به نام او

امتحان دوم درس طراحی الگوریتم (بهار ۱۴۰۱)

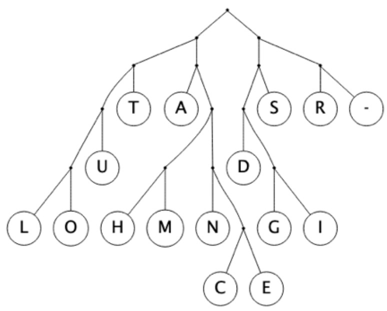
تاریخ امتحان: ۱۴۰۱/۲/۲۲ **مدت امتحان:** ۲ ساعت

توجه

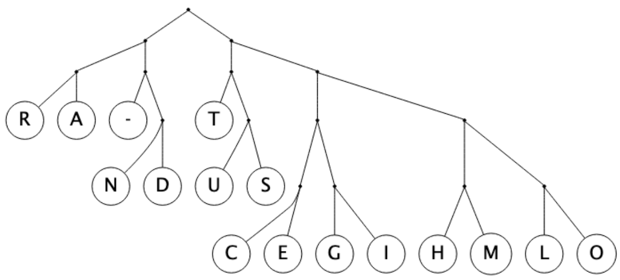
- در مدت امتحان وسایل هوشمند خود (نظیر گوشی همراه، ساعت هوشمند و لپتاپ) را خاموش کنید و آن‌ها را در کیف خود قرار دهید. در غیر اینصورت مشمول قوانین تقلب درس خواهید شد.
- لطفا فقط از حاصل تلاش خود برای حل سوالات استفاده کنید و هیچ‌گونه کمکی به دانشجویان دیگر نکنید.
- در ابتدای اولین صفحه‌ی پاسخ‌نامه، متن زیر را با خط خود نوشته و امضا نمایید:

۱- (۲۵ نمره) رشته‌ی DATA-STRUCTURES-AND-ALGORITHMS را در نظر بگیرید.

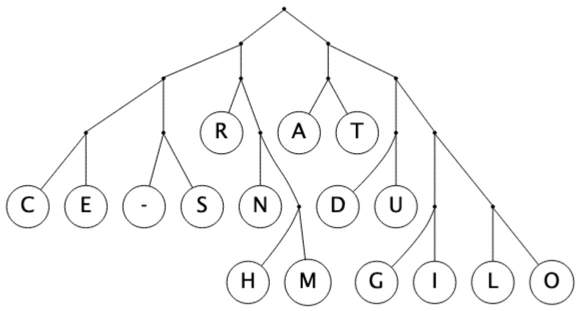
الف) کدام درخت (یا درختان) زیر مربوط به کد پیشوندی بهینه‌ی این رشته است؟ دلیل رد یا انتخاب هر گزینه را ذکر کنید.



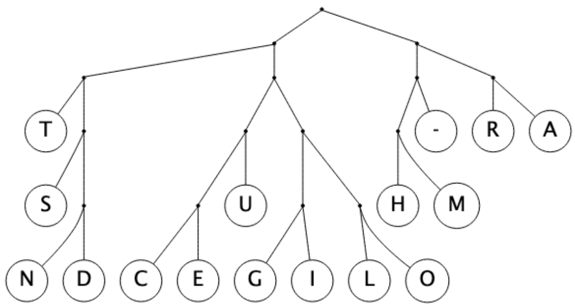
(۱)



(۲)



(۳)



(۴)

ب) میزان فشرده‌سازی (نسبت اندازه‌ی فایل فشرده شده به فایل اصلی) به کمک روش هافمن در مقایسه با ذخیره‌سازی به صورت متن چقدر است؟

نکته: هر بایت، ۸ بیت است.

پاسخ:

الف) (۲۰ نمره، ۵ نمره برای هر قسمت با دلیل، پاسخ بدون دلیل نمره‌ای ندارد) ابتدا جدول فراوانی حروف این رشته را تشکیل می‌دهیم و سپس طول کد هر حرف در هر درخت را بدست آورده و در انتها طول کل رشته کد شده را محاسبه می‌کنیم.

