2	9	10	0

631	2	10	10	1
632	2	11	10	1
633	2	12	10	1
634	2	13	10	1
635	2	14	10	1
636	2	15	10	1
637	2	16	10	1
638	2	17	10	0
639	2	18	10	1
640	2	19	10	1
641	2	0	9	1
642	2	1	9	1
643	2	2	9	0
644	2	3	9	1
645	2	4	9	1
646	2	5	9	1
647	2	6	9	1
648	2	7	9	1
649	2	8	9	1
650	2	9	9	1
651	2	10	9	1
652	2	11	9	1
653	2	12	9	1
654	2	13	9	1
655	2	14	9	1
656	2	15	9	1
657	2	16	9	1
658	2	17	9	0
659	2	18	9	1
660	2	19	9	1
661	2	0	8	1
662	2	1	8	1
663	2	2	8	0
664	2	3	8	1
665	2	4	8	1
666	2	5	8	1
667	2	6	8	1
668	2	7	8	1
669	2	8	8	1
670	2	9	8	1

671	2	10	8	1
672	2	11	8	1
673	2	12	8	1
674	2	13	8	1
675	2	14	8	1
676	2	15	8	1
677	2	16	8	1
678	2	17	8	0
679	2	18	8	1
680	2	19	8	1
681	2	0	7	1
682	2	1	7	1
683	2	2	7	0
684	2	3	7	1
685	2	4	7	1
686	2	5	7	1
687	2	6	7	1
688	2	7	7	1
689	2	8	7	1
690	2	9	7	0
691	2	10	7	1
692	2	11	7	1
693	2	12	7	1
694	2	13	7	1
695	2	14	7	1
696	2	15	7	1
697	2	16	7	1
698	2	17	7	0
699	2	18	7	1
700	2	19	7	1
701	2	0	6	1
702	2	1	6	1
703	2	2	6	0
704	2	3	6	1
705	2	4	6	1
706	2	5	6	1
707	2	6	6	1
708	2	7	6	1
709	2	8	6	1
710	2	9	6	0

711	2	10	6	1
712	2	11	6	1
713	2	12	6	1
714	2	13	6	1
715	2	14	6	1
716	2	15	6	1
717	2	16	6	1
718	2	17	6	0
719	2	18	6	1
720	2	19	6	1
721	2	0	5	1
722	2	1	5	1
723	2	2	5	0
724	2	3	5	1
725	2	4	5	1
726	2	5	5	1
727	2	6	5	1
728	2	7	5	1
729	2	8	5	1
730	2	9	5	0
731	2	10	5	1
732	2	11	5	1
733	2	12	5	1
734	2	13	5	1
735	2	14	5	1
736	2	15	5	1
737	2	16	5	1
738	2	17	5	0
739	2	18	5	1
740	2	19	5	1
741	2	0	4	1
742	2	1	4	1
743	2	2	4	0
744	2	3	4	1
745	2	4	4	1
746	2	5	4	1
747	2	6	4	1
748	2	7	4	1
749	2	8	4	1
750	2	9	4	0

751	2	10	4	0
752	2	11	4	0
753	2	12	4	0
754	2	13	4	0
755	2	14	4	0
756	2	15	4	0
757	2	16	4	0
758	2	17	4	0
759	2	18	4	1
760	2	19	4	1
761	2	0	3	1
762	2	1	3	1
763	2	2	3	0
764	2	3	3	1
765	2	4	3	1
766	2	5	3	1
767	2	6	3	1
768	2	7	3	1
769	2	8	3	1
770	2	9	3	1
771	2	10	3	1
772	2	11	3	1
773	2	12	3	1
774	2	13	3	1
775	2	14	3	1
776	2	15	3	1
777	2	16	3	1
778	2	17	3	1
779	2	18	3	1
780	2	19	3	1
781	2	0	2	1
782	2	1	2	1
783	2	2	2	0
784	2	3	2	0
785	2	4	2	0
786	2	5	2	0
787	2	6	2	0
788	2	7	2	0
789	2	8	2	0
790	2	9	2	0

791	2	10	2	1
792	2	11	2	1
793	2	12	2	1
794	2	13	2	1
795	2	14	2	1
796	2	15	2	1
797	2	16	2	1
798	2	17	2	1
799	2	18	2	1
800	2	19	2	1
801	2	0	1	1
802	2	1	1	1
803	2	2	1	1
804	2	3	1	1
805	2	4	1	1
806	2	5	1	1
807	2	6	1	1
808	2	7	1	1
809	2	8	1	1
810	2	9	1	1
811	2	10	1	1
812	2	11	1	1
813	2	12	1	1
814	2	13	1	1
815	2	14	1	1
816	2	15	1	1
817	2	16	1	1
818	2	17	1	1
819	2	18	1	1
820	2	19	1	1
821	2	0	0	1
822	2	1	0	1
823	2	2	0	1
824	2	3	0	1
825	2	4	0	1
826	2	5	0	1
827	2	6	0	1
828	2	7	0	1
829	2	8	0	1
830	2	9	0	1

831	2	10	0	1
832	2	11	0	1
833	2	12	0	1
834	2	13	0	1
835	2	14	0	1
836	2	15	0	1
837	2	16	0	1
838	2	17	0	1
839	2	18	0	1
840	2	19	0	1
841	3	0	20	1
842	3	1	20	1
843	3	2	20	1
844	3	3	20	1
845	3	4	20	1
846	3	5	20	1
847	3	6	20	1
848	3	7	20	1
849	3	8	20	1
850	3	9	20	1
851	3	10	20	1
852	3	11	20	1
853	3	12	20	1
854	3	13	20	1
855	3	14	20	1
856	3	15	20	1
857	3	16	20	1
858	3	17	20	1
859	3	18	20	1
860	3	19	20	1
861	3	0	19	1
862	3	1	19	1
863	3	2	19	1
864	3	3	19	1
865	3	4	19	1
866	3	5	19	1
867	3	6	19	1
868	3	7	19	1
869	3	8	19	1
870	3	9	19	1

871	3	10	19	1
872	3	11	19	1
873	3	12	19	1
874	3	13	19	1
875	3	14	19	1
876	3	15	19	1
877	3	16	19	1
878	3	17	19	1
879	3	18	19	1
880	3	19	19	1
881	3	0	18	1
882	3	1	18	1
883	3	2	18	0
884	3	3	18	0
885	3	4	18	0
886	3	5	18	0
887	3	6	18	0
888	3	7	18	0
889	3	8	18	0
890	3	9	18	0
891	3	10	18	1
892	3	11	18	0
893	3	12	18	0
894	3	13	18	0
895	3	14	18	0
896	3	15	18	0
897	3	16	18	0
898	3	17	18	0
899	3	18	18	1
900	3	19	18	1
901	3	0	17	1
902	3	1	17	1
903	3	2	17	0
904	3	3	17	1
905	3	4	17	1
906	3	5	17	1
907	3	6	17	1
908	3	7	17	1
909	3	8	17	1
910	3	9	17	0

911	3	10	17	1
912	3	11	17	1
913	3	12	17	1
914	3	13	17	1
915	3	14	17	1
916	3	15	17	1
917	3	16	17	1
918	3	17	17	0
919	3	18	17	1
920	3	19	17	1
921	3	0	16	1
922	3	1	16	1
923	3	2	16	0
924	3	3	16	1
925	3	4	16	1
926	3	5	16	1
927	3	6	16	1
928	3	7	16	1
929	3	8	16	1
930	3	9	16	0
931	3	10	16	1
932	3	11	16	1
933	3	12	16	0
934	3	13	16	1
935	3	14	16	0
936	3	15	16	0
937	3	16	16	0
938	3	17	16	0
939	3	18	16	1
940	3	19	16	1
941	3	0	15	1
942	3	1	15	1
943	3	2	15	0
944	3	3	15	1
945	3	4	15	1
946	3	5	15	1
947	3	6	15	1
948	3	7	15	1
949	3	8	15	1
950	3	9	15	0

951	3	10	15	1
952	3	11	15	1
953	3	12	15	0
954	3	13	15	1
955	3	14	15	1
956	3	15	15	1
957	3	16	15	1
958	3	17	15	0
959	3	18	15	1
960	3	19	15	1
961	3	0	14	1
962	3	1	14	1
963	3	2	14	0
964	3	3	14	1
965	3	4	14	1
966	3	5	14	1
967	3	6	14	1
968	3	7	14	1
969	3	8	14	1
970	3	9	14	0
971	3	10	14	1
972	3	11	14	1
973	3	12	14	0
974	3	13	14	1
975	3	14	14	1
976	3	15	14	1
977	3	16	14	1
978	3	17	14	0
979	3	18	14	1
980	3	19	14	1
981	3	0	13	1
982	3	1	13	1
983	3	2	13	0
984	3	3	13	1
985	3	4	13	1
986	3	5	13	1
987	3	6	13	1
988	3	7	13	1
989	3	8	13	1
990	3	9	13	0

991	3	10	13	1
992	3	11	13	1
993	3	12	13	0
994	3	13	13	0
995	3	14	13	0
996	3	15	13	0
997	3	16	13	0
998	3	17	13	0
999	3	18	13	1
1000	3	19	13	1
1001	3	0	12	1
1002	3	1	12	1
1003	3	2	12	0
1004	3	3	12	1
1005	3	4	12	1
1006	3	5	12	1
1007	3	6	12	1
1008	3	7	12	1
1009	3	8	12	1
1010	3	9	12	0
1011	3	10	12	1
1012	3	11	12	1
1013	3	12	12	1
1014	3	13	12	1
1015	3	14	12	1
1016	3	15	12	1
1017	3	16	12	1
1018	3	17	12	0
1019	3	18	12	1
1020	3	19	12	1
1021	3	0	11	1
1022	3	1	11	1
1023	3	2	11	0
1024	3	3	11	1
1025	3	4	11	1
1026	3	5	11	1
1027	3	6	11	1
1028	3	7	11	1
1029	3	8	11	1
1030	3	9	11	1

