آزمون درس: **الگوریتم های پیشرفته**

نام و نام خانوادگی دانشجو: **چنور حسینی** زمان آزمون : **48 ساعت**

شماره دانشجویی: **990182060** استاد درس : **سید علی ابراهیمی رضوی**

رشته و مقطع تحصیلی: **ارشد مهندسی نرم افزار**

سوالات میان ترم

1- روش حل سیستماتیک مسئله را الگوریتم می گویند. الگوریتم در واقع **مراحل قابل اجرا** برای حل یک مسئله است که باید ویژگی های زیر را داشته باشد:

الف) یک نقطه شروع و یک یا چند نقطه پایان دارد.

ب) این مراحل تکرار پذیر هستند به این معنی که میتوانیم یک الگوریتم را به دفعات اجرا کنیم.

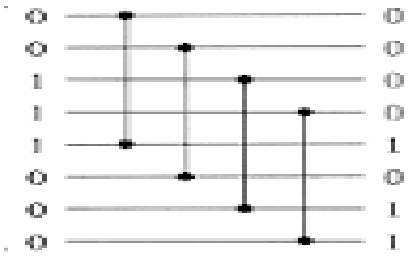
ج) زمان اجرای الگوریتم قابل قبول بوده و هزینه اجرای آن مناسب باشد.

قسمت دوم سوال اول:

در درس الگوریتم، موارد متفاوتی از مباحث الگوریتم ها مطالعه می شود ، مواردی مانند زمان اجرای الگوریتم ها، الگوریتم های مرتب سازی و ..... که در هر مبحثی روش خاصی بیان می شود.مهندسین نرم افزار می توانند از این روش ها و متد ها در برنامه های خودشان استفاده کنند. مثلا زمانیکه شخصی یک سیستم اتوماسیون اداری طراحی می کند با استفاده از مبحث زمان اجرا می تواند زمان تقریبی انجام عمل خاصی توسط سیستم طراحی شده اش را حدس بزند. به فرض اینکه در این سیستم هدف پیدا کردن یک نامه اداری است که برای این کار نیاز به جستجو دارد پس مهندس نرم افزار با ارزیابی داده های موجود در سیستم و مقایسه زمان اجرای الگوریتم های مختلف جستجو، می تواند کار آمد ترین روشی که در سیستم او قابل پیاده سازی است را انتخاب کند. در ضمن بسیاری از زبان های برنامه نویسی این الگوریتم ها را به صورت کتابخانه هایی در محیط کامپایلر خود دارند و در واقع می توانیم با خیال آسوده از این کدهای آماده در برنامه های خودمان استفاده کنیم و در مدت زمان کوتاه تری به نتیجه بهتر و مناسب تر برسیم.

تسلط بر مبحث تحلیل و طراحی الگوریتم ها و الگوریتم های پیشرفته به حل بهتر مسائل برنامه‌نویسی و نوشتن کدهایی با کارایی و سرعت اجرای بالاتر و مصرف حافظه کم‌تر، کمک زیادی می کند.

2- نیم پاک کننده در واقع یک شبکه مقایسه ای به عمق یک است یعنی تنها با انجام یک مرحله کار مقایسه ورودی های خودش را یکدست می کند ورودی های نیم پاک کننده را دنباله ای از صفر ویک ها در نظر می گیریم که بیتونیک (Bitonic) است پس از انجام مرحله اول نیم پاک کننده یکی از این دو حالت اتفاق می افتد: یا نیمه بالایی دنباله همه صفر می شوند ویا نیمه پایینی دنباله همه یک می شوند که به آنها پاک(Clean) می گویند و نصف دیگر دنباله نیز Bitonic خواهد بود. مثلا دنباله a={0,0,1,1,1,0,0,0} را در نظر بگیرید که Bitonic است و با ورود به نیم پاک کننده مطابق شکل زیر به دنباله b={0,0,0,0,1,0,1,1} تبدیل می شود.



