ساختمان داده‌ها و الگوریتم‌ها

نیم‌سال دوم ۹۷ـ۹۸

دانشکده‌ی مهندسی کامپیوتر

گردآورندگان: حسین ابراهیمی، علیرضا موسوی

پاسخنامه تمرین عملی اول

مسئله‌ی ۱. صف انتخاب واحد

لیست Sum را به این شکل تعریف می‌کنیم که $Sum[i]$ برابر است بامجموع شماره اعضای دیده شده پس از i بار عملیات .In در این صورت با هر عملیات In جدید لیست Sum در زمان ثابت به‌روز می‌شود و جمع هر زیر بازه دلخواه از i تا j نیز در زمان ثابت با کمک $Sum[j] - Sum[i]$ به دست می‌آید. حال از n اشاره‌گر استفاده می‌کنیم که اشاره‌گر jام مشخص می‌کند صف دانشکده j تا کجا پیش رفته است (در ابتدا همه اشاره‌گر ها مقدار -۱ دارند). به این ترتیب با هربار اجرای دستور Out کافیست اشاره‌گر صف مربوطه را جابه‌جا کنیم و با کمک لیست Sum پاسخ را چاپ کنیم. به این ترتیب هر دستور In و Out در زمان ثابت انجام می‌شود و مقداردهی اولیه اشاره‌گر ها نیز به $O(n)$ عملیات نیاز دارد پس به طور کلی الگوریتم در زمان $O(n+q)$ اجرا می‌شود.

مسئله‌ی ۲. بازه‌های دوست داشتنی

ابتدا به کمک الگوریتم مرتب‌سازی شمارشی در زمان $O(n)$ بازه‌ها را یک‌بار برحسب x و یک‌بار برحسب yشان مرتب می‌کنیم. یک لیست به طول n بگیرید و این لیست را $count$ بنامید. $count[i]$ مشخص می‌کند که چند نقطه دقیقا در i بازه آمده‌اند. همچنین یک متغیر $size$ در نظر بگیرید که در ابتدا صفر است. روی نقاط ابتدایی و انتهایی بازه‌ها که مرتب شده‌اند حرکت می‌کنیم و هربار از یک نقطه به یک نقطه جدید رسیدیم تعداد نقاط بین را به $count[size]$ اضافه می‌کنیم و اگر نقطه جدیدی که به آن رسیدیم x بود $size$ را یکی زیاد می‌کنیم و در غیر این صورت یکی کم می‌کنیم و به این ترتیب با پیمایش خطی برروی نقاط مرتب شده مسئله در زمان $O(n)$ حل می‌شود.

مسئله‌ی ۳. دانشگاه شما دانشگاه نیست

یک لیست جدید به اسم c بگیرید که $c[i] = abs(a[i] - b[i])$. تعداد تکرار هرکدام از $c[i]$ها را در لیستی به نام $count$ نگه می‌داریم به این شکل که $count[c[i]]$ برابر است با تعداد تکرار $c[i]$. با توجه به اینکه حداکثر مقدار $count$ برابر است با $۱۰^۳$ (مستقل از ورودی است) مقداردهی اولیه آن در زمان $o(n)$ امکان پذیر است. حال کافیست یک اشاره‌گر داشته باشیم که آخرین درایه غیر صفر $count$ را مشخص کند. این اشاره‌گر را $last$ می‌نامیم. سپس اگر $count[last]$ بزرگتر از k بود از آن k تا کم می‌کنیم، به $count[last-۱]$ نیز kتا اضافه می‌کنیم و حل تمام است. در غیر این صورت از $count[last]$ را صفر می‌کنیم و به همان مقدار به $count[last-۱]$ اضافه می‌کنیم و به همان مقدار از تعداد مراحل کم می‌کنیم و مجدد حل را ادامه می‌دهیم. این الگوریتم در زمان $O(n+k)$ اجرا می‌شود.