1031	3	10	11	1
1032	3	11	11	1
1033	3	12	11	1
1034	3	13	11	1
1035	3	14	11	1
1036	3	15	11	1
1037	3	16	11	1
1038	3	17	11	0
1039	3	18	11	1
1040	3	19	11	1
1041	3	0	10	1
1042	3	1	10	1
1043	3	2	10	0
1044	3	3	10	0
1045	3	4	10	0
1046	3	5	10	0
1047	3	6	10	0
1048	3	7	10	0
1049	3	8	10	0
1050	3	9	10	0
1051	3	10	10	1
1052	3	11	10	1
1053	3	12	10	1
1054	3	13	10	1
1055	3	14	10	1
1056	3	15	10	1
1057	3	16	10	1
1058	3	17	10	0
1059	3	18	10	1
1060	3	19	10	1
1061	3	0	9	1
1062	3	1	9	1
1063	3	2	9	0
1064	3	3	9	1
1065	3	4	9	1
1066	3	5	9	1
1067	3	6	9	1
1068	3	7	9	1
1069	3	8	9	1
1070	3	9	9	1

1071	3	10	9	1
1072	3	11	9	1
1073	3	12	9	1
1074	3	13	9	1
1075	3	14	9	1
1076	3	15	9	1
1077	3	16	9	1
1078	3	17	9	0
1079	3	18	9	1
1080	3	19	9	1
1081	3	0	8	1
1082	3	1	8	1
1083	3	2	8	0
1084	3	3	8	1
1085	3	4	8	1
1086	3	5	8	1
1087	3	6	8	1
1088	3	7	8	1
1089	3	8	8	1
1090	3	9	8	1
1091	3	10	8	1
1092	3	11	8	1
1093	3	12	8	1
1094	3	13	8	1
1095	3	14	8	1
1096	3	15	8	1
1097	3	16	8	1
1098	3	17	8	0
1099	3	18	8	1
1100	3	19	8	1
1101	3	0	7	1
1102	3	1	7	1
1103	3	2	7	0
1104	3	3	7	1
1105	3	4	7	1
1106	3	5	7	1
1107	3	6	7	1
1108	3	7	7	1
1109	3	8	7	1
1110	3	9	7	0

1111	3	10	7	1
1112	3	11	7	1
1113	3	12	7	1
1114	3	13	7	1
1115	3	14	7	1
1116	3	15	7	1
1117	3	16	7	1
1118	3	17	7	0
1119	3	18	7	1
1120	3	19	7	1
1121	3	0	6	1
1122	3	1	6	1
1123	3	2	6	0
1124	3	3	6	1
1125	3	4	6	1
1126	3	5	6	1
1127	3	6	6	1
1128	3	7	6	1
1129	3	8	6	1
1130	3	9	6	0
1131	3	10	6	1
1132	3	11	6	1
1133	3	12	6	1
1134	3	13	6	1
1135	3	14	6	1
1136	3	15	6	1
1137	3	16	6	1
1138	3	17	6	0
1139	3	18	6	1
1140	3	19	6	1
1141	3	0	5	1
1142	3	1	5	1
1143	3	2	5	0
1144	3	3	5	1
1145	3	4	5	1
1146	3	5	5	1
1147	3	6	5	1
1148	3	7	5	1
1149	3	8	5	1
1150	3	9	5	0

1151	3	10	5	1
1152	3	11	5	1
1153	3	12	5	1
1154	3	13	5	1
1155	3	14	5	1
1156	3	15	5	1
1157	3	16	5	1
1158	3	17	5	0
1159	3	18	5	1
1160	3	19	5	1
1161	3	0	4	1
1162	3	1	4	1
1163	3	2	4	0
1164	3	3	4	1
1165	3	4	4	1
1166	3	5	4	1
1167	3	6	4	1
1168	3	7	4	1
1169	3	8	4	1
1170	3	9	4	0
1171	3	10	4	0
1172	3	11	4	0
1173	3	12	4	0
1174	3	13	4	0
1175	3	14	4	0
1176	3	15	4	0
1177	3	16	4	0
1178	3	17	4	0
1179	3	18	4	1
1180	3	19	4	1
1181	3	0	3	1
1182	3	1	3	1
1183	3	2	3	0
1184	3	3	3	1
1185	3	4	3	1
1186	3	5	3	1
1187	3	6	3	1
1188	3	7	3	1
1189	3	8	3	1
1190	3	9	3	1

1191	3	10	3	1
1192	3	11	3	1
1193	3	12	3	1
1194	3	13	3	1
1195	3	14	3	1
1196	3	15	3	1
1197	3	16	3	1
1198	3	17	3	1
1199	3	18	3	1
1200	3	19	3	1
1201	3	0	2	1
1202	3	1	2	1
1203	3	2	2	0
1204	3	3	2	0
1205	3	4	2	0
1206	3	5	2	0
1207	3	6	2	0
1208	3	7	2	0
1209	3	8	2	0
1210	3	9	2	0
1211	3	10	2	1
1212	3	11	2	1
1213	3	12	2	1
1214	3	13	2	1
1215	3	14	2	1
1216	3	15	2	1
1217	3	16	2	1
1218	3	17	2	1
1219	3	18	2	1
1220	3	19	2	1
1221	3	0	1	1
1222	3	1	1	1
1223	3	2	1	1
1224	3	3	1	1
1225	3	4	1	1
1226	3	5	1	1
1227	3	6	1	1
1228	3	7	1	1
1229	3	8	1	1
1230	3	9	1	1

1231	3	10	1	1
1232	3	11	1	1
1233	3	12	1	1
1234	3	13	1	1
1235	3	14	1	1
1236	3	15	1	1
1237	3	16	1	1
1238	3	17	1	1
1239	3	18	1	1
1240	3	19	1	1
1241	3	0	0	1
1242	3	1	0	1
1243	3	2	0	1
1244	3	3	0	1
1245	3	4	0	1
1246	3	5	0	1
1247	3	6	0	1
1248	3	7	0	1
1249	3	8	0	1
1250	3	9	0	1
1251	3	10	0	1
1252	3	11	0	1
1253	3	12	0	1
1254	3	13	0	1
1255	3	14	0	1
1256	3	15	0	1
1257	3	16	0	1
1258	3	17	0	1
1259	3	18	0	1
1260	3	19	0	1
1261	4	0	20	1
1262	4	1	20	1
1263	4	2	20	1
1264	4	3	20	1
1265	4	4	20	1
1266	4	5	20	1
1267	4	6	20	1
1268	4	7	20	1
1269	4	8	20	1
1270	4	9	20	1

1271	4	10	20	1
1272	4	11	20	1
1273	4	12	20	1
1274	4	13	20	1
1275	4	14	20	1
1276	4	15	20	1
1277	4	16	20	1
1278	4	17	20	1
1279	4	18	20	1
1280	4	19	20	1
1281	4	0	19	1
1282	4	1	19	1
1283	4	2	19	1
1284	4	3	19	1
1285	4	4	19	1
1286	4	5	19	1
1287	4	6	19	1
1288	4	7	19	1
1289	4	8	19	1
1290	4	9	19	1
1291	4	10	19	1
1292	4	11	19	1
1293	4	12	19	1
1294	4	13	19	1
1295	4	14	19	1
1296	4	15	19	1
1297	4	16	19	1
1298	4	17	19	1
1299	4	18	19	1
1300	4	19	19	1
1301	4	0	18	1
1302	4	1	18	1
1303	4	2	18	0
1304	4	3	18	0
1305	4	4	18	0
1306	4	5	18	0
1307	4	6	18	0
1308	4	7	18	0
1309	4	8	18	0
1310	4	9	18	0

1311	4	10	18	1
1312	4	11	18	0
1313	4	12	18	0
1314	4	13	18	0
1315	4	14	18	0
1316	4	15	18	0
1317	4	16	18	0
1318	4	17	18	0
1319	4	18	18	1
1320	4	19	18	1
1321	4	0	17	1
1322	4	1	17	1
1323	4	2	17	0
1324	4	3	17	1
1325	4	4	17	1
1326	4	5	17	1
1327	4	6	17	1
1328	4	7	17	1
1329	4	8	17	1
1330	4	9	17	0
1331	4	10	17	1
1332	4	11	17	1
1333	4	12	17	1
1334	4	13	17	1
1335	4	14	17	1
1336	4	15	17	1
1337	4	16	17	1
1338	4	17	17	0
1339	4	18	17	1
1340	4	19	17	1
1341	4	0	16	1
1342	4	1	16	1
1343	4	2	16	0
1344	4	3	16	1
1345	4	4	16	1
1346	4	5	16	1
1347	4	6	16	1
1348	4	7	16	1
1349	4	8	16	1
1350	4	9	16	0

1351	4	10	16	1
1352	4	11	16	1
1353	4	12	16	0
1354	4	13	16	1
1355	4	14	16	0
1356	4	15	16	0
1357	4	16	16	0
1358	4	17	16	0
1359	4	18	16	1
1360	4	19	16	1
1361	4	0	15	1
1362	4	1	15	1
1363	4	2	15	0
1364	4	3	15	1
1365	4	4	15	1
1366	4	5	15	1
1367	4	6	15	1
1368	4	7	15	1
1369	4	8	15	1
1370	4	9	15	0
1371	4	10	15	1
1372	4	11	15	1
1373	4	12	15	0
1374	4	13	15	1
1375	4	14	15	1
1376	4	15	15	1
1377	4	16	15	1
1378	4	17	15	0
1379	4	18	15	1
1380	4	19	15	1
1381	4	0	14	1
1382	4	1	14	1
1383	4	2	14	0
1384	4	3	14	1
1385	4	4	14	1
1386	4	5	14	1
1387	4	6	14	1
1388	4	7	14	1
1389	4	8	14	1
1390	4	9	14	0

