به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

تحقیق در عملیات

پروژه پایانی

امیرحسین دبیری اقدم

810197502

بهمن ماه 1400

فهرست

عنوان	شماره صفحه
چکیده	3
بخش 1 – جمع آوری دیتاست	4
بخش 2 – مسیریابی با کمترین هزینه	6

چکیده

در این پروژه ابتدا با کمک نقشه گوگل چندین مکان در مناطق 3 و 6 تهران انتخاب شد و اطلاعات متناظر با آن‌ها نظیر مختصات جغرافیایی و فاصله نسبی براساس وضعیت ترافیک و ... استخراج شد و این اطلاعات در یک فایل *csv* ذخیره شد. در ادامه گراف جهت‌دار وزن‌دار متناظر با آن‌ها رسم و نقاط متناظر نیز روی تصویر نقشه هایلایت شد.

در نهایت مسئله کوتاه‌ترین مسیر در یک گراف به صورت یک مسئله بهینه‌سازی خطی که در درس تحقیق در عملیات آن را فرا گرفته بودیم و به صورت گسترده مورد بحث و بررسی قرار گرفته بود، مدل‌سازی شد، و مسئله برای اطلاعات استخراج شده از نقشه گوگل، با زبان پایتون و با کمک کتاب‌خانه *Pulp* مدل‌سازی و حل شد و خروجی مطلوب شامل مسیر بهینه و هزینه آن برای رفتن از هر کدام از مکان‌ها به دیگری، حاصل شد.

بخش اول – جمع آوری دیتاست

- **استخراج داده:**

همانطور که در صورت پروژه گفته شده بود، با استفاده از گوگل مپ 11 مکان شامل بیمارستان بقیه الله، مرکز قلب تهران، موزه هنرهای معاصر تهران، سینما آفریقا، متروی قلهک، پل طبیعت، پارک ملت، متروی میرزای شیرازی، میدان ولیعصر، بیمارستان کودکان مفید را در مناطق 3 و 6 تهران انتخاب کرده و طول و عرض جغرافیایی هر کدام را استخراج کردم. همچنین براساس فاصله آن‌ها از یکدیگر برای هرکدام از این مکان‌ها تعدادی همسایه در نظر گرفتم و بر این اساس و نیز براساس میانگین زمانی که طول می‌کشد از هر کدام از از مکان‌ها به مکان دیگر رفت (که به وضعیت ترافیک،ساعت شبانه روز و... وابسته است که این اطلاعات از طریق گوگل مپ در دسترس است) یکسری هزینه به صورت نسبی برای رفتن از هر مکان به همسایه‌هایش و برعکس در نظر گرفتم. در نهایت این اطلاعات در یک فایل با نام *Dataset.csv* ذخیره کردم که پیوست شده است.

همچنین در جدول 1 مشخصات مذکور برای هر مکان آمده است:

جدول 1 – مشخصات مکان‌های انتخاب شده (رنگ‌ها متناظر با این مکان‌ها روی شکل‌های 2 و 3 است برای یافتن ساده‌تر آن‌ها)