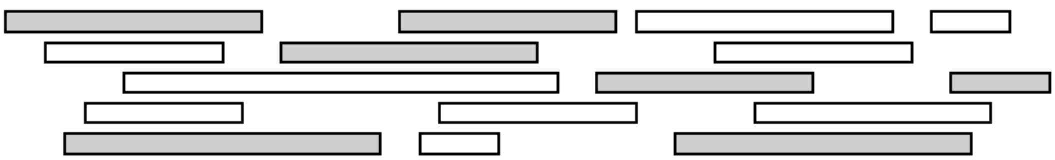
حرف	A	T	S	R	-	U	D	C	E	G	H	I	L	M	N	O	طول کل رشته کد شده
فراوانی	۴	۴	۳	۳	۳	۲	۲	۱	۱	۱	۱	۱	۱	۱	۱	۱	–
طول کد درخت ۱	۳	۳	۳	۳	۳	۴	۴	۶	۶	۵	۵	۵	۵	۵	۵	۵	۱۱۴
طول کد درخت ۲	۳	۳	۴	۳	۳	۴	۴	۵	۵	۵	۵	۵	۵	۵	۴	۵	۱۱۴
طول کد درخت ۳	۳	۳	۴	۳	۴	۴	۴	۴	۴	۵	۵	۵	۵	۵	۴	۵	۱۱۵
طول کد درخت ۴	۳	۳	۴	۳	۳	۴	۵	۵	۵	۵	۴	۵	۵	۴	۵	۵	۱۱۵

اگر درخت هافمن را رسم می‌کردیم به درختی مشابه درخت ۱ یا ۲ می‌رسیدیم که در آن طول کل رشته کد شده ۱۱۴ است. پس درخت‌های ۱ و ۲ درخت‌های پیشوندی بهینه هستند.

ب) (۵ نمره) $\frac{۱۱۴}{۳۰\times۸} = ۰/۴۷۵$

۲– (۲۵ نمره) فرض کنید n بازه متفاوت روی محور اعداد حقیقی به شما داده شده است. هر بازه i با دوتایی (s_i, f_i) که نشان‌دهنده‌ی نقاط شروع و پایان بازه است، مشخص می‌شود. الگوریتمی بهینه بدست آورید بطوری که با انتخاب کمترین بازه از بازه‌های داده شده، همه‌ی آن‌ها را پوشش دهد. بهینگی الگوریتم خود را ثابت کنید. شبه کد آن را نوشته و تحلیل زمان اجرا و حافظه‌ی آن‌را بدست آورید.

مثال: با انتخاب ۷ بازه از بازه‌های داده شده در زیر که به رنگ خاکستری در آمده‌اند، می‌توان همه‌ی بازه‌ها را پوشش داد.



پاسخ (الگوریتم ۸نمره، اثبات ۷ نمره، شبه کد ۶ نمره، تحلیل زمان اجرا ۲ نمره، تحلیل حافظه ۲ نمره):

این سوال شبیه سوال ۴ تمرین کتبی است. به کمک روش حریصانه مقابل می‌توان با کمترین تعداد بازه، همه‌ی بازه‌ها را پوشش داد. ابتدا بازه‌ها را بر اساس زمان شروعشان به صورت صعودی مرتب می‌کنیم. اگر زمان شروع دو بازه یکسان بود، بازه‌ها را بر اساس طولشان به صورت نزولی مرتب می‌کنیم و نتیجه را در لیستی ذخیره می‌کنیم. اولین بازه را انتخاب می‌کنیم و همه بازه‌هایی که توسط آن به طور کامل پوشش داده شده‌اند را حذف می‌کنیم. از بین بازه‌های با همپوشانی با بازه‌ی فعلی بازه‌ای را انتخاب می‌کنیم که زمان پایان بزرگتری دارد. اگر بازه‌ای با همپوشانی موجود نبود، بازه‌ی بعدی از لیست را انتخاب می‌کنیم و این کار را ادامه می‌دهیم تا بازه‌ای در لیست باقی نماند.

