بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

دانشگاه صنعتی نوشیروانی بابل
برق و کامپیوتر

پروژه کارشناسی رشته مهندسی کامپیوتر گرایش نرم افزار

اتوماسیون فرآیند انتخاب واحد بوسیله ارائه پیشنهاد برنامه‌های ترمی خودکار

نگارش
طه رستمی

استاد راهنما
دکتر سید محمود سخائي

شهریور ۱۳۹۷

چکیده

در این پروژه کوشیده شده است تا با تحلیل و بررسی و سپس توسعه برنامه ای کاربردی در قالب نمونه آزمایشی، راهکاری عملی برای اتوماسیون فرآیند انتخاب واحد که در ابتدای هر ترم توسط دانشجویان انجام می‌شود ارائه گردد.

کلیدواژه‌ها: انتخاب واحد خودکار، انتخاب واحد اتوماتیک، انتخاب واحد، برنامه ترمی، برنامه درسی، پیشنهاد برنامه ترمی، پیشنهاد برنامه درسی

فهرست نوشتار

عنوان صفحه

فهرست تصویر

عنوان صفحه

فهرست پیوست‌ها

عنوان صفحه

فصل اول:
مقدمه

۱-۱ موضوع پروژه

در این پروژه کوشیده شده است تا با تحلیل و بررسی و سپس توسعه برنامه ای کاربردی در قالب نمونه آزمایشی، راهکاری عملی برای اتوماسیون فرآیند انتخاب واحد که در ابتدای هر ترم توسط دانشجویان انجام می‌شود ارائه گردد.

۱-۲ اهمیت و کاربرد ها

این موضوع از این این جهت که می‌تواند فرآیند انتخاب واحد را برای دانشجویان تسریع و تسهیل نماید مفید و مورد علاقه است چرا که با پیشنهاد دادن تعدادی برنامه ترمی، مطابق با نیاز ها و اولویت های تعیین شونده توسط دانشجو وی را قادر می‌سازد تا با توجه به سلیقه خویش برنامه‌ای را جهت انتخاب واحد برگزیند و حال آنکه پیدا کردن چنین برنامه‌ای بصورت دستی در عمل، معمولا کاری دشوارتر و اغلب زمانبرتر است.

علاوه بر اینها دانشگاه ها نیز می‌توانند مشتریان این پروژه باشند و از آن برای اعمال تغییراتی در فرآیند انتخاب واحد ترمی دانشجویان استفاده کنند. برای مثال آنها می‌توانند دانشجویان را قادر سازند تا پس از انتخاب برنامه، علاوه بر آنکه دانشجو می‌تواند کد درس ها را بصورت دستی در سامانه ثبت کند، امکان ثبت اتوماتیک درس ها با توجه به برنامه انتخابی را نیز اضافه کنند.

۱-۳تعریف صورت مسئله بصورت دقیق تر

در ابتدای هر ترم، دانشگاه برنامه‌ای تحت عنوان **دروس ارائه شده در ترم** ارائه می‌کند که در واقع جدولی است که سطر های آن شامل مواردی از جمله درس ، دانشکده ارائه دهنده درس ، استاد ارائه دهنده درس ، زمان و مکان برگزاری کلاس ها و تاریخ امتحان ، ظرفیت و ... می‌باشد و دانشجو در ابتدای هر ترم باید با توجه به این جدول برنامه ای را برای ترم خود برگزیند.

هر دانشجو علاوه بر اینکه خود را ملزم می‌داند تا برنامه‌ای که انتخاب می‌کند مطابق با قوانین، از جمله قوانین مصّوب وزارت علوم و آئین نامه های داخلی دانشگاه باشد، بدنبال برنامه ایست که با اولویت ها و سلیقه وی مطابق باشد.

بنابراین صورت مسئله بصورت دقیق تر اینست که ما می‌خواهیم دانشجو را در یافتن برنامه ترمی مورد نظرش یاری کنیم و بجای آنکه دانشجو بطور دستی بخواهد برنامه ای را برای خود برگزیند ما خود بر اساس اولویت های وی برخی از برنامه های ترمی مجازی که محدودیت های قانونی را ارضا می‌کنند پیدا کنیم و تعدادی از بهترین برنامه هایی را که در بازه زمانی تعیین شونده‌ای پیدا کرده ایم به او ارائه دهیم.

تا به اینجا ما صورت مسئله را بصورت دقیق تر بیان کردیم اما

این صورت مسئله همچنان ابهام دارد چرا که ما هنوز محدوده پروژه را مشخص نکرده ایم و به همین دلیل از محدودیت هایی که با توجه به حدود پروژه مشخص می‌شوند غافل مانده ایم.

ما محدوده پروژه را بدین صورت تعریف می‌کنیم که، پروژه ما با توجه به سیستم **گلستان** که در حال حاضر یکی از سامانه های مورد اقبال در زمینه سیستم های جامع آموزشی در میان دانشگاه های کشور محسوب می‌شود و برای رشته های فنی و مهندسی و براساس قوانین دانشگاه صنعتی نوشیروانی بابل انجام شود.بعلاوه مسئله فقط لازم است برای دانشجویانی که تنها در یک رشته تحصیلی مشخص تحصیل می‌کنند و برای دوره کارشناسی حل شود.