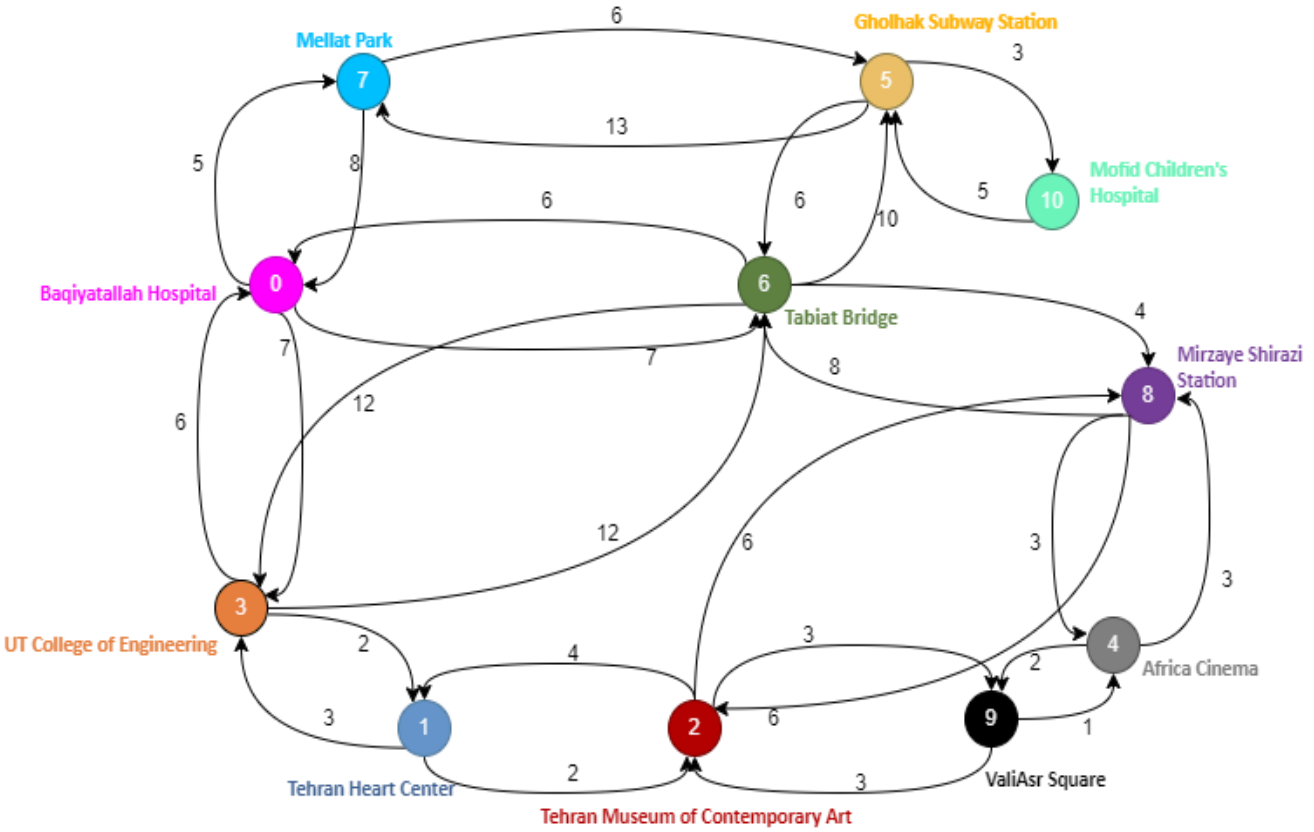
Place_index	Place_name	Lat, Long	Neighbors_indice	Neighbors_weight
0	Baqiyatallah Hospital	35.756, 51.395	3,6,7	7,7,5
1	Tehran Heart Center	35.720, 51.389	2,3	2,3
2	Tehran Museum of Contemporary Art	35.711, 51.391	1,8,9	4,6,3
3	UT College of Engineering	35.724, 51.388	0,1,6	6,2,12
4	Africa Cinema	35.716, 51.408	8,9	3,2
5	Gholhak Subway Station	35.773, 51.438	6,7,10	6,6,3
6	Tabiat Bridge	35.754, 51.420	0,3,5,8	6,12,10,4
7	Mellat Park	35.778, 51.411	0,5	8,13
8	Mirzaye Shirazi Station	35.728, 51.417	2,4,6	6,3,8
9	ValiAsr Square	35.712, 51.407	2,4	3,1
10	Mofid Children's Hospital	35.734, 51.329	5	5

