

**گزارش پروژه اول قسمت اول**

**پروژه یافتن بهترین پرواز**

**درس: مبانی و کاربردهای هوش مصنوعی**

**استاد راهنما: دکتر حسین کارشناس نجف آبادی**

**اعضای گروه:**

**علی‌اکبر احراری- 4003613001**

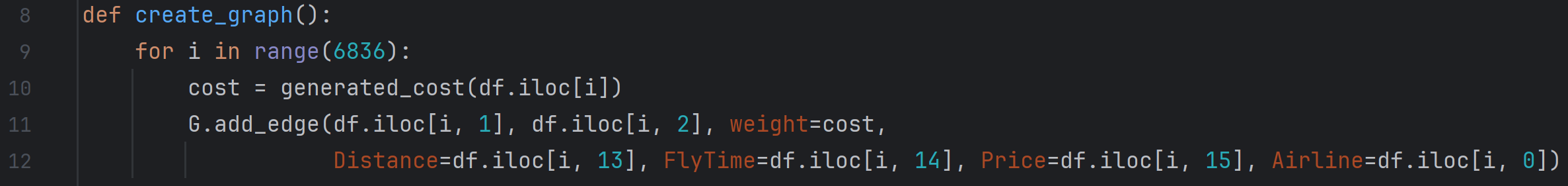
**مهرآذین مزروق- 4003613055**

**پاییز 1402**

# گزارش کار الگوریتم

## توابع مشترک

### create\_graph



در این الگوریتم، تمامی بلیت‌ها به عنوان یال گراف، به گراف اضافه می‌شوند.

### generated\_cost

A computer screen shot of a program code

Description automatically generated

این تابع، هزینه هر بلیت را محاسبه می‌کند. هزینه هر بلیت، تابعی خطی از قیمت بلیت، فاصله مبدا و مقصد و زمان پرواز می‌باشد

به علت اینکه FlyTime به‌طور میانگین، عددی تک رقمی است و اهمیت کمتری نسبت به دو پارامتر دیگر دارد، ضریب ۱۰۰ را دریافت می‌کند.

Distance به‌طور میانگین، سه‌رقمی است واولویت اول را در مسیریابی دارد؛ درنتیجه ضریب ۳ را دریافت می‌کند.

Price به‌طور میانگین، ۲ رقمی‌است و اولویت دوم را دارد؛ بنابراین ضریب 20 را دریافت می‌کند.

### desired\_result\_string

A screen shot of a computer program

Description automatically generated

در این تابع، رشته‌ای که در نهایت در فایل‌ها ذخیره می‌شود، طبق صورت سوال، تولید می‌شود.

### main function

A screen shot of a computer program

Description automatically generated

در این بخش، ابتدا گراف ساخته می‌شود، سپس از کاربر، فرودگاه مبدا و مقصد طبق صورت سوال گرفته می‌شود.

فایل مربوط به الگوریتم A\* ساخته و خود الگوریتم اجرا می‌شود. نتیجه‌ی اجرای الگوریتم در فایل ذخیره می‌شود و سپس دقیقا همین روند برای الگوریتم Dijkstra اجرا می‌شود.

پس از پایان اجرای هر دو الگوریتم و ساخت هر دو فایل output، برنامه پیامی مبنی بر پایان ساخت فایل نشان می‌دهد.

## الگوریتم دایکسترا

### dijkstra\_algorithm

A computer screen shot of a program

Description automatically generated

این تابع دو ورودی مبدا و مقصد را گرفته و یک لیست به عنوان خروجی باز می‌گرداند. کار این تابع بدین شکل است که ابتدا یک دیکشنری به عنوان shortest\_paths ایجاد می‌کند که در آن کوتاه ترین مسیر تا هر نقطه را ذخیره می‌کند.(واضح است که فاصله اولین نقطه تا خودش 0 است). سپس یک صف با نام queue ایجاد می‌شود که در آن نقاط با فاصله کمتر دارای اولویت بیشتری می‌باشند.