(a1 را با a5 مقایسه کرده و صفر و یک را تغییر نمی دهد چون عدد کوچکتر در جای درست خود قرار دارد ، a2 را با a6 مقایسه می کند، a3 را با a7 مقایسه کرده و محل صفر و یک را تغییر می دهد و در نهایت a4 را با a8 مقایسه می کند و از سمت بالا به سمت پایین دنباله b بدست می آید که نیمی از آن پاک است)

بنابراین اگر ورودی یک نیم پاک کننده دنباله ای Bitonic از صفر و یک باشد:

الف) هردو نیمه بالا و پایین دنباله خروجی Bitonic هستند.

ب) هر عنصر نیمه بالایی از هر عنصر نیمه پایینی کوچکتر یا مساوی است.

ج) حداقل یک نیمه پاک است یعنی یک دست یا صفر است و یا یک.

سوالات پایان ترم

1- در زمینه شعار سال ((تولید، پشتیبانی ها و مانع زدایی ها)) موارد زیادی برای گفتن وجود دارد مثلا در زمینه تولید یکی از مشکلاتی که در برابر تولید کنندگان وجود دارد ساختار اداری کشور است.مراحل بسیار سخت و طاقت فرسا برای گرفتن مجوز ، شروع به کار و..... در حالیکه تمام این موارد و رفت و آمد های بیش از حد قابل تغییر و کم شدن هستند در این زمینه می توانیم از مهندسین نرم افزار کمک بگیریم، تمام مراحل اداری دارای روند خاصی هستند که گام به گام باید طی شوند می توانیم این روند را به صورت نرم افزاری پیاده سازی کنیم که فرد بتواند در این سامانه ها و یا نرم افزار ها مدارک مورد نیازش را بارگزاری کند، موارد نیازمند تغییر را تغییر دهد و حتی با وجود شمایی از کارهای لازم به صورت مرحله به مرحله در این سامانه فرد داوطلب به راحتی می تواند مراحل لازم را ارزیابی کند و مدارک و مستندات خود را برای هرکدام از این مراحل آماده کند. و یا در مراحل تولید ، خود پروسه تولید را می توانیم به صورت اتوماتیک اجرا کنیم که کاملا مشخص است این کار توسط یک مهندس نرم افزار انجام می شود، و نیاز به گفتن نیست که در پشت پرده سامانه ها و یا نرم افزارها همیشه یک روند اجرا و یک الگوریتم وجود دارد که پایه و اساس ساخت و اجرای این سامانه ها می باشد.

در مورد پشتیبانی نیز روال کار را به همین صورت باید قرار بدهیم مثلا می توانیم از طریق یک وبسایت و قرار دادن فرمی در آن ابتدا به صورت نرم افزاری و بدون حضور متخصص مشکلات احتمالی وسیله ای که نیاز به پشتیبانی دارد را بررسی کنیم و باز هم اجرای این ایده و وبسایت و.....نیاز به طراحی یک الگوریتم و دنبال کردن مراحل آن است.

الگوریتم ها جزو یکی از ضروری ترین موارد موجود در زمینه کاری مهندسان نرم هستند و نقش بسیار پر رنگی در این زمینه دارند.

2- یکی از مشکلات عمده دانشجویان رشته مهندسی نرم افزار ناتوانی آنها در مشارکت در تولید نرم افزار است. در ادامه روندی برای مقابله با این مشکل مطرح شده است.

بازار کار مهندسی نرم افزار بسیار متنوع می باشد از کار در زمینه شبکه گرفته تا تولید نرم افزار یا اپلیکیشن و ..... اما در این سوال تنها تولید نرم افزار مد نظر است. این را می دانیم که کار عمده تولید نرم افزار کد زنی و برنامه نویسی است پس در این قسمت روند کاری برای یک دانشجو پیشنهاد می کنیم که بتواند در زمینه برنامه نویسی در بازار کار فعالیت داشته باشد.

الف) دانشجو پس از ورود به دانشگاه در ابتدا باید متوجه این مساله باشد که در محیط دانشگاه با زبان های برنامه نویسی در حد بسیار کمی آشنا خواهد شد. زمانی که این دیدگاه در دانشجویان شکل بگیرد در این زمینه تلاش بیشتری خواهند کرد.

