آیا توابع زیر برای روش $Open\ Addressing$ مناسب هستند؟ به ازای k و m های مختلف آن را بررسی کنید و علت نامناسب بودن آن‌ها را توضیح دهید. (سایز آرایه برابر با m است.)

$$H(k,i) = (k\ mod\ m + 3i\ mod\ m)\ mod\ m \quad , \quad m = 2t+1$$
$$H(k,i) = (k^2 + 2k\ +\ i(k\ mod\ m))\ mod\ m$$

آ) اگر $m = 3t$ باشد این تابع قبول نیست. چون همیشه $3i \equiv 3x \pmod{3t}$ و تمام خانه ها را پوشش نمی‌دهد، اما در غیر این صورت تابع قابل قبول خواهد بود.

ب) به دلیل این که ضریب i در مواردی می‌تواند صفر باشد، $k = tm$ پس این تابع مناسب نیست.

آرایه‌ای به طول n داریم. الگوریتمی ارائه دهید که عنصری که بیش از $n/2$ بار در آرایه تکرار شده است را با $O(n)$ پیدا کند.

مانند $quick\ sort$ عمل می‌کنیم و در هر مرحله یک $pivot$ برای آرایه در نظر می‌گیریم.
می‌دانیم که با $O(n)$ می‌توان عناصر آرایه را به دو دسته‌ی بزرگ‌تر از $pivot$ و کوچک‌تر مساوی $pivot$ تقسیم کرد.(چرا؟) حال در هر مرحله تنها کافی است که همین عمل را به صورت بازگشتی بر روی دسته‌ی بزرگ‌تر انجام دهیم، از آن جا که جواب بیش از $n/2$ بار دیده شده است پس حتما در دسته بزرگ‌تر است. شرط اتمام هم زمانی است که $pivot$ دسته‌ی k تایی را به دو دسته‌ی 0 و k تایی تقسیم کند به طوری که $n/2 < k$ باشد.

دو آرایه مرتب شده به طول n و m داریم. می‌خواهیم میانه‌ی تمام $n+m$ عدد را پیدا کنیم.

آ) الگوریتمی با زمان $log(n)*log(m)$ برای این کار ارائه دهید.

در واق می‌خواهیم دو عنصر n_i از آرایه‌ی n و m_j از آرایه‌ی m را به گونه‌ای پیدا کنیم که $i+j \;\; = (\frac{n+m}{2})$ شود.(چرا؟) برای این کار به این صورت عمل می‌کنیم که در ابتدا یک عنصر دلخواه از آرایه‌ی n انتخاب می‌کنیم (مثلا n_i) بعد بر روی آرایه‌ی m باینری سرچ می‌زنیم تا نزدیک ترین عدد به n_i را پیدا کنیم و آن را m_j می‌نامیم. حال سه حالت ممکن است:

۱) $i+j \;\; > (\frac{n+m}{2})$: در این حالت به ازای عناصر کوچکتر از n_i باید همین روند را ادامه دهیم.

۲) $i+j \;\; < (\frac{n+m}{2})$: در این حالت به ازای عناصر بزرگتر از n_i باید همین روند را ادامه دهیم.

۳) $i+j \;\; = (\frac{n+m}{2})$: جواب مساله.

در واقع ما دو تا باینری سرچ تو در تو بر روی آرایه‌ها می‌زنیم. پس زمان آن برابر است با $log(n)*log(m)$

ب) الگوریتمی با زمان $log(n)+log(m)$ برای این کار ارائه دهید.

مجدد دنبال همان اعضا هستیم، اما نیازی نیست به ازای هر عدد i که از آرایه ی اول پیدا می کنیم زمان $logm$ مصرف کنیم و دو اندیس i و j را همزمان حرکت می دهیم تا هر کدام در مجموع به ترتیب زمان های $logn$ و $logm$ طول بکشند. نکته ی این قسمت این است که اگر $i+j$ کمتر از $(n+m)/2$ باشد هیچ i' کوچکتری از i هم نمیتواند j' معادلی پیدا کند که $i'+j'$ برابر $(m+n)/2$ باشد. همینطور برای j هم چنین چیزی داریم. در نتیجه یکی درمیان یکی از آنها را رو به جلو یا عقب در آرایه ی خود حرکت می دهیم تا به جایی برسند که دیگر حرکت نکنیم (یعنی نزدیک ترین اعداد به هم باشند) و شرط مورد نظر نیز برقرار باشد.

پیروز باشید