به نام خدا

دانشگاه تهران، دانشکده مهندسی برق و کامپیوتر
تحلیل و طراحی الگوریتم‌ها

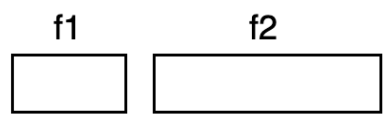
پاسخ تمرین کتبی سوم

---

۱. (آ) [۵ نمره]

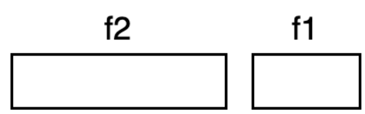
$f_1 : b_1 = 1, p_1 = 0.1$
$f_2 : b_2 = 2, p_2 = 0.9$

راه حل حریصانه:



$1 \times 0.1 + (1 + 2) \times 0.9 = 2.8$

راه حل بهینه:

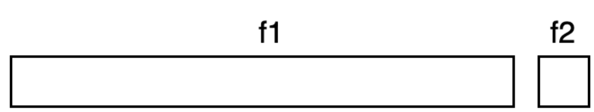


$2 \times 0.9 + (2 + 1) \times 0.1 = 2.1$

(ب) [۵ نمره] برای این قسمت در صورت سوال اشتباهی وجود داشته و کلمه‌ی صعودی در اصل نزولی بوده‌است. نمره‌ی سوال به کسانی که سوال را حل کرده‌اند تعلق می‌گیرد اما در پایین راه حل برای صورت سوال تصحیح شده آمده است.

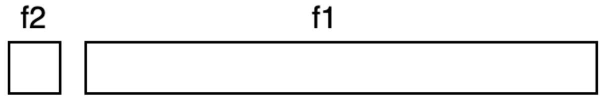
$f_1 : b_1 = 10, p_1 = 0.6$
$f_2 : b_2 = 1, p_2 = 0.4$

راه حل حریصانه:



$10 \times 0.6 + (10 + 1) \times 0.4 = 10.4$

راه حل بهینه:



$1 \times 0.4 + (1 + 10) \times 0.6 = 7.0$

(ج) [۱۵ نمره] فایل ها را بر حسب $\frac{b_i}{p_i}$ آن‌ها به‌صورت صعودی مرتب می‌کنیم. همانطور که واضح است هزینه‌ی اجرای این الگوریتم $O(n \log n)$ است.
فرض کنید ترتیب به دست‌آمده از الگوریتم ما به این‌صورت باشد:

$\{\ldots, f_m, \ldots\}$

در نظر می‌گیریم که جواب بهینه تا انتخاب f_m با جواب حریصانه مشترک است اما در اینجا با هم تفاوت می‌کنند و به صورت زیر است:

$\{\ldots, f_m, \ldots, f_j, f_i, \ldots\}$

چون الگوریتم بهینه از جایی به بعد متفاوت از راه حل حریصانه عمل کرده، پس می دانیم که در یک انتخاب روند صعودی آن قطع شده است که آن را انتخاب f_i در نظر می‌گیریم . می‌دانیم:

$\frac{b_i}{p_i} < \frac{b_j}{p_j}$

حال محسابه می‌کنیم که اگر در جواب بهینه جای فایل j و i را عوض کنیم، نتیجه چه تغییری می‌کند:
هزینه‌ی جواب بهینه نسبت به جواب جدید به این صورت است که فایل f_i ، b_j واحد جلو رفته که پس هزینه را به اندازه‌ی $p_i \times (b_j)$ افزایش داده‌است. همچنین فایل f_j به اندازه‌ی $p_j \times (b_i)$ هزینه را نسبت به حالت قبل کاهش می‌دهد. در مجموع هزینه به اندازه‌ی $p_i \times b_j - p_j \times b_i$ تغییر کرده است که:

$b_i \times p_j - p_i \times b_j < 0$

پس هزینه نسبت به حالت بهینه کمتر شد این تناقض اثبات می‌کند که الگوریتم حریصانه، الگوریتم بهینه است.