پس در ابتدای الگوریتم پیشنهادی داشنجو باید روحیه پرسش گری را در خود پرورش دهد. و توانایی یادگیری را در خودش ایجاد کرده و تقویت کند به این معنا که دانشجو باید همیشه یک دانشجو و یک آموزنده باقی بماند.

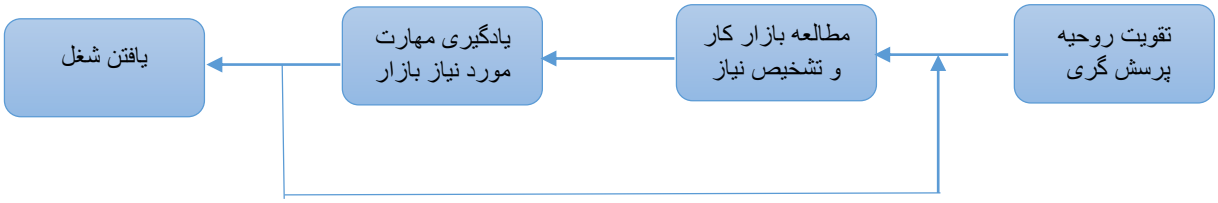
ب) پس از تقویت مهارت یادگیری و پرسش گری دانشجو باید به مطالعه بازار کار بپردازد (مثلا مشاهده استخدامی شرکتها) تا بتواند مهارت های مورد نیاز در بازار کار را شناسایی کند البته این مرحله باید به صورت مرتب تکرار شود به این معنی که دانشجو همیشه از نیازهای بازار کار مطلع باشد تا بتواند در سریع ترین زمان ممکن خود را با این نیاز ها تطبیق دهد.

ج) یادگیری آنچه در بازار کار مورد نیاز است و کاربرد دارد.(مثلا با مشاهده آگهی های استخدام در زمینه برنامه نویسی می بیند که بازار کار به کسی که در زمینه زبان برنامه نویسی پایتون مهارت داشته باشد نیاز دارد پس باید تلاش کند که این زبان برنامه نویسی را یاد بگیرد)

د) دانشجو پس از کسب مهارت های لازم باید سعی کند بسته به قابلیت ها و توانایی هایی که دارد کار پیدا کند.که البته بهترین راه حل برای شروع این است که به عنوان کارآموز کار کند و یا با درآمدهای بسیار کمتر نیز حاضر به کارکردن باشد.

ه) پس از انجام چند پروژه کاری دانشجو می تواند خواهان ترفیع رتبه و افزایش درآمد خود شده و با اعتماد به نفسی که پیدا کرده درصورت لزوم کار قبلی را ترک کند.

ی) اگر فردی بخواهد در زمینه تولید نرم افزار کارایی و مشارکت داشته باشد باید مراحل ب و ج را مرتب تکرار کند.



اکنون این سوال را میتوانیم از منظر دیگری هم بررسی کنیم:

فرض را بر این می گیریم که دانشجو یا دانش آموخته مد نظر ما تحصیل خود را به پایان رسانده و در زمینه تولید نرم افزار شروع به فعالیت کرده است:

اگر هدف از کارآمدی تولید نرم افزار با کیفیت و قابل اطمینان باشد و فرد در این زمینه موفق عمل کند باید مراحل زیر را با صبر و پشتکار فراوان انجام دهد.

- مرتب به خود یادآوری کند که چیزهای زیادی نمی داند:
  باید این نگرش را در خود به وجود بیاورد که همیشه باید چیزهای زیادی را بیاموزد و هرگز حس دانا بودن نداشته باشد.
- دست از تلاش برای اثبات درست بودن خود بردارید:
  گاهی نیاز است از الگوریتم ها و یا برنامه های آماده در کدهای خود استفاده کنید و البته برنامه نویسی موفق است که به دنبال نقص ها و کاستی ها باشد و می تواند این کار را با استفاده از داده های تستی امتحان کرد
- اینکه کد کار می کند، کافی نیست
  برنامه نویس با تجربه و قوی تنها به اینکه کدش کار می کند کفایت نمی کند و همیشه تلاش می کند کد نوشته شده را بهتر کند.
- کدهای نوشته شده توسط دیگران را بخوانید
  وقتی کد دیگران را می خوانید ، می بینید که چگونه شخص دیگری یک مسئله برنامه نویسی را حل کرده است. و به آن مانند یک درس و یا چالش نگاه کنید و از خود بپرسید : چگونه می توانم این کد را بنویسم؟ و تلاش کنید راه حل متفاوتی را پیدا کنید.
- کد نویسی کنید
  سعی کنید جدای از فعالیت هایی که در زمینه کاری انجام می دهید. تلاش کنید برای خودتان کد نویسی کنید این کار کمک می کند تا ابزارها و فن آوری هایی را یاد بگیرید که در شغل فعلی تان در دسترس نیست.
- با هر روشی که می توانید با برنامه نویس های متفاوت کار کنید.
  این ویژگی باعث می شود که مهارت گوش دادن به دیگران را بیاموزید
- تگنیک ها را یاد بگیرید ، نه ابزارها را