- **گراف متناظر با داده استخراج شده**

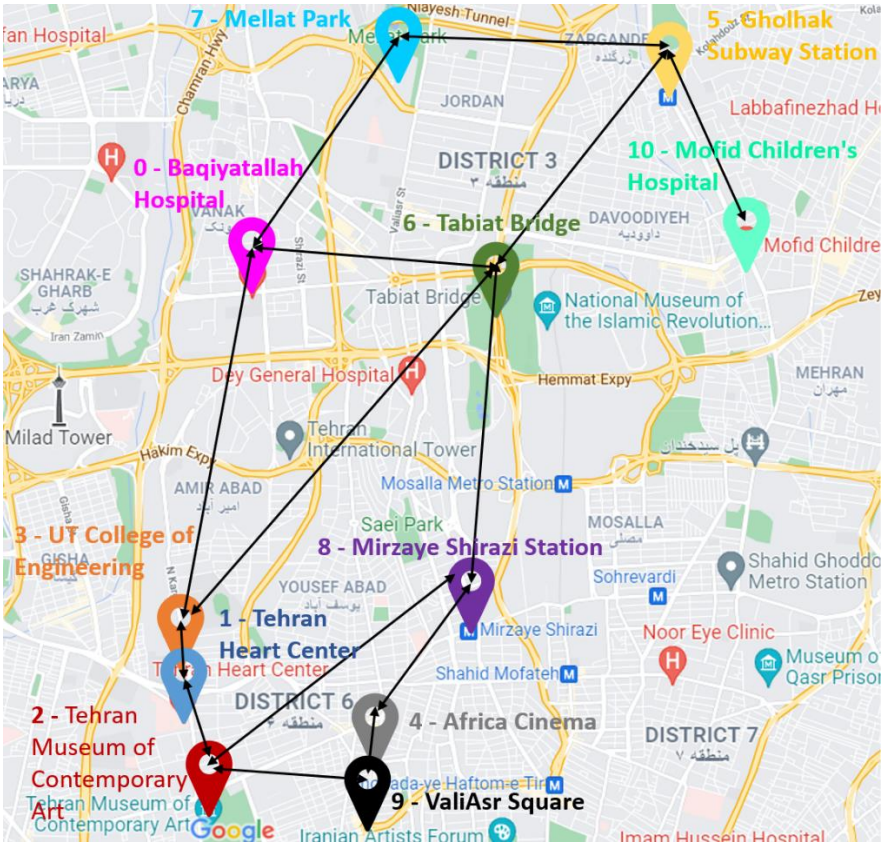
پس از استخراج داده‌ها با کمک سایت *DrawIO* اطلاعات جدول فوق را به صورت گراف درآوردم که در شکل 1 آمده است.

همچنین مکان‌های مذکور را روی نقشه با نماد 📍 مشخص کرده‌ام که در شکل 2 آمده است.

در شکل 2 مشاهده می‌شود که سعی کرده‌ام تا حد امکان تمام مناطق 3 و 6 تحت پوشش قرار بگیرند.



شکل 1 – گراف جهت‌دار وزن‌دار متناظر با مکان‌های استخراج شده



شکل 2 – مکان‌های استخراج شده روی نقشه

بخش دوم – مسیریابی با کمترین هزینه

- **مدل‌سازی ریاضی و حل مسئله**

مسئله یافتن کوتاه‌ترین مسیر از یک نود به یک نود دیگر در گراف جهت‌دار وزن‌دار یکی از معروف‌ترین مسائل تئوری گراف است؛ به همین سبب روش‌های متنوعی برای حل آن ارائه شده است. از میان این روش‌ها می‌توان به دو روش اشاره کرد که به صورت مستقیم در ارتباط با مباحث ارائه شده در درس تحقیق در عملیات است. روش نخست الگوریتم دایکسترا[1] است؛ این الگوریتم که یک الگوریتم برنامه‌ریزی پویا[2] است که به نوعی بر پایه اصل بهینگی بلمن طراحی شده است و از این واقعیت استفاده می‌کند که اگر نود c از یک گراف جهت‌دار وزن‌دار، نودی در کوتاه‌ترین مسیر بین دو نود a و b باشد، آنگاه کوتاه‌ترین مسیر از a تا c، همان زیر مسیری است که در کوتاه‌ترین مسیر بین دو نود a و b قرار دارد.