با مشخص شدن حدود پروژه محدودیت های جدیدی به پروژه اعمال می‌شوند برای مثال گفته شد پروژه قرار است برای سیستم گلستان نوشته شود و در حال حاضر در هنگام انتخاب واحد در این سیستم شما باید واحد ها را به ترتیبی ثبت کنید که همواره برنامه ترمی شما در وضعیت معتبری قرار داشته باشد یعنی برای مثال جهت اخذ درسی که همنیازی دارد ابتدا باید همنیاز آن درس را وارد کرده سپس خود درس را وارد کنید بنابراین به صورت مسئله این خواسته بصورت ضمنی اضافه می‌شود که علاوه بر پیشنهاد برنامه ترمی، درس های آن برنامه پیشنهادی را با ترتیبی قابل اخذ برای سیستم گلستان به نمایش درآورد.

بنابراین آنچه پیشتر بیان شد بعلاوه محدودیت هایی که حدود پروژه را تعیین می‌کنند صورت مسئله را بصورت دقیق تعریف می‌کنند.

۱-۴ اهداف پروژه

ما می‌خواهیم مسئله‌ای که پیشتر در آخر ۳-۱ بصورت دقیق تعریف کردیم را تحلیل کرده و یک برنامه کاربردی در قالب نسخه‌ای آزمایشی که بتواند مسئله را بصورت کارا حل کند پیاده‌سازی کنیم. همچنین ما می‌خواهیم در تحلیل و حتی‌الامکان در پیاده‌سازی خود بگونه‌ای عمل کنیم که این پروژه بجای آنکه به محدودیت های تعیین شده توسط حدود پروژه وابسته باشد، این محدودیت ها را ارضا کند اما تحلیل و پیاده‌سازی بگونه‌ای باشد که بتوان پروژه را برای محدوده های دیگر براحتی و بسرعت انجام داد برای مثال با صرف زمانی اندک یا حتی بدون صرف هیچ زمانی بتوان آنرا برای دانشگاه دیگری پیاده سازی کرد یا بتوان آنرا برای برنامه های درسی وزارت علوم که در آینده می‌آیند، بسرعت پیاده‌سازی کرد.

۱-۵ گام‌های انجام پروژه

پروژه‌ای که با آن سر و کار داریم یک پروژه نرم‌افزاری است و برای انجام یک پروژه نرم‌افزاری بصورت کلی می‌توانیم گام

های زیر را در نظر بگیریم:

- نیاز های پروژه را تشخیص داده و پروژه را تحلیل کنیم.
- به طراحی و پیاده سازی نرم‌افزار بپردازیم.
- به بازبینی و تست نرم‌افزار و ارزیابی تست بپردازیم.

مواردی که در بالا به آن اشاره کردیم بسیار کلی هستند و علیرغم آنکه تقریبا برای هر پروژه نرم‌افزاری لازم و ضروری می‌باشند اطلاعات ملموسی در اختیار ما نمی‌گذارند همچنین از ترتیب انجام کار ها نیز اطلاعاتی در اختیار نمی‌گذارند بنابراین فکر کردم خوب است بصورت تیتر وار مراحلی که واقعا برای انجام این پروژه طی شده است را ذکر کنم. این مراحل به ترتیب شامل موارد زیر می‌باشند:

- مطالعه و بررسی جدیدترین برنامه های درسی فنی مهندسی مصوب وزارت علوم و تحلیل کلی پروژه که طی آن داده های ورودی مسئله و همچنین مراحل لازم برای انجام مسئله از قبیل گام کاستن۱، گام کاستن۲، الگوریتم پیشنهاد برنامه ترمی و امکانات بیشتر بدست آمدند بعلاوه مشخص کردن تکنولوژی های مورد استفاده برای پیاده‌سازی و بررسی معماری های مختلف نیز در این بخش انجام شد.
- تهیه داده های اولیه - پیاده‌سازی بخش فراهم کننده خدمات فایل ها و مدلسازی دروس ارائه شده در ترم مبتنی بر سیستم گلستان - بعلاوه تست بخش پیاده‌سازی شده.
- مطالعه مجدد برنامه های درسی وزارت علوم بعلاوه برنامه های تازه تصویت شده در فاصله این بخش و بخش سری قبل با عمق بیشتر - تحلیل و بررسی این برنامه ها و تلاش برای ارائه ساختاری عمومی که بتواند برنامه های درسی وزارت علوم را مدل کند - پیاده‌سازی ساختار ایجاد شده و تست آن.
- پیاده‌سازی و تست بخش مقدماتی نرم‌افزار - پیاده‌سازی و تست گام های کاستن.
- تحلیل و پیاده‌سازی و تست امکانات بیشتر شامل نمایش جدولی برنامه های پیشنهاد شده توسط نرم‌افزار و نمایش آنها با ترتیبی قابل اخذ برای سیستم گلستان بعلاوه امکان افزودن اطلاعات جانبی برای هر برنامه ترمی پیشنهادی و ذخیره سازی آن.
- تهیه داده های جامع برای تست و ارزیابی نرم‌افزار.
- طراحی الگوریتم برای جستجو و یافتن و سپس پیشنهاد دادن برنامه های ترمی مناسب - پیاده‌سازی و تست الگوریتم.
- پیاده‌سازی بخش های **خانه** و **تنظیمات** و تست آنها.