۲. [۱۵ نمره] سوال را برعکس حل می‌کنیم. یعنی از خود عدد n شروع می‌کنیم و سعی می‌کنیم با دو عملیات زیر به عدد ۱ برسیم:

- کم کردن ۱ واحد از عدد
- تقسیم کردن عدد بر عدد ۲

واضح است که هر سری از حرکاتی که با ضرب و به علاوه انجام می شود را اگر بر عکس کنیم با منها و تقسیم انجام می شود و تناظر یک به یک با یکدیگر دارند. حال با شروع از عدد اگر فرد بود ۱ واحد از آن کم می‌کنیم و اگر زوج بود آن را بر ۲ تقسیم می‌کنیم و این روند را ادامه می‌دهیم تا به ۱ برسیم.
اثبات :
از برهان خلف استفاده می کنیم. فرض می کنیم جواب بهینه ای وجود داشته باشد که با مراحل کم تری به ۱ برسد. . اگر اعداد به دست آمده در هر مرحله جواب حریصانه به صورت زیر باشد

$\{x_1, x_2, \ldots, x_i, \ldots, x_n\}$

دنباله زیر را به عنوان جواب بهینه در نظر می گیریم:

$\{o_1, o_2, \ldots, o_i, \ldots, o_m\}$

به طوری که تا مرحله‌ی i ام هر دو انتخاب باهم برابرند و از این مرحله متفاوت می‌شوند.در هر مرحله اگر عدد فرد باشد، هر دو الگوریتم انتخاب مشابهی دارند. پس تفاوت آن‌ها در عددی زوج رخ می‌دهد. یعنی :

$x_{i-1} = o_{i-1} = 2k$

پس از رسیدن به عدد x_{i-1} الگوریتم حریصانه عدد را تقسیم بر ۲ کرده و جواب بهینه یک واحد از عدد کم می‌کند پس: $x_i \leq o_i$
اولین اندیسی را که در آن جواب بهینه به عددی کوچکتر یا مساوی x_i می‌رسد را k در نظر می‌گیریم. می‌دانیم:

$o_k \leq x_i < o_{k-1} \to o_k = x_i - a, 0 \leq a$

دنباله‌ی $\{o_1, o_2, \ldots, o_i, \ldots, o_k, \ldots, o_m\}$ را با دنباله‌ی $\{o_1, o_2, \ldots, x_i, x_i - 1, \ldots, x_i - a, \ldots, o_m\}$ جایگزین می‌کنیم.
که قطعا تعداد اعضایش از تعداد اعضای دنباله‌ی قبلی کمتر است (زیرا جواب بهینه در این بین فقط حداکثر یک تقسیم بر ۲ می تواند داشته باشد و در واقع جواب بهینه از تعدادی منهای یک و سپس تقسیم بر ۲ و جواب حریصانه از یک تقسیم بر ۲ و سپس تعدادی منهای یک تشکیل شده که طول کمتری دارد پس اگر این تکه را جایگزین کنیم به طول کمتری می رسیم) پس به جوابی با طول کمترین رسیدیم که خلاف فرض کم طولترین بودن جواب بهینه است. پس جواب بهینه همان جواب حریصانه است.

۳. [۱۰ نمره] ابتدا طناب‌هارا بر حسب طولشان در یک min heap قرار می‌دهیم که هزینه‌ی این کار $O(n \log n)$ است. البته ساخت heap را با هزینه‌ی $O(n)$ هم می‌توان انجام داد.
حال دو تا کوتاه‌ترین طناب‌ها را برمی‌داریم که هزینه‌ی آن برابر با دو بار delete min است. سپس طنابی با طولی برابر با مجموع طول این دو طناب در heap ، insert می‌کنیم. و این کار را تا زمانی که یک طناب واحد داشته باشیم ادامه می‌دهیم.