1391	4	10	14	1
1392	4	11	14	1
1393	4	12	14	0
1394	4	13	14	1
1395	4	14	14	1
1396	4	15	14	1
1397	4	16	14	1
1398	4	17	14	0
1399	4	18	14	1
1400	4	19	14	1
1401	4	0	13	1
1402	4	1	13	1
1403	4	2	13	0
1404	4	3	13	1
1405	4	4	13	1
1406	4	5	13	1
1407	4	6	13	1
1408	4	7	13	1
1409	4	8	13	1
1410	4	9	13	0
1411	4	10	13	1
1412	4	11	13	1
1413	4	12	13	0
1414	4	13	13	0
1415	4	14	13	0
1416	4	15	13	0
1417	4	16	13	0
1418	4	17	13	0
1419	4	18	13	1
1420	4	19	13	1
1421	4	0	12	1
1422	4	1	12	1
1423	4	2	12	0
1424	4	3	12	1
1425	4	4	12	1
1426	4	5	12	1
1427	4	6	12	1
1428	4	7	12	1
1429	4	8	12	1
1430	4	9	12	0

1431	4	10	12	1
1432	4	11	12	1
1433	4	12	12	1
1434	4	13	12	1
1435	4	14	12	1
1436	4	15	12	1
1437	4	16	12	1
1438	4	17	12	0
1439	4	18	12	1
1440	4	19	12	1
1441	4	0	11	1
1442	4	1	11	1
1443	4	2	11	0
1444	4	3	11	1
1445	4	4	11	1
1446	4	5	11	1
1447	4	6	11	1
1448	4	7	11	1
1449	4	8	11	1
1450	4	9	11	1
1451	4	10	11	1
1452	4	11	11	1
1453	4	12	11	1
1454	4	13	11	1
1455	4	14	11	1
1456	4	15	11	1
1457	4	16	11	1
1458	4	17	11	0
1459	4	18	11	1
1460	4	19	11	1
1461	4	0	10	1
1462	4	1	10	1
1463	4	2	10	0
1464	4	3	10	0
1465	4	4	10	0
1466	4	5	10	0
1467	4	6	10	0
1468	4	7	10	0
1469	4	8	10	0
1470	4	9	10	0

1471	4	10	10	1
1472	4	11	10	1
1473	4	12	10	1
1474	4	13	10	1
1475	4	14	10	1
1476	4	15	10	1
1477	4	16	10	1
1478	4	17	10	0
1479	4	18	10	1
1480	4	19	10	1
1481	4	0	9	1
1482	4	1	9	1
1483	4	2	9	0
1484	4	3	9	1
1485	4	4	9	1
1486	4	5	9	1
1487	4	6	9	1
1488	4	7	9	1
1489	4	8	9	1
1490	4	9	9	1
1491	4	10	9	1
1492	4	11	9	1
1493	4	12	9	1
1494	4	13	9	1
1495	4	14	9	1
1496	4	15	9	1
1497	4	16	9	1
1498	4	17	9	0
1499	4	18	9	1
1500	4	19	9	1
1501	4	0	8	1
1502	4	1	8	1
1503	4	2	8	0
1504	4	3	8	1
1505	4	4	8	1
1506	4	5	8	1
1507	4	6	8	1
1508	4	7	8	1
1509	4	8	8	1
1510	4	9	8	1

1511	4	10	8	1
1512	4	11	8	1
1513	4	12	8	1
1514	4	13	8	1
1515	4	14	8	1
1516	4	15	8	1
1517	4	16	8	1
1518	4	17	8	0
1519	4	18	8	1
1520	4	19	8	1
1521	4	0	7	1
1522	4	1	7	1
1523	4	2	7	0
1524	4	3	7	1
1525	4	4	7	1
1526	4	5	7	1
1527	4	6	7	1
1528	4	7	7	1
1529	4	8	7	1
1530	4	9	7	0
1531	4	10	7	1
1532	4	11	7	1
1533	4	12	7	1
1534	4	13	7	1
1535	4	14	7	1
1536	4	15	7	1
1537	4	16	7	1
1538	4	17	7	0
1539	4	18	7	1
1540	4	19	7	1
1541	4	0	6	1
1542	4	1	6	1
1543	4	2	6	0
1544	4	3	6	1
1545	4	4	6	1
1546	4	5	6	1
1547	4	6	6	1
1548	4	7	6	1
1549	4	8	6	1
1550	4	9	6	0

1551	4	10	6	1
1552	4	11	6	1
1553	4	12	6	1
1554	4	13	6	1
1555	4	14	6	1
1556	4	15	6	1
1557	4	16	6	1
1558	4	17	6	0
1559	4	18	6	1
1560	4	19	6	1
1561	4	0	5	1
1562	4	1	5	1
1563	4	2	5	0
1564	4	3	5	1
1565	4	4	5	1
1566	4	5	5	1
1567	4	6	5	1
1568	4	7	5	1
1569	4	8	5	1
1570	4	9	5	0
1571	4	10	5	1
1572	4	11	5	1
1573	4	12	5	1
1574	4	13	5	1
1575	4	14	5	1
1576	4	15	5	1
1577	4	16	5	1
1578	4	17	5	0
1579	4	18	5	1
1580	4	19	5	1
1581	4	0	4	1
1582	4	1	4	1
1583	4	2	4	0
1584	4	3	4	1
1585	4	4	4	1
1586	4	5	4	1
1587	4	6	4	1
1588	4	7	4	1
1589	4	8	4	1
1590	4	9	4	0

1591	4	10	4	0
1592	4	11	4	0
1593	4	12	4	0
1594	4	13	4	0
1595	4	14	4	0
1596	4	15	4	0
1597	4	16	4	0
1598	4	17	4	0
1599	4	18	4	1
1600	4	19	4	1
1601	4	0	3	1
1602	4	1	3	1
1603	4	2	3	0
1604	4	3	3	1
1605	4	4	3	1
1606	4	5	3	1
1607	4	6	3	1
1608	4	7	3	1
1609	4	8	3	1
1610	4	9	3	1
1611	4	10	3	1
1612	4	11	3	1
1613	4	12	3	1
1614	4	13	3	1
1615	4	14	3	1
1616	4	15	3	1
1617	4	16	3	1
1618	4	17	3	1
1619	4	18	3	1
1620	4	19	3	1
1621	4	0	2	1
1622	4	1	2	1
1623	4	2	2	0
1624	4	3	2	0
1625	4	4	2	0
1626	4	5	2	0
1627	4	6	2	0
1628	4	7	2	0
1629	4	8	2	0
1630	4	9	2	0

1631	4	10	2	1
1632	4	11	2	1
1633	4	12	2	1
1634	4	13	2	1
1635	4	14	2	1
1636	4	15	2	1
1637	4	16	2	1
1638	4	17	2	1
1639	4	18	2	1
1640	4	19	2	1
1641	4	0	1	1
1642	4	1	1	1
1643	4	2	1	1
1644	4	3	1	1
1645	4	4	1	1
1646	4	5	1	1
1647	4	6	1	1
1648	4	7	1	1
1649	4	8	1	1
1650	4	9	1	1
1651	4	10	1	1
1652	4	11	1	1
1653	4	12	1	1
1654	4	13	1	1
1655	4	14	1	1
1656	4	15	1	1
1657	4	16	1	1
1658	4	17	1	1
1659	4	18	1	1
1660	4	19	1	1
1661	4	0	0	1
1662	4	1	0	1
1663	4	2	0	1
1664	4	3	0	1
1665	4	4	0	1
1666	4	5	0	1
1667	4	6	0	1
1668	4	7	0	1
1669	4	8	0	1
1670	4	9	0	1

1671	4	10	0	1
1672	4	11	0	1
1673	4	12	0	1
1674	4	13	0	1
1675	4	14	0	1
1676	4	15	0	1
1677	4	16	0	1
1678	4	17	0	1
1679	4	18	0	1
1680	4	19	0	1
1681	5	0	20	1
1682	5	1	20	1
1683	5	2	20	1
1684	5	3	20	1
1685	5	4	20	1
1686	5	5	20	1
1687	5	6	20	1
1688	5	7	20	1
1689	5	8	20	1
1690	5	9	20	1
1691	5	10	20	1
1692	5	11	20	1
1693	5	12	20	1
1694	5	13	20	1
1695	5	14	20	1
1696	5	15	20	1
1697	5	16	20	1
1698	5	17	20	1
1699	5	18	20	1
1700	5	19	20	1
1701	5	0	19	1
1702	5	1	19	1
1703	5	2	19	1
1704	5	3	19	1
1705	5	4	19	1
1706	5	5	19	1
1707	5	6	19	1
1708	5	7	19	1
1709	5	8	19	1
1710	5	9	19	1

1711	5	10	19	1
1712	5	11	19	1
1713	5	12	19	1
1714	5	13	19	1
1715	5	14	19	1
1716	5	15	19	1
1717	5	16	19	1
1718	5	17	19	1
1719	5	18	19	1
1720	5	19	19	1
1721	5	0	18	1
1722	5	1	18	1
1723	5	2	18	0
1724	5	3	18	0
1725	5	4	18	0
1726	5	5	18	0
1727	5	6	18	0
1728	5	7	18	0
1729	5	8	18	0
1730	5	9	18	0
1731	5	10	18	1
1732	5	11	18	0
1733	5	12	18	0
1734	5	13	18	0
1735	5	14	18	0
1736	5	15	18	0
1737	5	16	18	0
1738	5	17	18	0
1739	5	18	18	1
1740	5	19	18	1
1741	5	0	17	1
1742	5	1	17	1
1743	5	2	17	0
1744	5	3	17	1
1745	5	4	17	1
1746	5	5	17	1
1747	5	6	17	1
1748	5	7	17	1
1749	5	8	17	1
1750	5	9	17	0

1751	5	10	17	1
1752	5	11	17	1
1753	5	12	17	1
1754	5	13	17	1
1755	5	14	17	1
1756	5	15	17	1
1757	5	16	17	1
1758	5	17	17	0
1759	5	18	17	1
1760	5	19	17	1
1761	5	0	16	1
1762	5	1	16	1
1763	5	2	16	0
1764	5	3	16	1
1765	5	4	16	1
1766	5	5	16	1
1767	5	6	16	1
1768	5	7	16	1
1769	5	8	16	1
1770	5	9	16	0
1771	5	10	16	1
1772	5	11	16	1
1773	5	12	16	0
1774	5	13	16	1
1775	5	14	16	0
1776	5	15	16	0
1777	5	16	16	0
1778	5	17	16	0
1779	5	18	16	1
1780	5	19	16	1
1781	5	0	15	1
1782	5	1	15	1
1783	5	2	15	0
1784	5	3	15	1
1785	5	4	15	1
1786	5	5	15	1
1787	5	6	15	1
1788	5	7	15	1
1789	5	8	15	1
1790	5	9	15	0