- ارزیابی نرم افزار، تشخیص عیوب و سپس بازنگری و اصلاح کد بعلاوه تغییر بعضی از نامگذاری ها.

ارزیابی مجدد نرم افزار.

ما فعلا از توضیحات فنی که دقیقا هر کدام از این موارد ذکر شده چه هستند صرف نظر می‌کنیم. در بخش های بعدی با این که هر کدام از این موارد دقیقا چه هستند بیشتر آشنا می‌شوید.

همانطور که از لیست بالا قابل فهم است گام‌های انجام پروژه مانند آنچه در ابتدا بصورت کلی و در سه بخش ارائه شده بود قابل تفکیک و جدا از هم نیست بلکه بنا به فراخور هر مرحله برای انجام آن مرحله اقدام شده است با این وجود از آنجا که نوشتن گزارشی بر مبنای ده موردی که به آنها اشاره شده بود کاری دشوار است ما فصل‌بندی ها را بگونه‌ای قرار می‌دهیم که نه مانند لیست ده موردی ذکر شده بلکه به گونه ای باشد که فهم پروژه را تسهیل کند و توضیح و تشریح آن آسان تر گردد.

۱-۶ خلاصه

در این فصل ابتدا پروژه را بصورت کلی معرفی کردیم و بعد از آن گفتیم که چرا این پروژه می‌تواند حائز اهمیت باشد سپس به تعریف دقیق مسئله‌ای که این پروژه قصد حل کردن آن را دارد پرادختیم و پس از برشمردن اهداف پروژه، گام‌های انجام آن را ابتدا در حالت کلی و سپس در حالت واقعی بیان کردیم و پس از مقایسه این دو حالت، استدلالی آوردیم که دلیل شیوه تنظیم فصل‌های ما را نمایان می‌کند.

فصل دوم:
نتایج تحلیل و طراحی و آشنایی با مفاهیم پایه

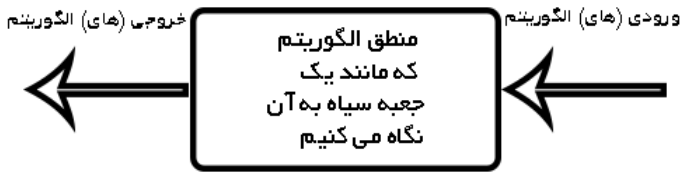
۲-۱ مقدمه

شرح کامل و جز به جز تحلیل و طراحی پروژه می‌تواند کاری خسته کننده باشد چرا که در فرآیند تحلیل اغلب به جزئیاتی برمی‌خوریم که برای تحلیل دقیق پروژه نیاز است آنها را بررسی کنیم اما در عمل و پس از اتمام فرآیند تحلیل و طراحی متوجه می‌شویم به بسیاری از این جزئیات نیازی نداشته ایم. ازینرو ما در اینجا بجای آنکه بخواهیم جز به جز آنچه تحلیل کرده ایم را بیان کنیم به تشریح نتایج حاصل از تحلیل و طراحی می‌پردازیم چرا که آنچه که عملا با آن سر و کار خواهیم داشت آنها هستند.

۲-۲ فشرده اول از تحلیل و طراحی

پس از تفکر پیرامون مسئله ای که در این پروژه قصد حل کردن آن را داشتیم متوجه شدیم که قلب این پروژه طراحی الگوریتمی جهت پیشنهاد برنامه ترمی می‌باشد.

از آنجا که هر الگوریتم در حالت کلی به ترتیب شامل سه قسمت : ورودی ، منطق الگوریتم و خروجی آن می‌باشد، سعی کردیم صرف نظر از جزیئات تصور کنیم که ورودی و خروجی الگوریتم پیشنهاد برنامه ترمی باید چگونه باشد.