۴.
- [۵ نمره] در این روز تصمیم اشتباه است.
مثال نقض:
اگر فردی در حساب خود ۳۰۰۰۰ تومان پول داشته باشد طبق الگوریتم مطرح شده به او یک اسکناس ۲۵۰۰۰ تومانی و ۵ اسکناس ۱۰۰۰ تومانی می‌دهیم. که در مجموع ۶ اسکناس است. در صورتی که با پرداخت ۳ اسکناس ۱۰۰۰۰ تومانی می‌توان تنها با مجموع ۳ اسکناس این پول را نقد کرد.
- [۱۰ نمره] در این روز تصمیم درست است.
اثبات درستی الگوریتم حریصانه:
فرض می‌کنیم راه حل بهینه‌ای وجود دارد که در انتخاب اسکناس‌ها تا جایی با الگوریتم حریصانه یکسان است اما پس از آن انتخاب متفاوتی دارد هر دو جواب را به صورت نزولی مرتب می‌کنیم. سپس اولین جایی که اسکناس انتخابی توسط دو الگوریتم با هم متفاوت است را در نظر می‌گیریم. . مقدار باقی‌مانده از کل پول در این مرحله m است.
روی مقادیر m حالت بندی می کنیم. درون هر بازه با توجه به تفاوت انتخاب حریصانه و انتخاب بهینه می دانیم که انتخاب حریصانه اسکناس با مقدار بیشتر را انتخاب می کند. پس ثابت می کنیم تا وقتی از این بازه به بازه های قبل برگردیم الگوریتم حریصانه ما اکیدا اسکناس کم تری انتخاب کرده است. پس در هر بازه بهترین انتخاب حریصانه است و در نتیجه به طور کلی به خاطر همان برتری که در انتخاب اول در نخستین بازه داشته است در کل هم اسکناس کم تری دارد
  - $m < 5000$: در این حالت انتخابی جز اسکناس ۱۰۰۰ تومانی وجود ندارد پس هردو الگوریتم مانند هم عمل می‌کنند.
  - $5000 \leq m < 10000$: در این حالت الگوریتم حریصانه اسکناس ۵۰۰۰ تومانی را انتخاب می‌کند، الگوریتم بهینه انتخاب متفاوتی دارد و اسکناس ۱۰۰۰ تومانی انتخاب می‌کند. در این حالت حتما انتخاب حریصانه ۴ اسکناس کمتر استفاده می‌کند. پس به تناقض می‌رسیم و راه حل حریصانه همانند راه حل بهینه است.
  - $10000 \leq m < 20000$: در این حالت الگوریتم حریصانه با انتخاب اسکناس ۱۰۰۰۰ تومانی مقدار m را به عددی در بازه‌ی قبل کاهش می‌دهد. اما راه حل بهینه تنها ۵۰۰۰ تومانی یا ۱۰۰۰ تومانی برمی‌دارد پس با انتخاب تعداد اسکناس بیشتر نسبت به حریصانه مقدار m را به بازه‌ی قبل کاهش می‌دهد.
  - $20000 \leq m < 25000$:
در این حالت الگوریتم حریصانه با انتخاب دواسکناس ۱۰۰۰۰ تومانی مقدار m را به عددی در بازه‌ی قبل کاهش می‌دهد. اما راه حل بهینه تنها ۵۰۰۰ تومانی یا ۱۰۰۰ تومانی برمی‌دارد پس با انتخاب تعداد اسکناس بیشتر نسبت به حریصانه مقدار m را به بازه‌ی قبل کاهش می‌دهد .
  - $25000 \leq m$:
در این حالت الگوریتم حریصانه تا جای که می‌تواند با انتخاب اسکناس ۲۵۰۰۰ تومانی مقدار m را به عددی در بازه‌های قبل کاهش می‌دهد. اما راه حل بهینه تنها ۵۰۰۰ تومانی یا ۱۰۰۰ تومانی یا ۱۰۰۰۰ تومانی برمی‌دارد پس با انتخاب تعداد اسکناس بیشتر نسبت به حریصانه مقدار m را به بازه‌های قبل کاهش می‌دهد .

۵. [۱۰ نمره] دو اشاره‌گر یکی به اولین خانه‌ای که در آن پلیس قرار دارد و یکی به اولین خانه‌ای که در آن دزد قرار دارد در نظر می‌گیریم. اگر دزد فعلی در فاصله‌ی کمتر از k از پلیس بود پلیس دزد را دستگیر. می‌کند و اشاره‌گر ها را به اولین دزد و پلیس بعدی به روزرسانی می‌کنیم. اگر فاصله‌ی دزد و پلیس بیشتر از k بود اشاره‌گر کوچکتر را به روزرسانی می‌کنیم تا به دزد یا پلیس بعدی اشاره کند و این کار را تکرار می‌کنیم. چون هرکدام از اشاره‌گر ها یک بار طول آرایه را طی می‌کنند هزینه‌ی زمانی این الگوریتم به صورت سرشکن از اردر $O(n)$ است.

