عنوان: **درخت‌ها و گراف‌ها قسمت پنجم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۲/۲۷ ۱۹:۳۰
آدرس: www.dotnettips.info
گروه‌ها: Data Structures, Tree, graph

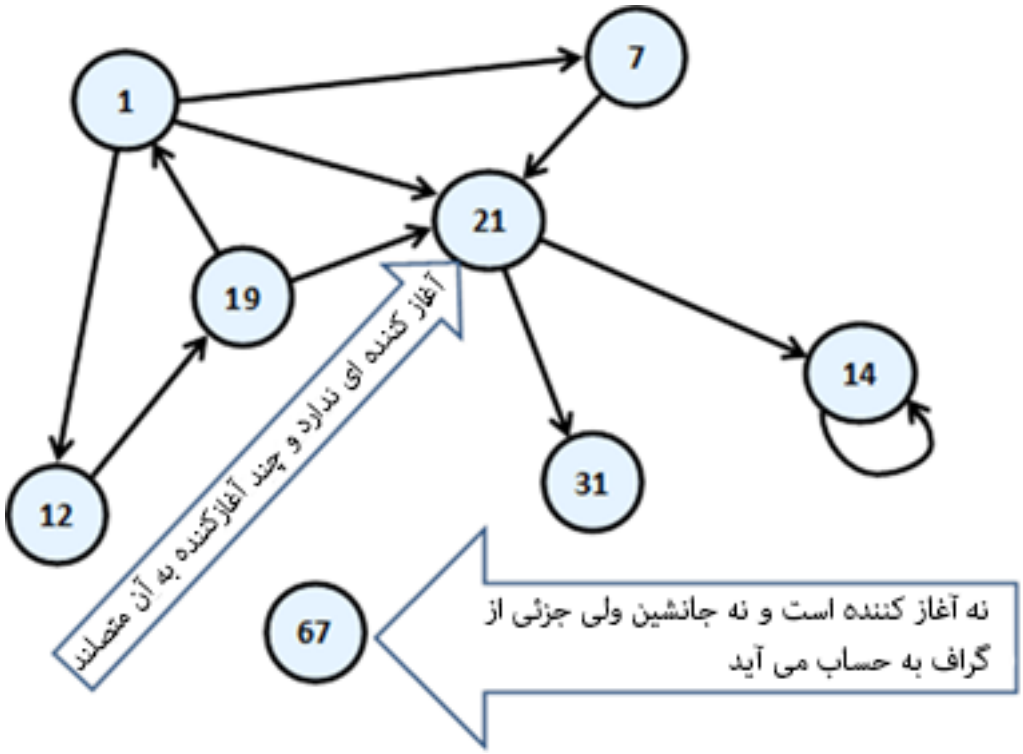
در ، قسمت قبلی پیاده سازی درخت‌ها را بررسی کردیم و در این قسمت مبحث گراف‌ها را آغاز می‌کنیم .

گراف‌ها یکی از پر اهمیت‌ترین و همچنین پر استفاده‌ترین ساختارها هستند و به خوبی روابط بین تمامی اشیاء را نشان می‌دهند و در عمل تقریبا همه چیز را پشتیبانی می‌کنند. در ادامه متوجه خواهید شد که درخت‌ها، زیر مجموعه‌ای از گراف‌ها هستند و همانطور که می‌دانید لیست‌ها هم زیر مجموعه‌ی درخت‌ها هستند. پس لیست‌ها هم زیر مجموعه‌ی گراف‌ها می‌شوند .

مفهوم پایه‌ای گراف

در این بخش تعدادی از مهمترین خصوصیات گراف‌ها را بررسی می‌کنیم که تعدادی از آن‌ها بسیار شبیه به ساختمان درخت‌هاست، ولی تفاوت‌های زیادی با هم دارند؛ زیرا که درخت، خود نوع متفاوتی از ساختمان گراف‌ها است .

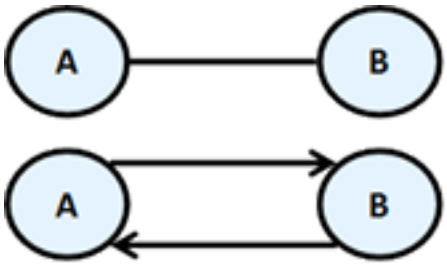
درخت زیر را در نظر بگیرید؛ این درخت هم مانند سایر درخت‌ها با گره‌های شماره دار، نامگذاری شده است که تشخیص آن‌ها را برای ما آسان‌تر می‌سازد. در گراف، به گره‌ها **راس** یا **vertice** هم می‌گویند. هر چند نام گره هم برای آن‌ها به کار برده می‌شود. به فلش‌هایی که به این رئوس اشاره می‌کنند، **لبه‌های جهت دار directed edges** گفته می‌شود که در ویکی پدیا و کتب آموزشی فارسی، به آن‌ها یال اطلاق می‌شود . به یال هایی که از هر راس بیرون می‌آیند **Predecessor** گفته می‌شود که به معنی آغاز کننده است و به راسی که اشاره می‌کند، **Successor** گویند که به معنی ارث برنده یا جایگزین شناخته می‌شود. در شکل زیر عدد راس 19 آغاز کننده راس 1 است و 1 هم جایگزین یا ارث برنده 19 و اگر با دقت به شکل نگاه کنید می‌بینید که یک راس می‌تواند از چند راس ارث برنده باشد؛ مثل راس 21 .