شکل ۲-۱ دید ما از یک الگوریتم در حالت کلی

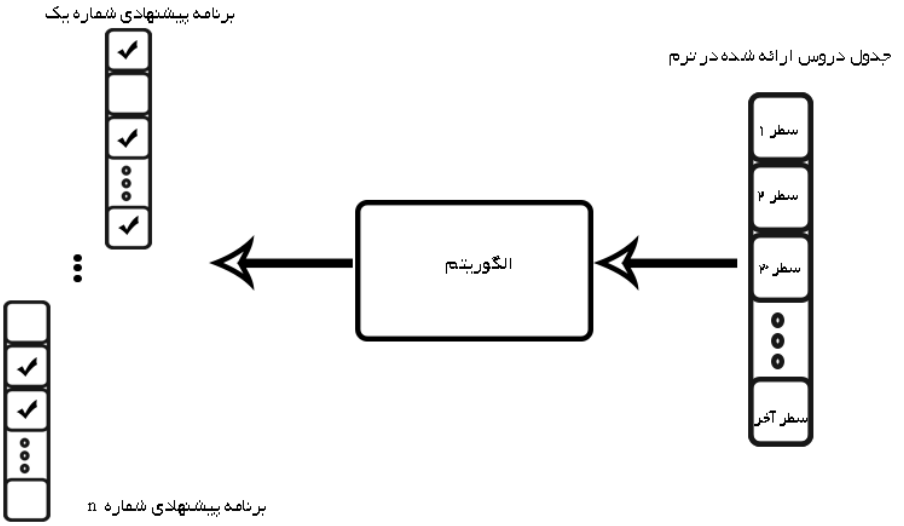
در واقع منطق الگوریتم را مانند جعبه سیاهی در نظر گرفتیم که فعلا نمی‌دانیم که قرار است چگونه باشد اما با مشخص کردن ورودی ها و خروجی های حاصل از آن می‌خواهیم عملکرد و تاثیری که این جعبه سیاه در خروجی هایش می‌گذارد را شناسایی کنیم.

در فصل قبل گفتیم **دروس ارائه شده در ترم** بصورت یک جدول، هر ترم توسط دانشگاه ارائه می‌گردد و این جدول شامل سطر هایی شامل مواردی از جمله درس ، دانشکده ارائه دهنده درس ، استاد ارائه دهنده درس ، زمان و مکان برگذاری کلاس ها و تاریخ امتحان ، ظرفیت و ... می‌باشد. در شکل زیر می‌توانید نمونه ای واقعی از سطر هایی از این جدول را که در سیستم گلستان ارائه شده است مشاهده کنید.

شماره و گروه درس	نام درس	واحد: کل	واحد: ع	ظرفیت	ثبت نام شده	تعداد لیست انتظار	جنسیت	نام استاد	زمان و مکان ارائه/ امتحان	توضیحات
۱۲۲۱۰۵۴_۰۱	هوش مصنوعي و سيستم هاي خبره	۳	۰	۵۰	۴۴	۰	مختلط	عمرانپور بندپي حسام	درس(ت): يك شنبه ۱۷:۳۰-۱۶:۰۰ مكان: ۲۰۸ دانشكده برق درس(ت): سه شنبه ۱۷:۳۰-۱۶:۰۰ مكان: ۲۰۸ دانشكده برق امتحان(۱۳۹۷/۰۳/۲۸) ساعت : ۱۶:۰۰-۱۴:۰۰	
۱۲۲۱۰۵۵_۰۱	اصول طراحي كامپايلر	۳	۰	۴۰	۲۷	۰	مختلط	غلامي رودي علي	درس(ت): يك شنبه ۱۵:۳۰-۱۴:۰۰ مكان: ۱۰۹ دانشكده برق درس(ت): سه شنبه ۱۵:۳۰-۱۴:۰۰ مكان: ۱۰۹ دانشكده برق امتحان(۱۳۹۷/۰۳/۲۰) ساعت : ۱۰:۰۰-۸:۰۰	
۱۲۲۱۰۵۶_۰۲	آزمايشگاه سيستم هاي عامل	۱	۱	۱۲	۱۲	۰	مختلط	جواهري عطيه	درس(ع): سه شنبه ۱۲:۰۰-۱۰:۰۰	
۱۲۲۱۰۵۶_۰۳	آزمايشگاه سيستم هاي عامل	۱	۱	۱۲	۱۱	۰	مختلط	غلامي رودي علي	درس(ع): شنبه ۱۲:۰۰-۱۰:۰۰	
۱۲۲۱۰۵۷_۰۲	آزمايشگاه مدارهاي منطقي و معماري كامپيوتر	۱	۱	۱۲	۱۲	۰	مختلط	كاكوئي محمد	درس(ع): سه شنبه ۱۰:۰۰-۸:۰۰	
۱۲۲۱۰۵۷_۰۳	آزمايشگاه مدارهاي منطقي و معماري كامپيوتر	۱	۱	۱۲	۱۲	۰	مختلط	كاكوئي محمد	درس(ع): سه شنبه ۱۸:۰۰-۱۶:۰۰	
۱۲۲۱۰۵۷_۰۴	آزمايشگاه مدارهاي منطقي و معماري كامپيوتر	۱	۱	۱۲	۱۲	۰	مختلط	كاكوئي محمد	درس(ع): شنبه ۱۶:۰۰-۱۴:۰۰	
۱۲۲۱۰۵۸_۰۱	روش پژوهش و ارايه	۲	۰	۴۵	۴۰	۰	مختلط	رماني فاطمه	درس(ت): يك شنبه ۱۶:۰۰-۱۴:۰۰ مكان: ۱۰۵ دانشكده برق امتحان(۱۳۹۷/۰۴/۰۹) ساعت : ۱۶:۰۰-۱۴:۰۰	
۱۲۲۱۰۵۹_۰۱	كارآموزي (مهندسي كامپيوتر)	۱	۰	۱۰	۱	۰	مختلط	اساتيد گروه آموزشي		
۱۲۲۱۰۶۶_۰۱	پروژه نرم افزار	۳	۳	۳۰	۳۰	۰	مختلط	اساتيد گروه آموزشي		دانشجويان جهت ثبت استاد وعنوان پروژه ازطريق پيشخوان خدمت اقدام نمايند.