با وارد شدن به while، حلقه تا زمانی که صف خالی نشود ادامه می‌دهد. در خط بعدی با استفاده از heapq، نقطه با کمترین فاصله را حذف می‌کنیم. با وارد شدن به حلقه for، برای هر همسایه از نقطه فعلی: 1- فاصله قدیمی را بدست می‌آوریم. 2- فاصله جدید را نیز بدست می‌آوریم. 3- اگر فاصله جدید از قدیم کوتاه تر بود آنگاه مسیر را ذخیره می‌کنیم.

سپس یک لیست خالی از مسیر می‌سازیم که قرار است با پیمایش مسیر از مقصد به مبداء رسیده و مسیر مورد نظر را به عنوان خروجی تابع برگردانیم.

## الگوریتم A\*

### Calculate\_distance

A computer screen shot of a program code

Description automatically generated

این تابع با دریافت دو پارامتر Latitude و Longitude فرودگاه مبدا و مقصد، فاصله‌ی فیزیکی دو فرودگاه را محاسبه می‌کند. دلیل محاسبات بالا، کروی بودن شکل زمین و تفاوت فاصله در شکل کروی نسبت به شکل مسطح می‌باشد.

### a\_star\_heuristic

**A screen shot of a computer code

Description automatically generated**

این تابع با دریافت نام فرودگاه مقصد، در حلقه‌ی for اول، ابتدا Latitude و Longitude فرودگاه را به‌دست می‌آورد. سپس در حلقه‌ی بعد، از کل جدول، پارامتر‌های Latitude و Longitude بقیه فرودگاه‌ها را به‌دست می‌آورد. این دو پارامتر به محاسبه‌ی فاصله‌ی فیزیکی دو فرودگاه، کمک می‌کند. دلیل اینکه ضریب فاصله 67043 است، این است که در دیتاست، قیمت و فاصله‌ی زمانی، رابطه‌ی مستقیم و خطی با فاصله‌ی فیزیکی دارند. بنابراین می‌توان از جمع ضرایبی که از تقسیم فاصله‌فیزیکی، به فاصله‌ی زمانی یا قیمت به‌دست می‌آیند، به عنوان ضریب مناسبی برای تخمین تابع heuristic استفاه کرد.