روش دیگر مدل کردن این مسئله به صورت یک مسئله برنامه ریزی خطی است؛ برای این کار برای هر یال از نود i به نود j در گراف، متغیر باینری x_{ij} را تعریف می‌کنیم که اگر این یال در کوتاه‌ترین مسیر وجود داشته باشد مقدار آن 1 و در غیر اینصورت 0 خواهد بود پس اگر وزن یال‌ها را به صورت $w_{ij} \neq 0$ در نظر بگیریم آنگاه تابع هزینه به صورت $\min J = \sum_{i,j} w_{ij}x_{ij}$ قابل تعریف است. اگر فرض کنیم که به دنبال کوتاه‌ترین مسیر از نود m به نود l $(m \neq l)$ هستیم آنگاه قیودی نیز باید بر مسئله فوق حاکم باشد وگرنه جواب بدیهی و نامطلوب $x_{ij} = 0$ حاصل می‌شود؛ قیودی که باید حاکم باشد یکی این است که تعداد یال‌های خروجی از نود m (نود شروع) باید یکی بیشتر از تعداد یال‌های ورودی به آن باشد تا حتماً نقطه شروع مسیر باشد؛ تعداد یال‌های خروجی از نود l (نود پایان) باید یکی کمتر از تعداد یال‌های ورودی به آن باشد تا حتماً نقطه پایان مسیر باشد و برای بقیه نودها تعداد یال‌های خروجی و ورودی باید برابر باشد تا اگر در طول مسیر بهینه وارد آن‌ها شدیم، حتما خارج شویم و یا اینکه اصلاً وارد آن‌ها نشویم. قیود گفته شده تضمین می‌کند که ما یک مسیر از نود m به نود l بدست می‌آوریم که با کمینه کردن تابع هزینه، این مسیر نیز کوتاه‌ترین مسیر خواهد بود. (این مدل‌سازی حتی برای حالاتی که یال‌ها مقدار منفی داشته باشند یا حالاتی که کوتاه‌ترین مسیر با گذر از یک نود بیش از یک بار حاصل می‌شود نیز قابل استفاده است)

بنابراین مسئله کوتاه‌ترین مسیر در یک گراف جهت‌دار وزن‌دار را به صورت مسئله برنامه ریزی خطی زیر می‌توان مدل کرد:

$$\text{minimize } J = \sum_{i,j} w_{ij}x_{ij}\ ;\ where\ x_{ij}\ are\ Binary\ (i.e.\ x_{ij} \in \mathbb{Z};\ 0 \leq x_{ij} \leq 1)\ and\ also\ w_{ij} \neq 0$$

$$subject\ to: \begin{cases} \sum_j x_{mj} = \sum_j x_{jm} + 1 & ;source\ node \\ \sum_j x_{lj} = \sum_j x_{jl} - 1 & ;sink\ node \\ \sum_j x_{ij} = \sum_j x_{ji} & ;for\ every\ i \neq m,l\ (other\ nodes) \end{cases}$$