شکل ۲-۲. سطرهایی از جدول دروس ارائه شده در یکی از ترم ها

ما همه‌ی ورودی ها و خروجی های الگوریتم را در این مرحله تعیین نمی‌کنیم اما بدیهی است که یکی از ورودی های اصلی برای مسئله ما جدول **دروس ارائه شده در ترم** می‌باشد و الگوریتم باید در خروجی تعدادی از سطر های این جدول را که می‌خواهد بعنوان برنامه پیشنهاد دهد برگرداند. با این اوصاف می‌توانیم هر سطر از این جدول را مانند جعبه سیاهی در نظر بگیریم که الگوریتم در مورد انتخاب شدن یا انتخاب نشدن آن سطر در هر برنامه پیشنهادی باید تصمیم گیری کند.



شکل ۲-۳ تشخیص نحوه برخورد الگوریتم با ورودی ها از روی خروجی ها

اما نکته اینست که تعداد سطرهایی که در جدول دروس ارائه شده در ترم وجود دارند نسبت به آنچه دانشجو مجاز به اخذ آنها در آن ترم است بسیار بیشتر است. این نکته برای ما الهام بخش ایده ای بود که آن را **کاستن** نام نهادیم.

ایده کاستن به این معنی است که ما سعی کنیم از فضای بزرگ مسئله بکاهیم و آنرا کوچک تر کنیم. در واقع این ایده سعی می‌کند برخی از سطرهای جدول دروس ارائه شده در ترم را که با توجه به شرایط دانشجو غیر قابل اخذ می‌باشند قبل از ارسال

برای الگوریتم از ورودی حذف کند.

۲-۳ فشرده دوم از تحلیل و طراحی

دیدیم که ایده کاستن چگونه می‌تواند با تلاش برای کوچک کردن فضای مسئله، فضای جستجو برای الگوریتم را کوچک تر کند. اما تا به اینجا ایده کاستن را توصیف کردیم و هنوز نگفتیم که این ایده قرار است چگونه در عمل بکار گرفته شود.در ادامه به تشریح این مقوله می‌پردازیم.

۲-۳-۱ گام کاستن ۱

هر دانشجو شرایطی دارد که با توجه به آن ممکن است دامنه سطرهای قابل اخذ جدول برنامه دروس ارائه شده در ترم برایش محدود تر شود.برای مثال باید سطرهایی که برای جنسیت زن بکار برده شده است را از دامنه سطرهای قابل اخذ برای دانشجو مرد حذف کرد.

دیدیم که یکی از راه های کاستن استفاده از شرایط دانشجو است. ما پس از تحلیل و بررسی بیشتر، شرایطی از دانشجو را که می‌توان از آن برای کاستن استفاده کرد مشخص کردیم و از میان آنها آن شرایطی را که می‌خواستیم در نرم‌افزار پیاده‌سازی کنیم مشخص کردیم.

در نهایت شرایطی را که برای پیاده‌سازی انتخاب کردیم شامل جنسیت، شماره ترمی که دانشجو در آن می‌خواهد انتخاب واحد کند(یعنی دانشجو ترم چند است)،تاریخچه دانشجو و تعداد واحد گذرانده شده توسط دانشجو (که این یک خصوصیت محاسباتی است و می‌توان آن را با توجه به تاریخچه دانشجو محاسبه کرد) می‌باشد.

ما تمام شرایط موجود دانشجو را انتخاب نکردیم برای مثال در سیستم گلستان سطرهایی وجود دارند که مشخص می‌کنند آن سطر خاص فقط برای ورودی های خاصی مثلا ورودی های ۹۱ و قبل از آن قابل اخذ می‌باشد و ما چنین شرطی را در لیستی که پیشتر گفتیم انتخاب نکردیم در این رابطه به چند نکته مهم توجه کنید. نکته اول آنکه برای این مثال خاص باید بدانید که سیستم گلستان برای هر سطر یک قسمت مربوط به توضیحات دارد و از آنجا که توضیحات احتمالا می‌توانند هر چیزی باشند ما در حالت کلی نمی-توانیم توضیحات را توسط نرم افزار پردازش کنیم مگر آنکه به برنامه قابلیت پردازش زبان طبیعی بدهیم و از آنجا که چنین موردی در دامنه فعلی پروژه ما نمی‌گنجید آن را انتخاب نکردیم. نکته دوم اینکه با این حال موردی که ذکر شد جز مواردی است که باید حتمی حذف سطر و کاستن برایش اتفاق بیفتد چرا که اگر در برنامه پیشنهادی الگوریتم چنین سطری موجود باشد، آن برنامه پیشنهادی نامعتبر خواهد بود ما برای مواجه با این مسئله دو راه حل داشتیم اولین راه‌حل آنکه هر چند قصد نداریم زبان طبیعی را پردازش کنیم اما این نوع توضیحات در سیستم گلستان و در دانشگاه صنعتی نوشیروانی بابل تا به اینجا اغلب و یا شاید همیشه با الگوهای خاص قابل تشخیصی نوشته شده‌اند بنابراین راه