اثبات بهینگی: با توجه به نحوه‌ی ساختن پاسخ، جواب حریصانه همیشه همه‌ی بازه‌ها را پوشش می‌دهد. حال باید ثابت کنیم این کار را به کمک کمترین تعداد بازه‌ی ممکن انجام می‌دهد. فرض کنیم اینگونه نباشد. شبیه‌ترین پاسخ بهینه به پاسخ حریصانه را انتخاب می‌کنیم. اگر بازه‌های انتخاب شده را به ترتیب صعودی زمان شروعشان (و در صورت مساوی بودن زمان شروع، بر ترتیب صعودی طولشان) مرتب کنیم، این پاسخ بهینه بیشترین بازه مشابه را پاسخ حریصانه دارد. این دو پاسخ را می‌توانیم به صورت زیر بنویسیم:

$S_{greedy}: I_1, I_2, \dots, I_k, \dots$

$S_{optimal}: I_1', I_2', \dots, I_k', \dots$

فرض کنیم این دو پاسخ، برای تمام k-1 بازه‌ی اول مشابه باشند و اولین تفاوت در بازه‌ی kام اتفاق بیفتد. به عبارتی $I_i = I_i', i < k$ و $I_k \neq I_k'$. در این حالت اگر زمان شروع هر دو بازه kام برابر باشد، با توجه به نحوه‌ی ساخت پاسخ حریصانه، طول بازه‌ی حریصانه بلندتر است پس $f_k > f_k'$. در این حالت می‌توان بازه‌ی I_k' را از پاسخ بهینه حذف کرد و بازه‌ی I_k را به آن اضافه کرد. پاسخ همچنان بهینه خواهد بود چون پوشش بازه‌ها کاهش نیافته است و جواب حاصل به پاسخ حریصانه شبیه‌تر خواهد بود که با فرض خلف در تناقض است. به طور مشابه اگر زمان شروع I_k و I_k' متفاوت باشد، با توجه به نحوه انتخاب حریصانه، $f_k \geq f_k'$. پس می‌توان بازه I_k' را حذف کرد و I_k را به پاسخ بهینه اضافه کرد، پاسخ هنوز بهینه باقی بماند (با توجه به نحوه‌ی انتخاب بازه، پوشش کاهش نمیابد) و جواب حاصل به پاسخ حریصانه شبیه‌تر خواهد بود که این نیز با فرض خلف در تناقض است. در نتیجه، پس پاسخ حریصانه بهینه می‌باشد.

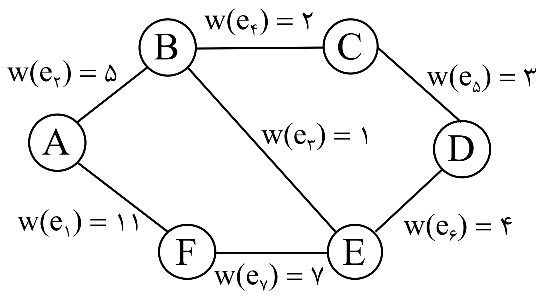
```
CoverIntervals(I) {
    n = s.length
    // Sort by start time; break ties based on interval length
    I_sorted = sort(I, key=lambda i: (i[0], i[1] - i[0])
    latest = null
    A = []
    for i = 0 to n - 1 {
        A += [I_sorted[i]]
        for j = i + 1 to n - 1 {
            if I_sorted[j][0] > I_sorted[i][1] {
                i = j - 1
                latest = null
                break
            }
            if I_sorted[j][1] > I_sorted[i][1] AND latest == null
                latest = I_sorted[j]
                A += [I_sorted[j]]
            } else if I_sorted[j][1] > I_sorted[i][1] AND I_sorted[j][1] > latest[1] {
                latest = I_sorted[j]
```

```
                // Update the last interval
                A[-1] = I_sorted[j]
            }
        }
    }
    return A
}
```

پیچیدگی زمان اجرا $O(n\ log\ n)$ می‌باشد و پیچیدگی حافظه، $O(n)$ است.