[1] Dijkstra's algorithm
[2] Dynamic Programming

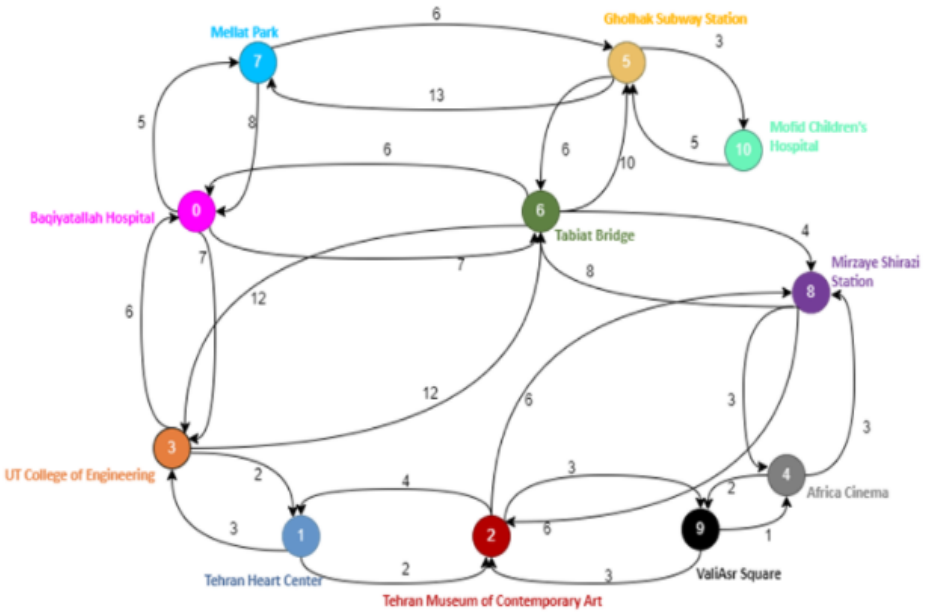
- **پیاده‌سازی و حل مسئله با پایتون**

از آنجا که در صورت پروژه خواسته شده مسئله را به صورت یک مسئله بهینه‌سازی مدل کنیم پس با دومین روش گفته شده مسئله را مدل و حل می‌کنم. همچنین مشابه تمرین‌های در طول ترم، در این پروژه هم از کتاب‌خانه قدرتمند *Pulp* برای پیاده‌سازی و حل مسئله فوق الذکر استفاده کردم که کدهای آن در فولدر *code*، و در نوت‌بوک ژوپیتر با نام *code.ipynb* ضمیمه شده است.

در این نوت‌بوک سعی کردم‌ام که تا حد امکان مراحل مسئله را به بخش‌های کوچک بشکنم و برای هر قسمت از کد نیز کامنتی کوتاه قرار داده ام؛ تابع *FindShortestPath* بخش اصلی کد است که دیتاست و نود مبدا و مقصد را گرفته و براساس آن‌ها و مدل‌سازی گفته شده، مسئله را پیاده و حل می‌کند و در نهایت خروجی خواسته شده را چاپ می‌کند (در مدل‌سازی گفته شده فرض شده بود که نود شروع و پایان یکسان نیست یعنی $l \neq m$ اما در کد برای این حالت هم فرض کرده‌ام که مسیر بهینه با هزینه صفر است که از همان نود به خودش است.)

همچنین تابع کمکی *RunMe* را برای خواندن فایل *Dataset.csv* و چاپ مکان‌های بدست آمده از آن (با فراخوانی تابع کمکی *PrintLocations*) و نیز گرفتن اندیس مکان (نود) شروع و پایان از کاربر، نوشته‌ام که اگر ورودی مناسب نباشد مثلاً شماره نود غیر مجاز وارد شود یا ... خطای مناسب چاپ می‌شود و در غیر اینصورت خروجی خواسته شده شامل اندیس و مختصات نود مبدا و مقصد، مسیر بهینه و هزینه آن چاپ می‌شود.

در تصویر 3 یک نمونه از اجرای آخرین سلول نوت‌بوک *code.ipynb* آمده است که در آن تابع *RunMe* فراخوانی شده است؛ کاربر در آن مبدا را دانشکده فنی دانشگاه تهران (اندیس 3) و مقصد را سینما آفریقا (اندیس 4) تعیین کرده است و در خروجی آنچه در صورت پروژه گفته شده بود یعنی اندیس و مختصات نود مبدا و مقصد، مسیر بهینه و هزینه آن به درستی تعیین و چاپ شده است.




```
RunMe()
```

```
Following locations are available:
index = 0, name = Baqiyatallah Hospital, coordinates = 35.756, 51.395
index = 1, name = Tehran Heart Center, coordinates = 35.720, 51.389
index = 2, name = Tehran Museum of Contemporary Art, coordinates = 35.711, 51.391
index = 3, name = University of Tehran College of Engineering, coordinates = 35.724, 51.388
index = 4, name = Africa Cinema, coordinates = 35.716, 51.408
index = 5, name = Gholhak Subway Station, coordinates = 35.773, 51.438
index = 6, name = Tabiat Bridge, coordinates = 35.754, 51.420
index = 7, name = Mellat Park, coordinates = 35.778, 51.411
index = 8, name = Mirzaye Shirazi Station, coordinates = 35.728, 51.417
index = 9, name = ValiAsr Square, coordinates = 35.712, 51.407
index = 10, name = Mofid Children's Hospital, coordinates = 35.734, 51.329

Please Enter index of your desired origin: 3
Please Enter index your desired destination: 4

Origin: index = 3, Latitude, Longtitude = 35.724, 51.388
Destination: index = 4, Latitude, Longtitude = 35.716, 51.408

Optimal Route with the total cost of 8:
3 -> 1 -> 2 -> 9 -> 4
```

شکل 3 – یک نمونه ورودی و خروجی از کد پایتون ارائه شده