اول ما می‌تواند این باشد که این الگو ها را شناسایی کنیم و پردازش را به همین الگو های شناسایی شده محدود کنیم اما همانطور که در فصل اول قسمت اهداف پروژه ذکر شده بود ما نمی-خواهیم پروژه خود را وابسته به محدوده مسئله کنیم و حال آنکه چنین راه‌کاری دقیقا یکی از مواردی است که می‌تواند قسمتی از پروژه را وابسته به محدوده مسئله کند. در عوض ما راه‌کار دیگری را برگزیدیم که در آن این نوع تشخیص بر عهده کاربر قرار می‌گیرد که در ادامه با آن آشنا خواهید شد.اما نکته سوم یک تاکید بیشتر است بر این موضوع که توجه داشته باشید که ما گفتیم می‌خواهیم **سعی**(سعی با انجام دادن حتمی کار بصورت کاملا موفق متفاوت است) کنیم از فضای حالات کلی بکاهیم و کاستن در اینجا به معنی کوچک تر کردن فضای مسئله با حذف **برخی** از سطر-های غیر مجاز است و نباید به اشتباه تصور کرد که الزاما آنچه بعنوان ورودی به الگوریتم داده می‌شود صرفا حالات مجاز است.

ما پیشتر از لفظ **تاریخچه دانشجو** استفاده کردیم مقصود ما از تاریخچه دانشجو اطلاعات مربوط به دروسی است که دانشجو آنها را با موفقیت گذرانده یا آنها را اخذ کرده اما نتوانسته با موفقیت بگذراند. ما با استفاده از اطلاعات حاصل از لیست انتخاب شده برای شرایط دانشجو به خصوص از تاریخچه دانشجو و با توجه به برنامه درسی وزارت علومی که دانشجو با آن در ارتباط است می‌توانیم متوجه شویم دانشجو در یک ترم خاص مجاز است چه درس هایی را در صورت ارائه شدن اخذ نماید.

از این پس ما به فرآیند حذف کردن سطرهایی از جدول برنامه دروس ارائه شده در ترم که با استفاده از شرایط انتخاب شده دانشجو شامل جنسیت، شماره ترمی که دانشجو در آن می‌خواهد انتخاب واحد کند،تاریخچه دانشجو و تعداد واحد گذرانده شده توسط دانشجو بعلاوه برنامه و اطلاعات مخصوص مربوط به برنامه درسی مصوب وزارت علومی که مرتبط با دانشجو می‌باشد، **گام کاستن ۱** می‌گوئیم.

۲-۳-۲ گام کاستن ۲

گام کاستن ۱ تلاش لازمی بود که باید نهایتاً در قسمتی از پروژه اگر نه خود آن ولی حداقل عملکرد آن اعمال می‌شد با این حال این گام تنها گامی نیست که می‌توان در جهت کاهش فضای حالات برداشت. در این قسمت با گام کاستن ۲ آشنا می‌شوید که گامی است در حالت کلی اختیاری ولیکن ممکن است در شرایطی به حالت اجبار نیز در آید.

شاید پیدا کردن یک برنامه ترمی مجاز با توجه به جدول برنامه دروس ارائه شده در ترم اغلب کار آسانی باشد اما کار زمانی سخت می‌شود که پای الویت های محدود کننده دانشجو به میان می-آید. بنابراین پیش از آنکه بگوئیم گام کاستن ۲ چیست می‌خواهیم بدانیم الویت های دانشجویان برای انتخاب برنامه ترمی مورد نظرشان چگونه است.

دانشجویان برنامه ترمی خود را بر اساس الویت هایی انتخاب می-کنند که ما این اولیت ها را به دو دسته کلی **الویت‌های باید** و **الویت های شاید** دسته‌بندی کردیم. الویت های باید الویت هایی هستند که باید در برنامه ترمی دانشجو **الزاما** ارضا شوند درحالیکه الویت های شاید الزام آور نیستند و می‌توانند در حالت کلی با سه جمله **بهتر است این گونه باشد ، بهتر است اینگونه نباشد** و **فرقی ندارد اینگونه باشد یا نباشد** توصیف شوند در کنار دو نوع الویت باید و شاید، الویت دیگری بنام **نباید** وجود دارد اما از آنجا که می‌توان نباید را با نقیض باید توصیف کرد از قرار دادن آن به عنوان یک دسته جداگانه از الویت ها خودداری کردیم.