1791	5	10	15	1
1792	5	11	15	1
1793	5	12	15	0
1794	5	13	15	1
1795	5	14	15	1
1796	5	15	15	1
1797	5	16	15	1
1798	5	17	15	0
1799	5	18	15	1
1800	5	19	15	1
1801	5	0	14	1
1802	5	1	14	1
1803	5	2	14	0
1804	5	3	14	1
1805	5	4	14	1
1806	5	5	14	1
1807	5	6	14	1
1808	5	7	14	1
1809	5	8	14	1
1810	5	9	14	0
1811	5	10	14	1
1812	5	11	14	1
1813	5	12	14	0
1814	5	13	14	1
1815	5	14	14	1
1816	5	15	14	1
1817	5	16	14	1
1818	5	17	14	0
1819	5	18	14	1
1820	5	19	14	1
1821	5	0	13	1
1822	5	1	13	1
1823	5	2	13	0
1824	5	3	13	1
1825	5	4	13	1
1826	5	5	13	1
1827	5	6	13	1
1828	5	7	13	1
1829	5	8	13	1
1830	5	9	13	0

1831	5	10	13	1
1832	5	11	13	1
1833	5	12	13	0
1834	5	13	13	0
1835	5	14	13	0
1836	5	15	13	0
1837	5	16	13	0
1838	5	17	13	0
1839	5	18	13	1
1840	5	19	13	1
1841	5	0	12	1
1842	5	1	12	1
1843	5	2	12	0
1844	5	3	12	1
1845	5	4	12	1
1846	5	5	12	1
1847	5	6	12	1
1848	5	7	12	1
1849	5	8	12	1
1850	5	9	12	0
1851	5	10	12	1
1852	5	11	12	1
1853	5	12	12	1
1854	5	13	12	1
1855	5	14	12	1
1856	5	15	12	1
1857	5	16	12	1
1858	5	17	12	0
1859	5	18	12	1
1860	5	19	12	1
1861	5	0	11	1
1862	5	1	11	1
1863	5	2	11	0
1864	5	3	11	1
1865	5	4	11	1
1866	5	5	11	1
1867	5	6	11	1
1868	5	7	11	1
1869	5	8	11	1
1870	5	9	11	1

1871	5	10	11	1
1872	5	11	11	1
1873	5	12	11	1
1874	5	13	11	1
1875	5	14	11	1
1876	5	15	11	1
1877	5	16	11	1
1878	5	17	11	0
1879	5	18	11	1
1880	5	19	11	1
1881	5	0	10	1
1882	5	1	10	1
1883	5	2	10	0
1884	5	3	10	0
1885	5	4	10	0
1886	5	5	10	0
1887	5	6	10	0
1888	5	7	10	0
1889	5	8	10	0
1890	5	9	10	0
1891	5	10	10	1
1892	5	11	10	1
1893	5	12	10	1
1894	5	13	10	1
1895	5	14	10	1
1896	5	15	10	1
1897	5	16	10	1
1898	5	17	10	0
1899	5	18	10	1
1900	5	19	10	1
1901	5	0	9	1
1902	5	1	9	1
1903	5	2	9	0
1904	5	3	9	1
1905	5	4	9	1
1906	5	5	9	1
1907	5	6	9	1
1908	5	7	9	1
1909	5	8	9	1
1910	5	9	9	1

1911	5	10	9	1
1912	5	11	9	1
1913	5	12	9	1
1914	5	13	9	1
1915	5	14	9	1
1916	5	15	9	1
1917	5	16	9	1
1918	5	17	9	0
1919	5	18	9	1
1920	5	19	9	1
1921	5	0	8	1
1922	5	1	8	1
1923	5	2	8	0
1924	5	3	8	1
1925	5	4	8	1
1926	5	5	8	1
1927	5	6	8	1
1928	5	7	8	1
1929	5	8	8	1
1930	5	9	8	1
1931	5	10	8	1
1932	5	11	8	1
1933	5	12	8	1
1934	5	13	8	1
1935	5	14	8	1
1936	5	15	8	1
1937	5	16	8	1
1938	5	17	8	0
1939	5	18	8	1
1940	5	19	8	1
1941	5	0	7	1
1942	5	1	7	1
1943	5	2	7	0
1944	5	3	7	1
1945	5	4	7	1
1946	5	5	7	1
1947	5	6	7	1
1948	5	7	7	1
1949	5	8	7	1
1950	5	9	7	0

1951	5	10	7	1
1952	5	11	7	1
1953	5	12	7	1
1954	5	13	7	1
1955	5	14	7	1
1956	5	15	7	1
1957	5	16	7	1
1958	5	17	7	0
1959	5	18	7	1
1960	5	19	7	1
1961	5	0	6	1
1962	5	1	6	1
1963	5	2	6	0
1964	5	3	6	1
1965	5	4	6	1
1966	5	5	6	1
1967	5	6	6	1
1968	5	7	6	1
1969	5	8	6	1
1970	5	9	6	0
1971	5	10	6	1
1972	5	11	6	1
1973	5	12	6	1
1974	5	13	6	1
1975	5	14	6	1
1976	5	15	6	1
1977	5	16	6	1
1978	5	17	6	0
1979	5	18	6	1
1980	5	19	6	1
1981	5	0	5	1
1982	5	1	5	1
1983	5	2	5	0
1984	5	3	5	1
1985	5	4	5	1
1986	5	5	5	1
1987	5	6	5	1
1988	5	7	5	1
1989	5	8	5	1
1990	5	9	5	0

1991	5	10	5	1
1992	5	11	5	1
1993	5	12	5	1
1994	5	13	5	1
1995	5	14	5	1
1996	5	15	5	1
1997	5	16	5	1
1998	5	17	5	0
1999	5	18	5	1
2000	5	19	5	1
2001	5	0	4	1
2002	5	1	4	1
2003	5	2	4	0
2004	5	3	4	1
2005	5	4	4	1
2006	5	5	4	1
2007	5	6	4	1
2008	5	7	4	1
2009	5	8	4	1
2010	5	9	4	0
2011	5	10	4	0
2012	5	11	4	0
2013	5	12	4	0
2014	5	13	4	0
2015	5	14	4	0
2016	5	15	4	0
2017	5	16	4	0
2018	5	17	4	0
2019	5	18	4	1
2020	5	19	4	1
2021	5	0	3	1
2022	5	1	3	1
2023	5	2	3	0
2024	5	3	3	1
2025	5	4	3	1
2026	5	5	3	1
2027	5	6	3	1
2028	5	7	3	1
2029	5	8	3	1
2030	5	9	3	1

2031	5	10	3	1
2032	5	11	3	1
2033	5	12	3	1
2034	5	13	3	1
2035	5	14	3	1
2036	5	15	3	1
2037	5	16	3	1
2038	5	17	3	1
2039	5	18	3	1
2040	5	19	3	1
2041	5	0	2	1
2042	5	1	2	1
2043	5	2	2	0
2044	5	3	2	0
2045	5	4	2	0
2046	5	5	2	0
2047	5	6	2	0
2048	5	7	2	0
2049	5	8	2	0
2050	5	9	2	0
2051	5	10	2	1
2052	5	11	2	1
2053	5	12	2	1
2054	5	13	2	1
2055	5	14	2	1
2056	5	15	2	1
2057	5	16	2	1
2058	5	17	2	1
2059	5	18	2	1
2060	5	19	2	1
2061	5	0	1	1
2062	5	1	1	1
2063	5	2	1	1
2064	5	3	1	1
2065	5	4	1	1
2066	5	5	1	1
2067	5	6	1	1
2068	5	7	1	1
2069	5	8	1	1
2070	5	9	1	1

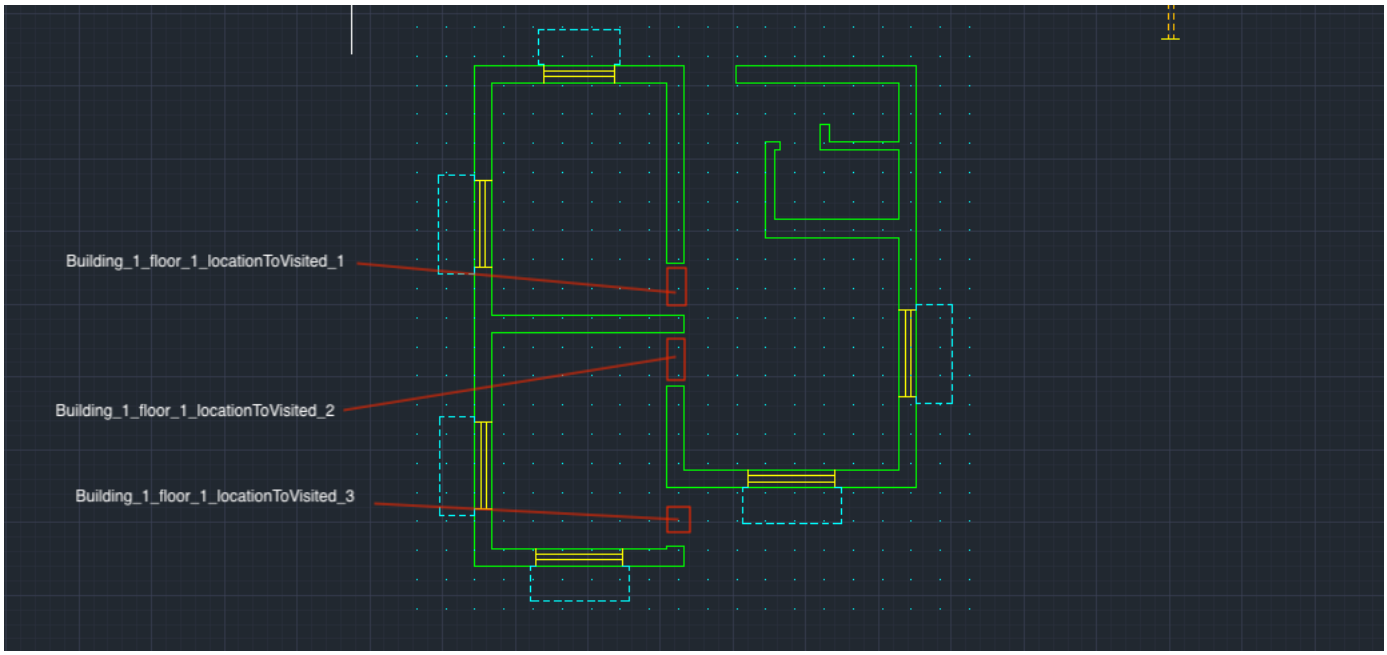
2071	5	10	1	1
2072	5	11	1	1
2073	5	12	1	1
2074	5	13	1	1
2075	5	14	1	1
2076	5	15	1	1
2077	5	16	1	1
2078	5	17	1	1
2079	5	18	1	1
2080	5	19	1	1
2081	5	0	0	1
2082	5	1	0	1
2083	5	2	0	1
2084	5	3	0	1
2085	5	4	0	1
2086	5	5	0	1
2087	5	6	0	1
2088	5	7	0	1
2089	5	8	0	1
2090	5	9	0	1
2091	5	10	0	1
2092	5	11	0	1
2093	5	12	0	1
2094	5	13	0	1
2095	5	14	0	1
2096	5	15	0	1
2097	5	16	0	1
2098	5	17	0	1
2099	5	18	0	1
2100	5	19	0	1