زبان ها ، ابزارها و روش های برنامه نویسی می آیند و می روند. به همین دلیل دریافت هرچه بیشتر تجربه با بیشترین زبان و چارچوب می تواند هزینه داشته باشد. بر اصول برنامه نویسی تمرکز کنید ، زیرا اصول هرگز تغییر نمی کنند.

البته می توان باز هم به این سوال دید دیگری داشت:

اینکه در فرآیند تولید نرم افزار مهندس نرم افزار چه مراحلی را تولید کند تا نتیجه کار مفید و کارآمد باشد:

که جواب این سوال مراحل تولید نرم افزار است:

تحلیل نیازمندی ها

طراحی نرم افزار

پیاده سازی و یکپارچه سازی

تست نرم افزار یا اعتبار سنجی

گسترش و یا نصب نرم افزار

نگه داری نرم افزار

3 - مسائل از نظر سختی به سه دسته تقسیم می شوند(منظور از سختی در واقع زمان اجرای الگوریتم های آنها است)

الف) مسائل P : در این گروه از مسائل اگر n اندازه ورودی باشد برای آنها الگوریتمی وجود دارد که قطعی بوده و زمان اجرای آن به صورت یک چند جمله ای از n بوده و دربدترین حالت ممکن زمان اجرا به صورت $O(n^k)$ است که k یک ثابت است.به طور کلی مسئله هایی را که برای آنها الگوریتم های چند جمله ای وجود دارد مسائل رام شدنی یا ساده در نظر می گیریم.

ب) مسائل NP: شامل آن دسته از مسئله‌هایی است که در زمان چند جمله ای تحقیق پذیر هستند به این معنی که ممکن است پیدا کردن جواب برای آن‌ها نیاز به زمان زیادی داشته باشد اما با داشتن یک جواب میتوان درستی یا نادرستی جواب را با استفاده از یک الگوریتم با پیچیدگی زمانی چند جمله ای تعیین کرد. هر مسئله ای در P در NP نیز هست چرا که اگر یک مسئله در P باشد بدون بررسی کردن جواب مسئله می توانیم آن را در زمان چند جمله ای حل کنیم.

ج) مسائل NP-Complete : مسائلی هستند که اثبات شده است به سرعت قابل حل نیستند. در تئوری پیچیدگیNP-Complete ها دشوارترین مسائل کلاس NP هستند و جزء مسائلی می‌باشند که احتمال حضورشان در کلاس P خیلی کم است. علت این امر این می‌باشد که اگر یک راه‌حل پیدا شود که بتواندیک مسئله NP-Complete را حل کند، می‌توان از آن الگوریتم برای حل کردن سریع همه مسائل NP-Complete استفاده کرد.

در یک حالت دیگر از تعریف NP_Complete می توان گفت:

1 – مسئله NP است.

2- مسئله NP – Hard است.

د) مسائل NP-Hard : مسائلی هستند که سخت تر یا مساوی مسائل NP هستند به این معنی که هر مسئله NP را می توان به یک مسئله NP-Hard تقلیل کرد و در واقع می توانیم بگوییم مسئله NP-Hard یک بخش از مسئله NP مطرح شده است و اگر بتوانیم برای NP-Hard یک راه حل پیدا کنیم به کمک آن می توانیم مسئله NP را نیز حل کنیم.