رعایت قوانین از جمله قوانین وزارت علوم و آئین نامه های داخلی دانشگاه از الویت های باید به حساب می‌آیند اما الویت های باید الزاما محدود به رعایت قوانین نیستند بلکه اولویت های باید می‌توانند بر اساس آگاهی دانشجو از اموری **حتمی** ولی **غیر قابل مشاهده** برای ما (نرم افزار) الزام آور شوند.

می‌توان منشا اولویت های دانشجو را از **اهداف استراتژیک** وی دانست. اهداف استراتژیک اهدافی هستند که دانشجو با توجه به آنچه خود برنامه ریزی و هدف گذاری می‌کند مشخص می‌شوند برای مثال ممکن است دانشجویی بخواهد همزمان با تحصیل کار نیمه وقتی داشته باشد و بنابراین زمان های خاصی را الزاما در برنامه ترمی خود بخواهد خالی نگه دارد یا مثلا دانشجویی با این هدف که هر چه زود تر فارغ‌التحصیل شود بخواهد هر ترم حداکثر تعداد مجاز واحد را اخذ کند یا بعنوان مثالی دیگر دانشجویی درس خاصی را بخواهد حتمی با استاد خاصی بگیرد یا نگیرد و مثال های این چنینی دیگر که بسیارند.بنابراین برنامه ما باید این قابلیت را داشته باشد که دانشجو بتواند با استفاده از آن الویت های مد نظرش را برای نرم افزار مشخص کند.

پس از بحث بالا در این نوبت باید مشخص کنیم که نرم افزار ما چه اولویت هایی را می‌خواهد پوشش دهد. برای رسیدن به این هدف ابتدا لیستی از برخی نتایج معمول حاصل از اهداف استراتژیک دانشجویان به شرح زیر تهیه کردیم:

- مشخص کردن محدوده واحد (مثلا این که دانشجو بخواهد برنامه ترمی وی بین ۱۶ تا ۱۹ واحد باشد)
- اخذ کردن یک یا چند درس بصورت باید (دانشجو بخواهد درس یا درس هایی را در ترم جاری حتمی اخذ کند)
- اخذ نکردن یک یا چند درس بصورت نباید (دانشجو نخواهد در هر حال یک یا چند درس خاص را در ترم جاری اخذ کند.)
- اخذ کردن یک کلاس خاص(معادل یک سطر از جدول برنامه دروس ارائه شده در ترم) از یک درس بصورت باید (مثلا از بین چهار کلاس ارائه شده برای درس سیگنال و سیستم دانشجو بخواهد

حتمی یک سطر خاص از این درس اخذ شود)

- اخذ نکردن یک یا چند کلاس خاص از یک درس به هر حال.
- اخذ کردن یکی از چند درس مشخص بصورت باید (مثلا دانشجو بخواهد بین دو درس از گروه معارف حتمی یک درس را از بین آنها در ترم جاری اخذ کند)

علاوه بر موارد بالا می‌توان برای بعضی از آنها حالت شاید نیز اضافه کرد (برای مثال اینکه بگوییم این درس بهتر است در این ترم اخذ شود یا نشود) همچنین احتمالا علاوه بر همه این ها بتوان موارد دیگری نیز به این لیست افزود.

اگر بخواهیم از نگاه توانمندی کاربر به این لیست نگاه کنیم هر چه این لیست شامل موارد جزئی تر شود احتمالا کاربر در توصیف اولویت هایش توانا تر خواهد بود اما دو نکته از نظر تحلیل و طراحی وجود دارد اول آنکه زیاد شدن بیش از اندازه موارد می‌تواند بجای آنکه کار کاربر را آسان کند عملا باعث سختی بیشتر کار او شود چرا که قبل از آنکه کاربر بتواند از امکانات نرم افزار استفاده کند باید بداند که آنها چیستند و چگونه کار می‌کنند. نکته دوم اینکه از دید تحلیلی فقط توصیف اولویت های باید می‌توانند به کاستن فضای حالات مسئله بصورت مستقیم کمک کنند چرا که الویت های شاید تنها درصدی از اقبال و علاقه کاربر نسبت به سطرهای جدول برنامه دروس ارائه شده در ترم را بیان می‌کنند و در نتیجه نمی‌توان آنها را از جدول حذف کرد.با این اوصاف ما باید امکاناتی را برای پیاده سازی انتخاب کنیم که هم بتواند کاربر را در توصیف اولویت هایش توانا کند و هم باید از زیاد شدن بیش از اندازه امکانات بگونه ای که عملا کار را برای کاربر دشوار کند پرهیز کنیم، غیر از آن نباید این امکانات بگونه ای کم یا غیر کاربردی باشند که کاربر تمایلی به استفاده از آنها نداشته باشد (توجه داریم که گام کاستن ۲ در حالت کلی اختیاری است) علاوه بر همه این ها تمایل داریم امکاناتی را انتخاب کنیم که از نظر پیاده‌سازی نیز ساده‌تر باشند. با سبک سنگین کردن همه این موارد به این نتیجه رسیدیم که پنج مورد اول از لیست پیش‌تر بیان شده را پیاده‌سازی کنیم. در واقع برای گزینه های شاید فقط حالتی که با **فرقی ندارد اینگونه باشد یا نباشد** بیان می‌شود را باقی گذاشتیم و مورد ششم از لیست ذکر شده را صرفا برای راحت تر شدن پیاده‌سازی و همچنین بخاطر ترس از پیچیده شدن بیش از اندازه الگوریتم از لیست انتخابی حذف کردیم.