locationToVisit - table test

The location in building 1 floor 1 is as follow

Coordinate of location 1 and location 2:

- location 1: (9,11)
- location 2: (9,9)
- location 3: (9,3)




```
floor = 1
x = 9
y= 11
sql = 'select * from'\
' (select * from floorMapPoints fmp'\
' where fmp.idfloorMap = ' + str(floor) + ') fp'\
' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)

print(sql)
df = pd.read_sql(sql,connection)['coordinatePoint'][0]
df
```

```
select * from (select * from floorMapPoints fmp where fmp.idfloorMap = 1) fp where fp.x = 9 and fp.y
= 11




190
```

```
# Generate locations for building 1 floor 1
def generate_location_to_visited(floor=[], locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = location_coordinate[1]
            location_no = len(locations_coordinate) if id_location % len(locations_coordinate)==0
else id_location % len(locations_coordinate)

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]


result.append([id_location,map_point_id,'floor_'+str(floor)+'_location_'+str(location_no),'floor_'+s
```

```
tr(floor)+'_location_'+str(location_no)+'_description'])
            id_location+=1

    return result
```

```
location_to_visited_data = generate_location_to_visited(floor=[1,2,3,4,5],locations_coordinate=
[(9,11),(9,9),(9,3)])
```

```
location_to_visited_columns = ['idlocationToVisit','coordinatePoint','name','description']
location_to_visited =
pd.DataFrame(data=location_to_visited_data,columns=location_to_visited_columns)
```

```
pd.io.sql.to_sql(location_to_visited,'locationtovisit',connection,schema='mydb',if_exists='append',i
ndex=False)
```

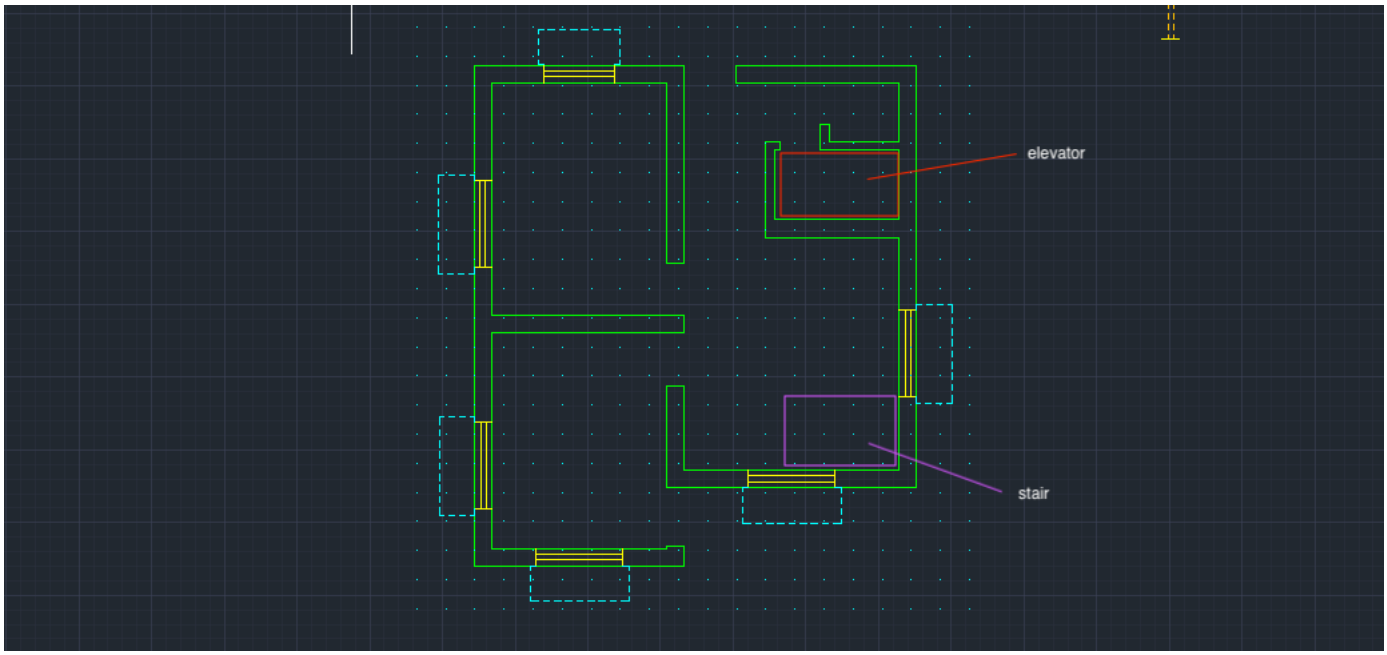
```
%%sql
select * from locationToVisited
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
15 rows affected.
```

idlocationToVisiti	coordinatePoint	name	description
1	190	floor_1_location_1	floor_1_location_1_description
2	230	floor_1_location_2	floor_1_location_2_description
3	350	floor_1_location_3	floor_1_location_3_description
4	610	floor_2_location_1	floor_2_location_1_description
5	650	floor_2_location_2	floor_2_location_2_description
6	770	floor_2_location_3	floor_2_location_3_description
7	1030	floor_3_location_1	floor_3_location_1_description
8	1070	floor_3_location_2	floor_3_location_2_description
9	1190	floor_3_location_3	floor_3_location_3_description
10	1450	floor_4_location_1	floor_4_location_1_description
11	1490	floor_4_location_2	floor_4_location_2_description
12	1610	floor_4_location_3	floor_4_location_3_description
13	1870	floor_5_location_1	floor_5_location_1_description
14	1910	floor_5_location_2	floor_5_location_2_description
15	2030	floor_5_location_3	floor_5_location_3_description

locationToChangeFloor - table test

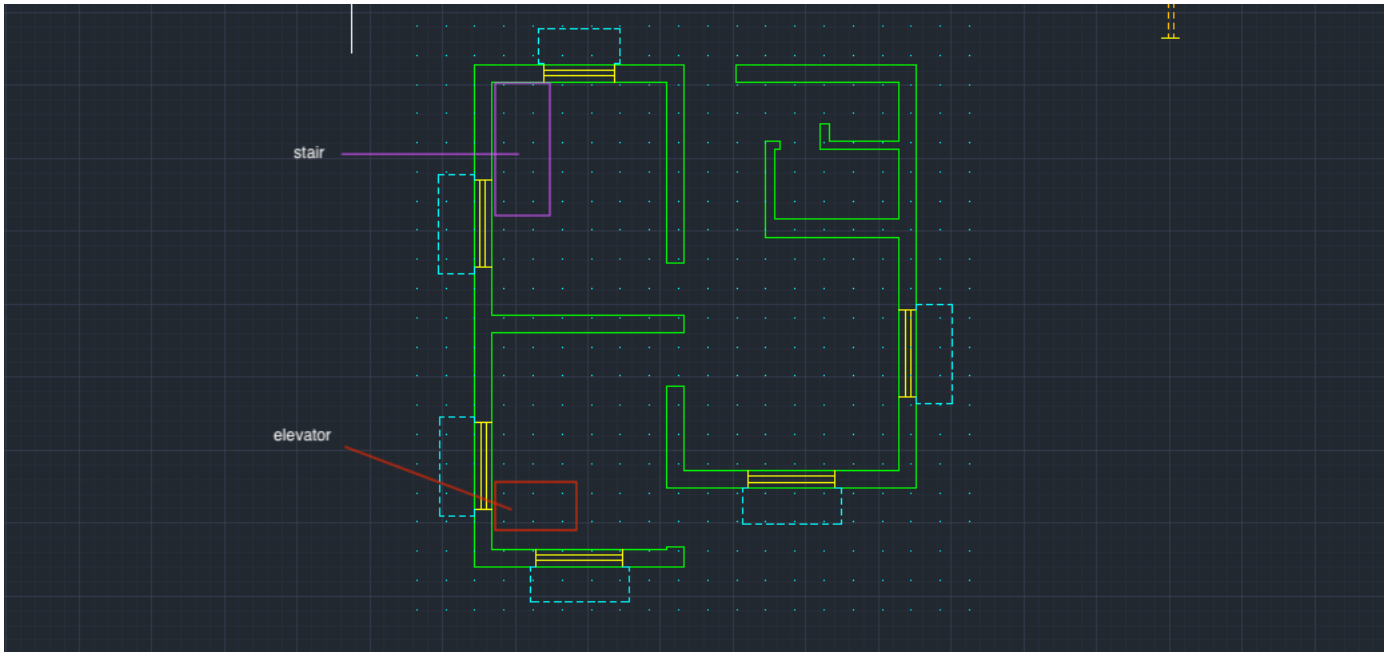
Building 1 floor 1 2 3



In building 1:

- elevator: (13,16)
- start: (13,6)

Building 2 floor 1 2



In building 2:

- elevator: (4,4)
- stair: (4,16)

```
# Generate locationToChangeFloor for building 1 floor 1
def generate_location_to_change_floor(floor=[], name=[], locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = location_coordinate[1]
            location_no = len(locations_coordinate) if id_location % len(locations_coordinate)==0
else id_location % len(locations_coordinate)
```

```
            location_type = name[location_no-1]
            is_friendly = True if name[location_no-1] == 'elevator' else False
            timeCost = 5 if is_friendly else 10

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]

            result.append([id_location,
                           map_point_id,

'floor_'+str(floor)+'_location_to_change_floor_'+str(location_no)+'_'+name[location_no-1],
                           location_type,
                           is_friendly,
                           timeCost])
            id_location+=1

    return result
```

```
location_to_change_floor_data = generate_location_to_change_floor(floor=[1,2,3],name=
['elevator','stair'], locations_coordinate=[(13,16),(13,6)])
```

```
location_to_change_floor_columns =
['idlocationToChangeFloor','coordinatePoint','name','type','isdisabilityFriendly','distance']
location_to_change_floor =
pd.DataFrame(data=location_to_change_floor_data,columns=location_to_change_floor_columns)
location_to_change_floor
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idlocationToChangeFloor	coordinatePoint	name	type	isdisabilityFriendly	distance
0	1	94	floor_1_location_to_change_floor_1_elevator	elevator	True	5
1	2	294	floor_1_location_to_change_floor_2_stair	stair	False	10
2	3	514	floor_2_location_to_change_floor_1_elevator	elevator	True	5
3	4	714	floor_2_location_to_change_floor_2_stair	stair	False	10
4	5	934	floor_3_location_to_change_floor_1_elevator	elevator	True	5
5	6	1134	floor_3_location_to_change_floor_2_stair	stair	False	10

```
pd.io.sql.to_sql(location_to_change_floor,'locationToChangeFloor',connection,schema='mydb',if_exists
='append',index=False)
```

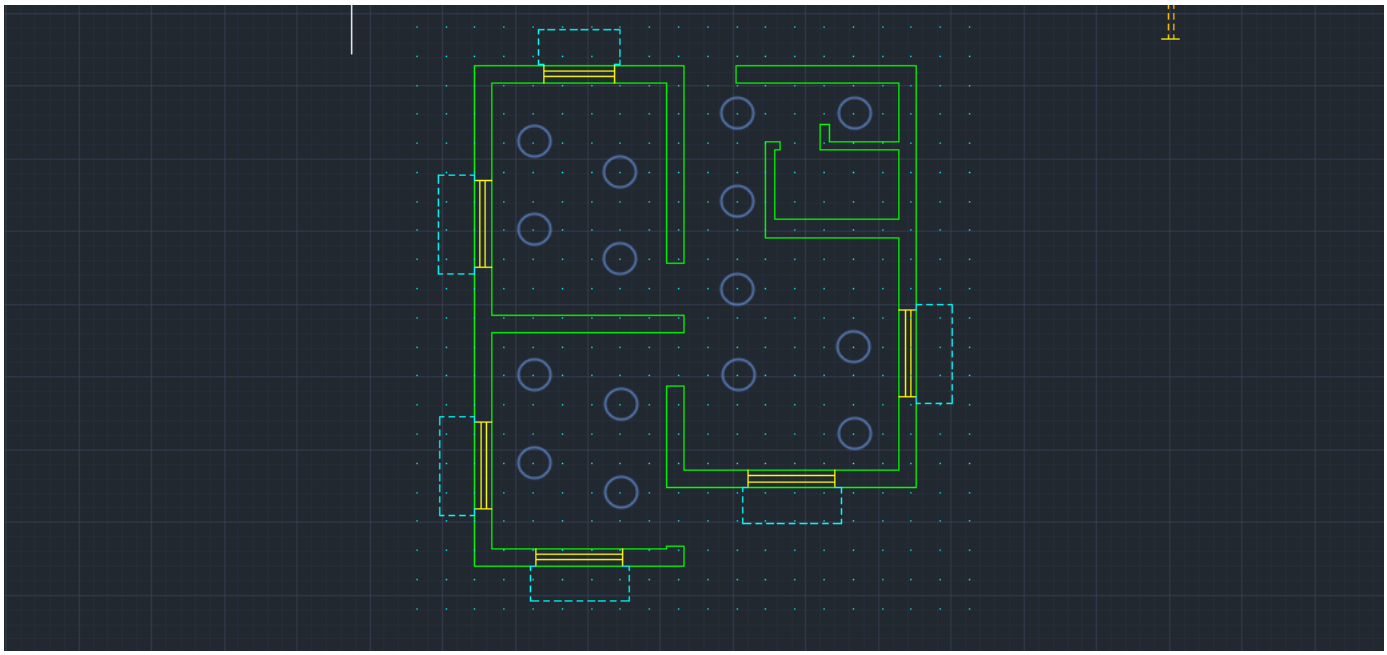
```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pandas/io/sql.py:1336: UserWarning: The
provided table name 'locationToChangeFloor' is not found exactly as such in the database after
writing the table, possibly due to case sensitivity issues. Consider using lower case table names.
  warnings.warn(msg, UserWarning)
```

```
%%sql
select * from locationToChangeFloor
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
6 rows affected.
```

idlocationToChangeFloor	coordinatePoint	name	type	isdisabilityFriendly	distance
1	94	floor_1_location_to_change_floor_1_elevator	elevator	1	5
2	294	floor_1_location_to_change_floor_2_stair	stair	0	10
3	514	floor_2_location_to_change_floor_1_elevator	elevator	1	5
4	714	floor_2_location_to_change_floor_2_stair	stair	0	10
5	934	floor_3_location_to_change_floor_1_elevator	elevator	1	5
6	1134	floor_3_location_to_change_floor_2_stair	stair	0	10

beacons - table test



These are beacons positions inside floors

I assume all floors use the same pattern

The position:

- (4,16)
- (7,15)
- (4,13)
- (7,12)
- (11,17)
- (15,17)
- (11,14)
- (11,11)
- (11,8)
- (15,9)
- (15,6)

```
# Generate beacon for all floor
def generate_beacon_location(floor=[], beacons_locations_coordinate=[]):
    result = []
    id_location = 1
    for floor_id in floor:
        for location_coordinate in beacons_locations_coordinate:
            floor = floor_id
            x = location_coordinate[0]
            y = 20 - location_coordinate[1]
            location_no = len(beacons_locations_coordinate) if id_location %
len(beacons_locations_coordinate)==0 else id_location % len(beacons_locations_coordinate)

            sql = 'select * from'\
            ' (select * from floorMapPoints fmp'\
            ' where fmp.idfloorMap = ' + str(floor) + ') fp'\
            ' where fp.x = ' + str(x) + ' and fp.y = '+ str(y)
            map_point_id = pd.read_sql(sql,connection)['coordinatePoint'][0]

            result.append([id_location,
                           map_point_id,
                           'floor_'+str(floor)+'_beacon_model_'+str(location_no),
                           'floor_'+str(floor)+'_beacon_'+str(location_no)+'_description',
                           True])
            id_location+=1

    return result
```

```
beacon_location_data = generate_beacon_location(floor=[1,2,3,4,5],beacons_locations_coordinate=
[(4,4),(7,5),(4,7),(7,8),(11,3),(15,3),(11,6),(11,9),(11,12),(15,11),(15,14)])
```

```
beacon_location_columns = ['idbeacon','coordinatePoint','beaconModel','description','ifwork']
beacon_location = pd.DataFrame(data=beacon_location_data,columns=beacon_location_columns)
```