۳- (~~۲۰ نمره~~ ۵۰ نمره) الف) فرض کنید حین یافتن کوتاه‌ترین مسیر با روش دایکسترا یا بلمن-فورد در یک گراف، در یکی از مراحل میانی، dist[v]=۱۶ و dist[w]=۲۵. از طرفی می‌دانیم یال $e = v \to w$ وزن ۷ دارد. پس از فراخوانی $\mathrm{Relax}(e)$ مقداری (dist[v], dist[w]) چقدر خواهد بود؟

برای قسمت‌های (ب)، (ج) و (د) گراف مقابل را در نظر بگیرید. (قسمت الف ربطی به این گراف ندارد.)



ب) در گراف بالا، فرض کنید درخت پوشای بهینه را توسط الگوریتم کروسکال پیدا می‌کنید. ترتیب اضافه شدن یال‌ها به درخت پوشای بهینه را بدست آورید.

ج) در گراف بالا، اگر الگوریتم دایکسترا از راس A شروع شود، رئوس به چه ترتیبی از صف اولویت‌دار (priority queue) حذف می‌شوند؟

د) فرض کنید گراف بالا به یک گراف جهت‌دار تبدیل شده است به طوری که یال‌های جهت‌دار در هر دو جهت یال بدون جهت ایجاد شده‌اند. به عنوان مثال یک یال با وزن ۷ از راس $E \to F$ و یک یال با همین وزن از $F \to E$ ایجاد می‌شود. اگر الگوریتم بلمن-فورد را از راس A با ریلکس کردن یال‌های $e_{۱}, e_{۲}, e_{۳}, e_{۴}, e_{۵}, e_{۶}, e_{۷}$ (از چپ به راست) به ترتیب اجرا کنیم، dist[F] چند بار تغییر خواهد کرد؟ در هر بار تغییر، مقادیر آن (پیش و پس از تغییر) چقدر خواهد بود؟

پاسخ:

الف) (۱۰ نمره) (۱۶, ۲۳)

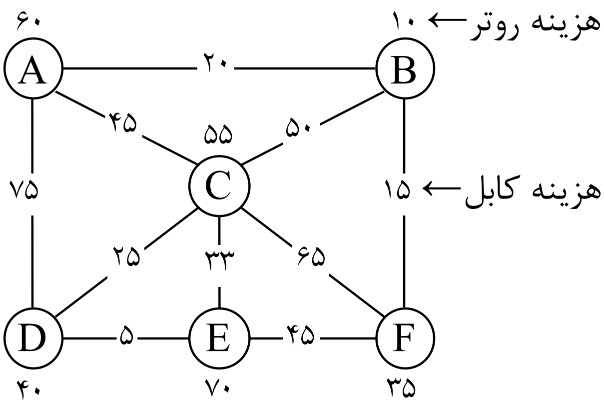
ب) (۱۵ نمره) پاسخ از چپ به راست برابر $e_{۳}, e_{۴}, e_{۵}, e_{۲}, e_{۷}$ است.

ج) (۱۵ نمره) A, B, E, C, D, F

د) (۱۰ نمره) یک بار تغییر می‌کند. میزان تغییر از بی‌نهایت به ۱۱ است.

۴- (~~۳۰ نمره~~ ۰ نمره) بعد از حضوری شدن دانشگاه، دانشجویان خوابگاهی از کیفیت پایین اینترنت ناراضی هستند. پس از رایزنی‌های مکرر، دانشگاه حاضر شده برای افزایش کیفیت اینترنت، در هر اتاق یک روتر (router) نصب کرده یا اتاق را با کابل شبکه به اتاق دیگری که در آن روتر نصب شده متصل کند (اتصال کابل بین دو اتاق بدون روتر بی‌فایده است). هزینه نصب روتر در اتاق iام برابر r_i و هزینه نصب کابل از اتاق i به اتاق j برابر $c_{ij} = c_{ji}$ است. با توجه به محدودیت بودجه، مسئول خوابگاه از شما خواسته به او کمک کنید تا با کمترین هزینه، اینترنت با کیفیت را به همه‌ی اتاق‌ها برسانید. فرض کنید خوابگاه n اتاق دارد و هزینه‌ی نصب روترها و کابل‌ها بین هر اتاق به شما داده شده است. الگوریتم بهینه‌ای بدست آورید و پیچیدگی زمان اجرا و حافظه‌ی آن‌را پیدا کنید. نیازی به نوشتن شبه کد نیست.

