1402/11/10

Project 1

(Document)

Adel Azizi

بخش اول : منطق

منطق کلی پروژه مانند پروژه ی قبل است و ابتدا داده هارا تبدیل به گراف کرده

سپس با استفاده از الگوریتم (دی اف اس) بلند ترین مسیر را دراین گراف پیدا میکنیم

بخش دوم: توابع و ماژول ها

1. get_input:

ورودی: یک عدد صحیح

در این تابع به تعداد عدد ورودی تابع از کاربر ورودی دریافت میکنیم . که این ورودی ها گروه خونی بیماران و همراهان انها است . سپس این گروه خونی ها را در دو لیست ذخیره میکنیم .

(خروجی:دو لیست (گروه خونی اهداکننده ها و دریافت کننده ها

2. match_blood:

(ورودی :دو عدد صحیح(اولی گروه خونی اهدا کننده و دومی گروه خونی دریافت کننده

این تابع با داشتن لیست ایمکه کدام گروه خونی میتواند به کدام گروه خونی کلیه بدهد ، برسی میکند که ببیند ایا این اهدا کننده میتواند به این دریافت کننده کلیه بدهد یا نه

(خروجی :یک مقدار بولین(درست یا غلط

3. make_graph:

ورودی :دو عدد لیست که در یک از انها گروه خونی اهدا کننده ها و در دیگری گروه خونی دریافت کننده ها قرار دارد.

این تابع با داشتن این دو لیست در ابتدا با استفاده از تابع شماره ی(2) برسی میکند کههر یک از اعضای لیست اهدا کننده ها به کدام یک از اعضای لیست دریافت کننده ها میتواند کلیه بدهد . سپس یک تابل از ان را در لیست یال ها ذخیره میکند

سپس با استفاده از لیست یال ها یک گراف با تایپ دیکشنری میسازیم که کی های ان راس های گراف و ولیو ها ان همسایه های ان راس در گراف است

سپس یک لیست از بن بست ها درست میکنیم .تا در مراحل بعدی وارد انها نشویم

4. `find_longest_chain`:

ورودی: یک گراف که تایپ ان دیکشنری است و یک لیست از بن بست ها است.

این تابع با استفاده از تابع (دی اف اس)و یک حلقه ی (فور) طول بلند ترین زنجیره را در گراف ورودی نشان میدهد .

خروجی:یک عدد صحیح که طول بلدترین مسیر ساده گراف است و یک لیست که بیان گر بلند ترین زنجیره ی ممکن است

3. dfs:

ورودی ها:

node (int):

راسی از یال که روی ان هستیم)

visited (set):

یک مجموعه از راس هایی که از انها بازدید کردیم

path (list):

یک لیست که مسیر راس های بازدید شده رو نشون میده

دراین تابع با استفاده از الگوریتم (دی اف اس) که قبلا توضیح ان را داده ایم طول بزرگ ترین زنجیره را میابیم . این الگورتیم در این تابع به این صورت است که با دریافت ورودی ها ابتدا راس در حال بازدید را به عنوان بازدید شده علامت میزند و ان را به مجموعه ی بازدید شده ها اضافه می کند سپس ان را به لیست زنجیره اضافه میکند . سپس در همسایه های ان میگردد و اگر بازدید نشده بودند دوباره این تابع را برای انها فراخوانی میکند . این روند تا جایی ادامه پیدا میکند که تمامی همسایه های ان راس دیده شده باشند سپس اخرین درایه ی لیست مسیر را که همان راس فعلی است از لیست مسیر ها پاک میکنه و همچنین ان راس را از مجموعه ی بازدیده شده ها پاک . میکنه اینشکلی یک مرحله به عقب برگشتیم و همسایه های راس قبلی رو دوباره بازدید میکنیم و اگر دیده شده بودند دوباره یک مرحله بر میگردیم عقب . این روند به شکلی ادامه پیدا میکند که تمام مسیر های ساده ممکن در این گراف دید شده باشن و ما با در نظر گرفتن یک متغییر برای طول طولانی ترین مسیر .هر بار طول . مسیر فعلی را با ان متغییر میسنجیم و اگه بزرگ تر بود ان را اپدیت میکنیم

خروجی:

None

بخش سوم : نمونه

ورودی :

6

300 100

300 200

101 200

231 301

230 100

201 300

خروجی:

Length of longest chain: 5

Longest chain: [1, 2, 5, 3, 6]