### a\_star\_algorithm

A computer screen shot of a program

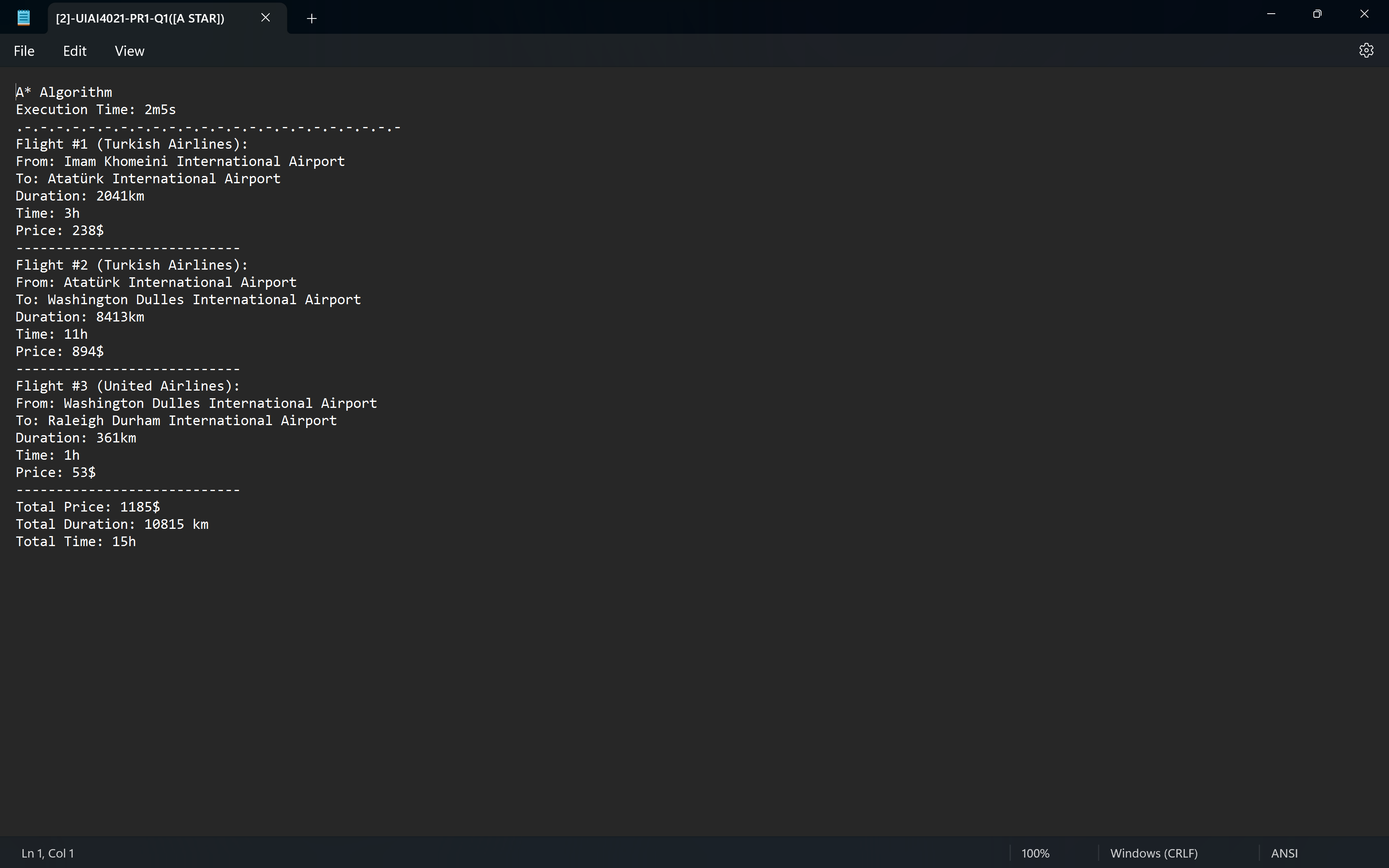
Description automatically generatedاین تابع در ابتدا با دریافت نام فرودگاه مبدا و مقصد، با توجه به فرودگاه مقصد، مقدار heuristic هر فرودگاه را ذخیره می‌کند.سپس صف اولویتی با فرودگاه مبدا می‌سازد؛ و visited را ستی از فرودگاه‌های بازدیدشده تشکیل می‌دهد. cost\_so\_far هزینه هر گره تا الان می‌باشد. came\_from گره‌ای که از آن آمده‌ایم. current به معنای گره فعلی می‌باشد.

تا زمانی که صف خالی نشده‌است، گره‌ای که کمترین هزینه را دارد از صف برمیداریم و اگر گره فعلی، مقصد باشد، حلقه متوقف می‌شود. درغیراین‌صورت، گره فعلی به مجموعه گره‌های بازدید شده اضافه می‌شود. سپس برای هر گره همسایه‌ (فرودگاهی که از این فرودگاه برایش بلیت موجود است) هزینه جدید را محاسبه می‌کنیم و اگر گره جدید کمترین هزینه را داشته باشد، هزینه اصلی بروزرسانی می‌شود، اولویت این گره مشخص می‌شود، گره را به صف اولویت اضافه می‌کنیم.

در نهایت، از روی came\_from مسیر نهایی را مشخص می‌کنیم.

# نمونه‌ای از خروجی

## A\*



## Dijkstra

A screenshot of a computer

Description automatically generated

# کتابخانه‌های استفاده شده

networkx : برای ساخت گراف، استفاده از گره‌ها و یال‌های گراف

pandas : استفاده از دیتا‌های دیتاست

math : به‌دست آوردن فاصله‌ی دو فرودگاه با استفاده از طول و عرض جغرافیایی فرودگاه‌ها

heapq : صف هر دو الگوریتم

time : به‌دست آوردن زمان اجرای هر الگوریتم

# منابع

<https://www.w3schools.com/python/pandas/default.asp>

<https://www.udacity.com/blog/2021/10/implementing-dijkstras-algorithm-in-python.html>

<https://pypi.org/project/networkx/>

<https://blog.enterprisedna.co/python-write-to-file/#:~:text=The%20write()%20method%20is,it%20to%20the%20specified%20file>

[Bing Chat with GPT-4](https://www.bing.com/search?form=NTPCHB&q=Bing+AI&showconv=1)