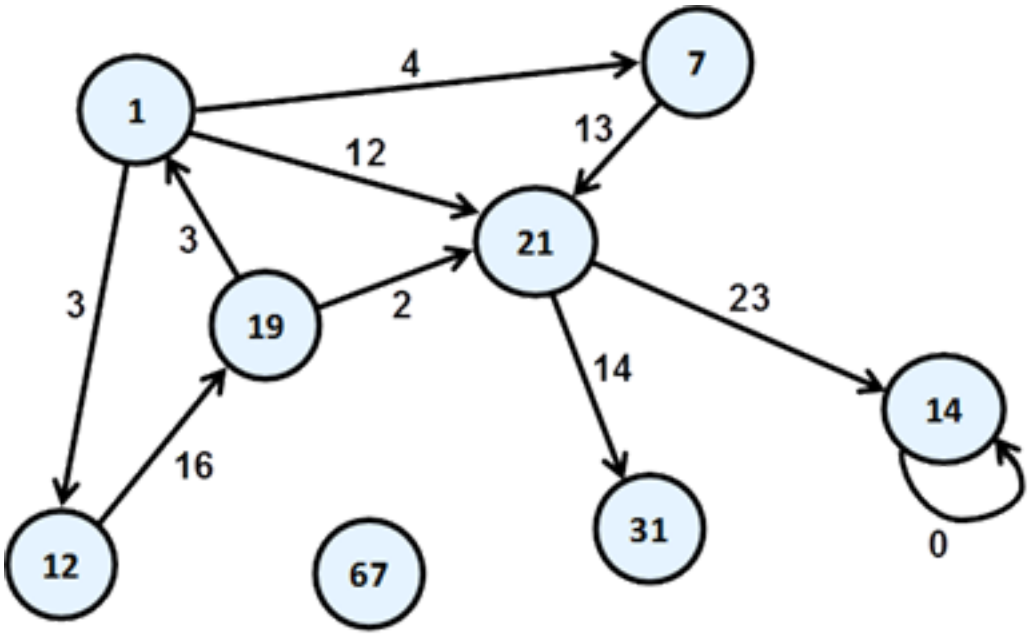
برای نمایش گراف، ما از عبارت (V,E) استفاده می‌کنیم که V مجموعه‌ای از راس‌ها و E مجموعه‌ای از لبه‌هاست. هر لبه (که با e

کوچک نمایش داده می‌شود) و در مجموعه E قرار دارد، پیوند دو راس v و u را نشان می‌دهد یا به عبارتی به صورت ریاضی می‌شود e=(u,v) .
برای اینکه مطالب را بهتر درک کنیم بهتر است که هر راس را یک شهر و هر لبه را یک جاده‌ی یک طرفه برای ارتباط با این راس‌ها فرض کنیم. مثلا اگر یکی از راس‌ها را تهران تصور کنیم و دیگری را کاشان، لبه یا جاده‌ی یک طرفه‌ای که این دو شهر را به هم متصل می‌کند، می‌شود جاده یا لبه‌ی تهران کاشان.

در بعضی مواقع از لبه‌های بدون جهت استفاده می‌شود که این لبه‌ها را لبه‌های دو طرفه می‌گویند؛ مثل جاده‌ی دو طرفه. گاهی هم از دو لبه‌ی جهت دار به جای یک لبه‌ی بدون جهت استفاده می‌کنند که نمونه‌ی آن را در شکل زیر می‌بینید.



دو راسی که به وسیله‌ی یک یال به یکدیگر متصل می‌شوند را **همسایه Neighbor** می‌نامند. هر یال می‌تواند یک عدد برای خود داشته باشد که به این عدد وزن یال یا لبه می‌گویند **Weight (Cost)** و در مثال بالا می‌توان گفت وزن هر یال می‌شود مسافت آن جاده؛ مسافتی که بین دو شهر همسایه باید طی شود. تصویر زیر یک گراف را نشان می‌دهد که وزن یال‌های آن در کنار هر یال نوشته شده است.



مسیر Path در گراف همانند درخت‌ها، طی کردن مسیری است که از یک راس به راس دیگر می‌رسد. در مثال بالا باید گفت که برای رسیدن از شهر مبدا به شهر مقصد، باید از چه شهرهایی عبور کرد. در شکل بالا مسیر 1 - 12 - 19 - 21 - 7 - 21 و 1 مسیر نیستند؛ چرا که راس 21 هیچ لبه‌ی آغاز کننده‌ای ندارد و بیشتر ارث برنده است.

طول Length هر مسیر هم تعداد یال‌هایی است که در طول مسیر قرار دارد یا تعداد راس‌ها منهای یک؛ به این مثال دقت کنید:

مسیر 1 -12-19-21 مسیری است که طول آن سه می‌باشد.

وزن مسیر هم از جمع وزن یال‌هایی که در طول مسیر طی می‌شود به دست می‌آید.

حلقه Loop مسیری است که راس اولیه با راس نهایی یکی باشد. نمونه‌ی آن مسیر 1-12-19-1 می‌باشد. ولی مسیر 1-7-21 حلقه‌ای تشکیل نمی‌دهد.

لبه‌ی حلقه ای Looping Edge لبه‌ای است که مبداء یا آغاز کننده‌ی آن با مقصد یا ارث برنده‌ی آن یکی باشد. یعنی به راسی وصل شود که از همان، آغاز شده است. مثل لبه‌ی متصل به راس 14.

یک کلاس گراف به چه مواردی نیاز دارد:

عملیات ایجاد گراف

افزودن و حذف یک راس یا لبه

بررسی اینکه بین دو راس لبه ای وجود دارد یا خیر

جست و جوی جانشین‌های یک راس

در قسمت آینده کد آن را در سی شارپ پیاده سازی خواهیم کرد.