مثال: در شکل روبرو خوابگاه ۶ اتاق دارد. اگر یالی بین دو راس وجود نداشته باشد، نصب کابل بین آن دو راس غیر ممکن (با هزینه بی‌نهایت) خواهد بود. پاسخ بهینه در این حالت نصب روتر در اتاق‌های B و D (با هزینه ۱۰+۴۰) و کشیدن کابل بین اتاق‌های A-B، B-F، C-D و D-E (با هزینه ۲۰+۱۵+۲۵+۵) است.



پاسخ:

اگر شرط « اتصال کابل بین دو اتاق بدون روتر بی‌فایده است» را حذف کنیم و فرض کنیم برای اتصال هر اتاق به اینترنت کافی است روتری در آن نصب شده یا با کابل مسیری به اتاقی با روتر وجود داشته باشد، مسئله با راه حلی که در ادامه می‌آید، در زمان چندجمله‌ای قابل حل است. در حالت فعلی، مسئله معادل مسئله‌ی درخت پوشای کمینه دو گام (2-hop minimum spanning tree) است، که جزء مسائل NP-hard به حساب می‌آید (رجوع کنید به [*]) و بهترین راه حل شناخته شده برای آن غیرچندجمله‌ای است. در مورد مسائل NP-hard در آخرین بخش درس بیشتر می‌خوانیم. به دلیل سختی ناخواسته، این سوال حذف و نمره‌ی سوال ۳ از ۵۰ محاسبه خواهد شد.

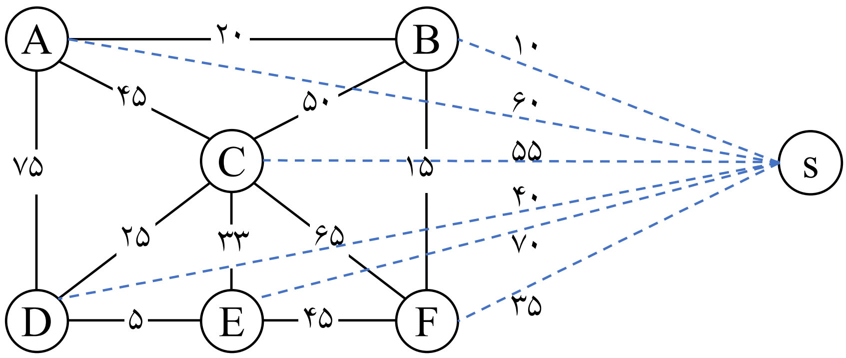
[*] E. Althaus, et al. "Approximating k-hop minimum-spanning trees," Operations Research Letters 33.2 (2005): 115-120.

یک گراف وزن‌دار با n + ۱ راس می‌سازیم. هر راس نشان‌دهنده‌ی یک اتاق است به علاوه یک راس s به گراف اضافه می‌کنیم. یال‌ها را به صورت زیر به گراف اضافه می‌کنیم.

- بین s و هر راس، یک یال با وزن r_i اضافه می‌کنیم.
- بین هر دو راس i و j، در صورتی که c_{ij} یک یال با وزن c_{ij} اضافه می‌کنیم. دقت کنید که یال‌ها جهت‌دار نیستند.

پاسخ مسئله، MST در این گراف است که می‌توان به کمک کروسکال آن را بدست آورد. در MST، یال‌هایی که از s به راس i می‌روند، نشان‌دهنده‌ی روتری است که در اتاق i نصب می‌شود. یال i به j نیز نشان‌دهنده‌ی نصب یک کابل بین اتاق i و j است.

در شکل زیر، ساخت چنین گرافی را برای مثال داده شده می‌بینید.



مشابه الگوریتم کروسکال، زمان اجرا برابر $O(|E|log|V|)$ و میزان حافظه مصرفی برابر $O(|V| + |E|)$ است. دقت کنید که زمان ساخت گراف $O(|V| + |E|)$ است.