۶. (آ) [۱۰ نمره]

به صورت حریصانه هنگامی که به پمپ بنزین شماره ی i رسیدیم، اگر به‌اندازه‌ی رسیدن به پمپ بنزین شماره‌ی $i+1$ بنزین در باک داشتیم در این پمپ نمی ایستیم و تا پمپ بعدی میرویم. اما اگر مقدار بنزین در باکمان کافی نبود به ناچار در پمپ i ام ایستاده، باک را پر میکنیم. برای اثبات اینکه این روش بهینه است به این شکل عمل میکنیم: پاسخی که الگورتیم خروجی میدهد را S بگیرید و فرض کنید بهینه نباشد، از بین پاسخهای بهینه، نزدیکترین پاسخ به S را برمیگزینیم و آن را S' می نامیم (نزدیکی دو پاسخ را بزرگترین i ی میگیریم که هر دو پاسخ تا رسیدن به پمپ بنزین i ام، در ایستادن یا نایستادن، همانند هم رفتار کرده باشند)

فرض کنید S و S' تا پمپ $k-1$ ام همانند هم رفتار کرده اند و در پمپ k ام رفتار گوناگونی دارند در این صورت باید S در پمپ k ام توقف نکرده باشد اما S' توقف کرده باشد. در این صورت پاسخی مانند S'' را در نظر بگیرید که در پمپ k ام نمی ایستد و در پمپ $k+1$ سوخت گیری می کند از آنجا به بعد نیز مانند پاسخ بهینه رفتار میکند. شمار ایستادن های S'' با S' برابر است اما به S نزدیکتر است و این خلاف فرض ما بود

(ب) [۱۵ نمره] اینجا نیز یک روش حریصانه ارائه میدهیم. فرض کنید در پمپ بنزین i ام ایستاده ایم، دقیقا به میزانی بنزین میزنیم که به همراه بنزین باقی مانده در باک به نخستین پمپ بنزینی برسیم که قیمت بنزین آن ارزانتر است و به آن پمپ می رویم . اگر با یک باک پر به چنین پمپ بنزینی نمیرسیم، باک را پر میکنیم و به پمپ بنزین بعدی میرویم. در آنجا همینکار را تکرار میکنیم. برای اثبات اینکه این روش میزان پول مصرفی را کمینه میکند به این شکل عمل میکنیم: پاسخی که الگورتیم میدهد را S بگیرید و فرض کنید S بهینه نباشد، از بین پاسخهای بهینه، نزدیکترین به S را برمیگزینیم و آن را S' مینامیم. نخستین جایی که در نظر بگیرید که S مانند S' رفتار نمیکند. دو حالت ممکن است رخ دهد:

- اگر S در حالتی باشد که باک را کامل پر کرده چون نمی توانسته با یک باک پر به یک پمپ بنزین با قیمت پایینتر برسد. در این صورت S' میزان کمتری بنزین زده است. چون با این میزان بنزین قطعا مجبور به بنزین زدن در جای گرانتر است S' را میتوان با پر کردن باک در این مرحله بهتر کرد که تناقض است.
- اگر S در حالتی باشد که دقیقا به میزانی بنزین زده شده که تا نخستین پمپ بنزین ارزان تر برسیم: اگر S' میزان کمتری بنزین زده باشد، قطعا در یک پمپ بنزین با قیمت بالاتر توقف کرده و بنزین زده که میتوان با جایگزینی S' را بهتر کرد. اگر S' میزان بیشتری بنزین زده باشد، میزان اضافه را میتوان در پمپ بنزین با قیمت پایینتر زد و این هم تناقض است

برای پیاده سازی الگورتیم باید برای هر پمپ بنزین محاسبه کنیم که نخستین پمپ بنزین کم قیمت تر پس از آن کدام است و چقدر تا آنجا راه است. این محاسبه هم باید از $O(n)$ باشد. برای این کار میتوان از یک پشته بهره گرفت: از پمپ ۱ام شروع میکنیم و آن را در پشته میگذاریم، سپس در هر مرحله پیش از گذاشتن پمپ i ام در پشته، تا جایی که قیمت پمپ i ام از قیمت پمپ سر پشته پایینتر است، سر پشته را برمیداریم. برای هر پمپی که از سر پشته برداشته می شود میتوان مقدار نخستین کم قیمت ترین پمپ و فاصله ی آن را حساب کرد (که برابر پمپی است که مایه ی برداشتن آن از پشته شده) پس از برداشتن پمپ های گفته شده، پمپ i ام در سر پشته گذاشته میشود. بدین ترتیب در پشته همیشه پمپ ها به ترتیب قیمت افزایشی چیده شده اند که درستی مقدار محاسبه شده را تضمین میکند.