```
pd.io.sql.to_sql(beacon_location,'beacons',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from beacons
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
55 rows affected.
```

idbeacon	coordinatePoint	beaconModel	description	ifwork
1	85	floor_1_beacon_model_1	floor_1_beacon_1_description	1
2	108	floor_1_beacon_model_2	floor_1_beacon_2_description	1
3	145	floor_1_beacon_model_3	floor_1_beacon_3_description	1
4	168	floor_1_beacon_model_4	floor_1_beacon_4_description	1
5	72	floor_1_beacon_model_5	floor_1_beacon_5_description	1
6	76	floor_1_beacon_model_6	floor_1_beacon_6_description	1
7	132	floor_1_beacon_model_7	floor_1_beacon_7_description	1
8	192	floor_1_beacon_model_8	floor_1_beacon_8_description	1

9	252	floor_1_beacon_model_9	floor_1_beacon_9_description	1
10	236	floor_1_beacon_model_10	floor_1_beacon_10_description	1
11	296	floor_1_beacon_model_11	floor_1_beacon_11_description	1
12	505	floor_2_beacon_model_1	floor_2_beacon_1_description	1
13	528	floor_2_beacon_model_2	floor_2_beacon_2_description	1
14	565	floor_2_beacon_model_3	floor_2_beacon_3_description	1
15	588	floor_2_beacon_model_4	floor_2_beacon_4_description	1
16	492	floor_2_beacon_model_5	floor_2_beacon_5_description	1
17	496	floor_2_beacon_model_6	floor_2_beacon_6_description	1
18	552	floor_2_beacon_model_7	floor_2_beacon_7_description	1
19	612	floor_2_beacon_model_8	floor_2_beacon_8_description	1
20	672	floor_2_beacon_model_9	floor_2_beacon_9_description	1
21	656	floor_2_beacon_model_10	floor_2_beacon_10_description	1
22	716	floor_2_beacon_model_11	floor_2_beacon_11_description	1
23	925	floor_3_beacon_model_1	floor_3_beacon_1_description	1
24	948	floor_3_beacon_model_2	floor_3_beacon_2_description	1
25	985	floor_3_beacon_model_3	floor_3_beacon_3_description	1
26	1008	floor_3_beacon_model_4	floor_3_beacon_4_description	1
27	912	floor_3_beacon_model_5	floor_3_beacon_5_description	1
28	916	floor_3_beacon_model_6	floor_3_beacon_6_description	1
29	972	floor_3_beacon_model_7	floor_3_beacon_7_description	1
30	1032	floor_3_beacon_model_8	floor_3_beacon_8_description	1
31	1092	floor_3_beacon_model_9	floor_3_beacon_9_description	1
32	1076	floor_3_beacon_model_10	floor_3_beacon_10_description	1
33	1136	floor_3_beacon_model_11	floor_3_beacon_11_description	1
34	1345	floor_4_beacon_model_1	floor_4_beacon_1_description	1
35	1368	floor_4_beacon_model_2	floor_4_beacon_2_description	1
36	1405	floor_4_beacon_model_3	floor_4_beacon_3_description	1
37	1428	floor_4_beacon_model_4	floor_4_beacon_4_description	1
38	1332	floor_4_beacon_model_5	floor_4_beacon_5_description	1
39	1336	floor_4_beacon_model_6	floor_4_beacon_6_description	1
40	1392	floor_4_beacon_model_7	floor_4_beacon_7_description	1
41	1452	floor_4_beacon_model_8	floor_4_beacon_8_description	1
42	1512	floor_4_beacon_model_9	floor_4_beacon_9_description	1
43	1496	floor_4_beacon_model_10	floor_4_beacon_10_description	1
44	1556	floor_4_beacon_model_11	floor_4_beacon_11_description	1
45	1765	floor_5_beacon_model_1	floor_5_beacon_1_description	1
46	1788	floor_5_beacon_model_2	floor_5_beacon_2_description	1
47	1825	floor_5_beacon_model_3	floor_5_beacon_3_description	1
48	1848	floor_5_beacon_model_4	floor_5_beacon_4_description	1

49	1752	floor_5_beacon_model_5	floor_5_beacon_5_description	1
50	1756	floor_5_beacon_model_6	floor_5_beacon_6_description	1
51	1812	floor_5_beacon_model_7	floor_5_beacon_7_description	1
52	1872	floor_5_beacon_model_8	floor_5_beacon_8_description	1
53	1932	floor_5_beacon_model_9	floor_5_beacon_9_description	1
54	1916	floor_5_beacon_model_10	floor_5_beacon_10_description	1
55	1976	floor_5_beacon_model_11	floor_5_beacon_11_description	1

User information simulation

- adminUser
- customer

```
# Add admin user data
admin_user_data = [[1,'admin','admin']]
```

```
admin_user_columns = ['idadminUser','name','password']
admin_user = pd.DataFrame(data = admin_user_data,columns=admin_user_columns)
```

```
pd.io.sql.to_sql(admin_user,'adminUser',connection,schema='mydb',if_exists='append',index=False)
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/pandas/io/sql.py:1336: UserWarning: The
provided table name 'adminUser' is not found exactly as such in the database after writing the
table, possibly due to case sensitivity issues. Consider using lower case table names.
  warnings.warn(msg, UserWarning)
```

```
%%sql
select * from adminUser
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
1 rows affected.
```

idadminUser	name	password
1	admin	admin

```
# Add customer user data
customer_user_data = []
for i in range(10):
    customer_user_data.append([
        i+1,
        'user_name_'+str(i),
        'password_'+str(i),
        'first_name_'+str(i),
        'last_name_'+str(i),
        False
    ])
customer_user_data.append([
    11,
    'disability_1',
```

```
    'password_'+str(i),
    'first_name_'+str(i),
    'last_name_'+str(i),
    True
])
```

```
customer_user_columns = ['idcustomer','userName','password','firstName','lastName','isdisability']
customer_user = pd.DataFrame(data = customer_user_data,columns=customer_user_columns)
```

```
pd.io.sql.to_sql(customer_user,'customer',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from customer
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
11 rows affected.
```

idcustomer	userName	password	firstName	lastName	isdisability
1	user_name_0	password_0	first_name_0	last_name_0	0
2	user_name_1	password_1	first_name_1	last_name_1	0
3	user_name_2	password_2	first_name_2	last_name_2	0
4	user_name_3	password_3	first_name_3	last_name_3	0
5	user_name_4	password_4	first_name_4	last_name_4	0
6	user_name_5	password_5	first_name_5	last_name_5	0
7	user_name_6	password_6	first_name_6	last_name_6	0
8	user_name_7	password_7	first_name_7	last_name_7	0
9	user_name_8	password_8	first_name_8	last_name_8	0
10	user_name_9	password_9	first_name_9	last_name_9	0
11	disability_1	password_9	first_name_9	last_name_9	1

Positioning test

```
# Generate a device for disability_1
customer_device_data = [
    [1,11,'iphone',0]
]
```

```
customer_device_columns = ['iddevice','idcustomer','deviceTpye','NumConnectionsAvailable']
customer_device = pd.DataFrame(data = customer_device_data,columns=customer_device_columns)
```

```
pd.io.sql.to_sql(customer_device,'device',connection,schema='mydb',if_exists='append',index=False)
```

```
%%sql
select * from device
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
1 rows affected.
```

iddevice	idcustomer	deviceTpye	NumConnectionsAvailable
1	11	iphone	0

```
%%sql
# Select coordinatePoint in a particular building floor
select fmp.coordinatePoint, fmp.x,fmp.y  from floorMap flm
join floorMapPoints fmp
on flm.idfloorMap = fmp.idfloorMap
where flm.idbuilding = 1 and flm.floorNumber = 1
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
420 rows affected.
```

coordinatePoint	x	y
1	0	20
2	1	20
3	2	20
4	3	20
5	4	20
6	5	20
7	6	20
8	7	20
9	8	20
10	9	20
11	10	20
12	11	20
13	12	20
14	13	20
15	14	20
16	15	20
17	16	20
18	17	20
19	18	20
20	19	20
21	0	19
22	1	19
23	2	19

24	3	19
25	4	19
26	5	19
27	6	19
28	7	19
29	8	19
30	9	19
31	10	19
32	11	19
33	12	19
34	13	19
35	14	19
36	15	19
37	16	19
38	17	19
39	18	19
40	19	19
41	0	18
42	1	18
43	2	18
44	3	18
45	4	18
46	5	18
47	6	18
48	7	18
49	8	18
50	9	18
51	10	18
52	11	18
53	12	18
54	13	18
55	14	18
56	15	18
57	16	18
58	17	18
59	18	18
60	19	18
61	0	17
62	1	17
63	2	17

64	3	17
65	4	17
66	5	17
67	6	17
68	7	17
69	8	17
70	9	17
71	10	17
72	11	17
73	12	17
74	13	17
75	14	17
76	15	17
77	16	17
78	17	17
79	18	17
80	19	17
81	0	16
82	1	16
83	2	16
84	3	16
85	4	16
86	5	16
87	6	16
88	7	16
89	8	16
90	9	16
91	10	16
92	11	16
93	12	16
94	13	16
95	14	16
96	15	16
97	16	16
98	17	16
99	18	16
100	19	16
101	0	15
102	1	15
103	2	15

104	3	15
105	4	15
106	5	15
107	6	15
108	7	15
109	8	15
110	9	15
111	10	15
112	11	15
113	12	15
114	13	15
115	14	15
116	15	15
117	16	15
118	17	15
119	18	15
120	19	15
121	0	14
122	1	14
123	2	14
124	3	14
125	4	14
126	5	14
127	6	14
128	7	14
129	8	14
130	9	14
131	10	14
132	11	14
133	12	14
134	13	14
135	14	14
136	15	14
137	16	14
138	17	14
139	18	14
140	19	14
141	0	13
142	1	13
143	2	13

144	3	13
145	4	13
146	5	13
147	6	13
148	7	13
149	8	13
150	9	13
151	10	13
152	11	13
153	12	13
154	13	13
155	14	13
156	15	13
157	16	13
158	17	13
159	18	13
160	19	13
161	0	12
162	1	12
163	2	12
164	3	12
165	4	12
166	5	12
167	6	12
168	7	12
169	8	12
170	9	12
171	10	12
172	11	12
173	12	12
174	13	12
175	14	12
176	15	12
177	16	12
178	17	12
179	18	12
180	19	12
181	0	11
182	1	11
183	2	11

184	3	11
185	4	11
186	5	11
187	6	11
188	7	11
189	8	11
190	9	11
191	10	11
192	11	11
193	12	11
194	13	11
195	14	11
196	15	11
197	16	11
198	17	11
199	18	11
200	19	11
201	0	10
202	1	10
203	2	10
204	3	10
205	4	10
206	5	10
207	6	10
208	7	10
209	8	10
210	9	10
211	10	10
212	11	10
213	12	10
214	13	10
215	14	10
216	15	10
217	16	10
218	17	10
219	18	10
220	19	10
221	0	9
222	1	9
223	2	9

224	3	9
225	4	9
226	5	9
227	6	9
228	7	9
229	8	9
230	9	9
231	10	9
232	11	9
233	12	9
234	13	9
235	14	9
236	15	9
237	16	9
238	17	9
239	18	9
240	19	9
241	0	8
242	1	8
243	2	8
244	3	8
245	4	8
246	5	8
247	6	8
248	7	8
249	8	8
250	9	8
251	10	8
252	11	8
253	12	8
254	13	8
255	14	8
256	15	8
257	16	8
258	17	8
259	18	8
260	19	8
261	0	7
262	1	7
263	2	7

264	3	7
265	4	7
266	5	7
267	6	7
268	7	7
269	8	7
270	9	7
271	10	7
272	11	7
273	12	7
274	13	7
275	14	7
276	15	7
277	16	7
278	17	7
279	18	7
280	19	7
281	0	6
282	1	6
283	2	6
284	3	6
285	4	6
286	5	6
287	6	6
288	7	6
289	8	6
290	9	6
291	10	6
292	11	6
293	12	6
294	13	6
295	14	6
296	15	6
297	16	6
298	17	6
299	18	6
300	19	6
301	0	5
302	1	5
303	2	5

304	3	5
305	4	5
306	5	5
307	6	5
308	7	5
309	8	5
310	9	5
311	10	5
312	11	5
313	12	5
314	13	5
315	14	5
316	15	5
317	16	5
318	17	5
319	18	5
320	19	5
321	0	4
322	1	4
323	2	4
324	3	4
325	4	4
326	5	4
327	6	4
328	7	4
329	8	4
330	9	4
331	10	4
332	11	4
333	12	4
334	13	4
335	14	4
336	15	4
337	16	4
338	17	4
339	18	4
340	19	4
341	0	3
342	1	3
343	2	3

344	3	3
345	4	3
346	5	3
347	6	3
348	7	3
349	8	3
350	9	3
351	10	3
352	11	3
353	12	3
354	13	3
355	14	3
356	15	3
357	16	3
358	17	3
359	18	3
360	19	3
361	0	2
362	1	2
363	2	2
364	3	2
365	4	2
366	5	2
367	6	2
368	7	2
369	8	2
370	9	2
371	10	2
372	11	2
373	12	2
374	13	2
375	14	2
376	15	2
377	16	2
378	17	2
379	18	2
380	19	2
381	0	1
382	1	1
383	2	1

384	3	1
385	4	1
386	5	1
387	6	1
388	7	1
389	8	1
390	9	1
391	10	1
392	11	1
393	12	1
394	13	1
395	14	1
396	15	1
397	16	1
398	17	1
399	18	1
400	19	1
401	0	0
402	1	0
403	2	0
404	3	0
405	4	0
406	5	0
407	6	0
408	7	0
409	8	0
410	9	0
411	10	0
412	11	0
413	12	0
414	13	0
415	14	0
416	15	0
417	16	0
418	17	0
419	18	0
420	19	0

```
%%sql
select * from beacons b
join
# Select coordinatePoint in a particular building floor
```

```
(select fmp.coordinatePoint, fmp.x,fmp.y  from floorMap flm
join floorMapPoints fmp
on flm.idfloorMap = fmp.idfloorMap
where flm.idbuilding = 1 and flm.floorNumber = 1) bfp
on b.coordinatePoint = bfp.coordinatePoint
```

```
 * mysql+pymysql://brickea_mac:***@rm-0xih4pk94w41k3c5j8o.mysql.rds.aliyuncs.com/mydb
11 rows affected.
```

idbeacon	coordinatePoint	beaconModel	description	ifwork	coordinatePoint_1	x	y
5	72	floor_1_beacon_model_5	floor_1_beacon_5_description	1	72	11	17
6	76	floor_1_beacon_model_6	floor_1_beacon_6_description	1	76	15	17
1	85	floor_1_beacon_model_1	floor_1_beacon_1_description	1	85	4	16
2	108	floor_1_beacon_model_2	floor_1_beacon_2_description	1	108	7	15
7	132	floor_1_beacon_model_7	floor_1_beacon_7_description	1	132	11	14
3	145	floor_1_beacon_model_3	floor_1_beacon_3_description	1	145	4	13
4	168	floor_1_beacon_model_4	floor_1_beacon_4_description	1	168	7	12
8	192	floor_1_beacon_model_8	floor_1_beacon_8_description	1	192	11	11
10	236	floor_1_beacon_model_10	floor_1_beacon_10_description	1	236	15	9
9	252	floor_1_beacon_model_9	floor_1_beacon_9_description	1	252	11	8
11	296	floor_1_beacon_model_11	floor_1_beacon_11_description	1	296	15	6

```
# Get beacon data from a particular building floor
building = 1
floor = 1
sql = 'select * from beacons b'\
        ' join'\
        ' (select fmp.coordinatePoint, fmp.x,fmp.y  from floorMap flm'\
        ' join floorMapPoints fmp'\
        ' on flm.idfloorMap = fmp.idfloorMap'\
        ' where flm.idbuilding = '+str(building)+' and flm.floorNumber = '+str(floor)+') bfp'\
        ' on b.coordinatePoint = bfp.coordinatePoint'
beacon_points = pd.read_sql(sql,connection)
beacon_points
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idbeacon	coordinatePoint	beaconModel	description	ifwork	coordinatePoint	x	y
0	5	72	floor_1_beacon_model_5	floor_1_beacon_5_description	1	72	11.0	17.0
1	6	76	floor_1_beacon_model_6	floor_1_beacon_6_description	1	76	15.0	17.0
2	1	85	floor_1_beacon_model_1	floor_1_beacon_1_description	1	85	4.0	16.0
3	2	108	floor_1_beacon_model_2	floor_1_beacon_2_description	1	108	7.0	15.0
4	7	132	floor_1_beacon_model_7	floor_1_beacon_7_description	1	132	11.0	14.0

	idbeacon	coordinatePoint	beaconModel	description	ifwork	coordinatePoint	x	y
5	3	145	floor_1_beacon_model_3	floor_1_beacon_3_description	1	145	4.0	13.0
6	4	168	floor_1_beacon_model_4	floor_1_beacon_4_description	1	168	7.0	12.0
7	8	192	floor_1_beacon_model_8	floor_1_beacon_8_description	1	192	11.0	11.0
8	10	236	floor_1_beacon_model_10	floor_1_beacon_10_description	1	236	15.0	9.0
9	9	252	floor_1_beacon_model_9	floor_1_beacon_9_description	1	252	11.0	8.0
10	11	296	floor_1_beacon_model_11	floor_1_beacon_11_description	1	296	15.0	6.0

```
# Generate random 4 connections between device and beacon
simulated_4_beacons = beacon_points.sample(n=4)
simulated_4_beacons
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idbeacon	coordinatePoint	beaconModel	description	ifwork	coordinatePoint	x	y
9	9	252	floor_1_beacon_model_9	floor_1_beacon_9_description	1	252	11.0	8.0
8	10	236	floor_1_beacon_model_10	floor_1_beacon_10_description	1	236	15.0	9.0
0	5	72	floor_1_beacon_model_5	floor_1_beacon_5_description	1	72	11.0	17.0
10	11	296	floor_1_beacon_model_11	floor_1_beacon_11_description	1	296	15.0	6.0

```
# Get floor map points for a particular building floor
sql = 'select fmp.coordinatePoint, fmp.x,fmp.y,fmp.ifpassable from floorMap flm'\
        ' join floorMapPoints fmp'\
        ' on flm.idfloorMap = fmp.idfloorMap'\
        ' where flm.idbuilding = '+str(building)+' and flm.floorNumber = '+str(floor)
floor_points = pd.read_sql(sql,connection)
floor_points
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	coordinatePoint	x	y	ifpassable
0	1	0.0	20.0	1
1	2	1.0	20.0	1
2	3	2.0	20.0	1
3	4	3.0	20.0	1

	coordinatePoint	x	y	ifpassable
4	5	4.0	20.0	1
...	...	...	...	...
415	416	15.0	0.0	1
416	417	16.0	0.0	1
417	418	17.0	0.0	1
418	419	18.0	0.0	1
419	420	19.0	0.0	1

420 rows × 4 columns

```
# Generate random position for device
simulated_device_current_point = floor_points.sample(n=1)
while simulated_device_current_point.ifpassable.item() == 0:
    # The random position for device must be passable
    simulated_device_current_point = floor_points.sample(n=1)
simulated_device_current_point
```

```
/Users/brickeawang/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:3: FutureWarning:
`item` has been deprecated and will be removed in a future version
  This is separate from the ipykernel package so we can avoid doing imports until
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	coordinatePoint	x	y	ifpassable
394	395	14.0	1.0	1

```
# Calculate simulated distance between device and beacons
beacon_device_distance = []
for i in range(len(simulated_4_beacons)):
    distance = np.sqrt(np.square(simulated_device_current_point.iloc[0].x -
simulated_4_beacons.iloc[i].x)
                       + np.square(simulated_device_current_point.iloc[0].y -
simulated_4_beacons.iloc[i].y))
    beacon_device_distance.append((distance))
beacon_device_distance
```

```
[7.615773105863909, 8.06225774829855, 16.278820596099706, 5.0990195135927845]
```

```
# Trilateration
def trilateration(beacons=None,distance=None):
    A = 2*(beacons.iloc[2].y - beacons.iloc[0].y)
    A_ = 2*(beacons.iloc[1].y - beacons.iloc[0].y)
    B = 2*(beacons.iloc[2].x - beacons.iloc[0].x)
```

```
    B_ = 2*(beacons.iloc[1].x - beacons.iloc[0].x)

    print(A)
    print(A_)
    print(B)
    print(B_)

    delta_1 = np.square(distance[0]) - np.square(distance[2]) + np.square(beacons.iloc[2].y) -
np.square(beacons.iloc[0].y) + np.square(beacons.iloc[2].x) - np.square(beacons.iloc[0].x)
    delta_2 = np.square(distance[0]) - np.square(distance[1]) + np.square(beacons.iloc[1].y) -
np.square(beacons.iloc[0].y) + np.square(beacons.iloc[1].x) - np.square(beacons.iloc[0].x)

    print(delta_1)
    print(delta_2)

    device_x = (delta_1 * A_ - delta_2 * A)/(B * A_ - B_ * A)
    device_y = (delta_1 * B_ - delta_2 * B)/(B_ * A - B * A_)

    return (round(device_x),round(device_y))
```

```
simulated_4_beacons.iloc[:3]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	idbeacon	coordinatePoint	beaconModel	description	ifwork	coordinatePoint	x	y
9	9	252	floor_1_beacon_model_9	floor_1_beacon_9_description	1	252	11.0	8.0
8	10	236	floor_1_beacon_model_10	floor_1_beacon_10_description	1	236	15.0	9.0
0	5	72	floor_1_beacon_model_5	floor_1_beacon_5_description	1	72	11.0	17.0

```
beacon_device_distance[:3]
```

```
[7.615773105863909, 8.06225774829855, 16.278820596099706]
```

```
trilateration(beacons = simulated_4_beacons.iloc[:3],distance = beacon_device_distance[:3])
```

```
18.0
2.0
0.0
8.0
18.0
114.00000000000003



(14.0, 1.0)
```