اما برای اینکه بتوانیم با توجه به اولویت های کاربر در مورد مواردی از قبیل ممکن بودن اولویت کاربر ، متناقض نبودن آن با سایر الویت هایش آگاهی یابیم و همچنین برای تصمیم گیری در مورد حذف کردن برخی از سطر ها جهت کاستن از فضای حالات، باید بدانیم که در بررسی هایمان کلاس هایی که ظرفیت آنها پُر

شده است را فیلتر کنیم یا فیلتر نکنیم بعلاوه باید بدانیم تداخل امتحانی نیز چک شود یا نشود. پس باید این دو مورد نیز از کاربر پرسیده شود.

از این پس به فرآیند دریافت اطلاعات و اولویت های کاربر و تلاش برای حذف سطرهایی از جدول برنامه دروس ارائه شده در ترم با توجه به اولویت های مشخص شده، **گام کاستن۲** می‌گوئیم.

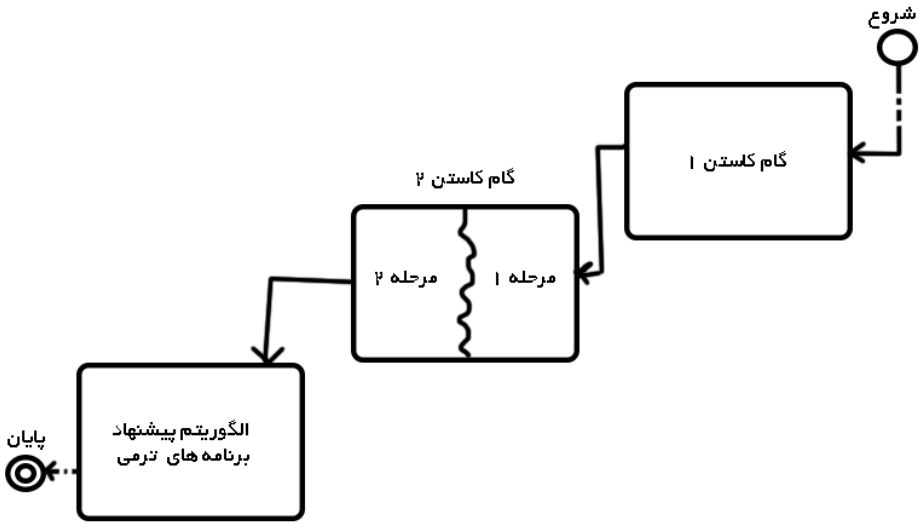
همچنین به فرآیند دریافت الویت ها و اطلاعات کاربر شامل حداقل و حداکثر واحد برای اخذ در ترم جاری،بررسی یا عدم بررسی تداخل امتحانات و فیلتر کردن یا فیلتر نکردن سطرهای با ظرفیت پُر و تلاش برای حذف سطرهایی از جدول برنامه دروس ارائه شده در ترم با توجه به اولویت های مشخص شده، **مرحله۱ از گام کاستن۲** می‌گوئیم.

همینطور به فرآیند دریافت الویت های کاربر بجز آنچه در مرحله۱ از گام کاستن۲ مشخص شد و تلاش برای حذف سطرهایی از جدول برنامه دروس ارائه شده در ترم با توجه به اولویت های مشخص شده، **مرحله۲ از گام کاستن۲** می‌گوئیم.

قبل از به پایان بردن این بخش می‌خواهم به قسمتی که در آن این سوال مطرح شده بود که در قبال ستون توضیحات سیستم گلستان چگونه عمل کنیم بازگشته، پاسخ دهم. اکنون اگر به لیست مواردی که برای گام کاستن۲ مشخص کرده ایم توجه کنید می‌بینید می‌توان براحتی با استفاده از این گام مشکل مربوط به ستون توضیحات را به کمک کاربر حل کرد و این یکی از همان شرایطی است که طی آن گام کاستن۲ از حالت اختیاری خارج می‌شود.

۲-۴ سایر مولفه ها و سازماندهی بخش های مربوطه

با توجه به آنچه تا به اینجا گفته شد می‌توان پروژه را پس از دریافت ورودی اولیه (یعنی ساختار توصیف کننده برنامه درسی مصوبی از وزارت علوم که با دانشجو در ارتباط است) به ترتیب به صورت بخش‌های گام کاستن۱، مرحله۱ از گام کاستن۲ ،مرحله۲ از گام کاستن۲ و سپس الگوریتم پیشنهاد برنامه ترمی در نظر گرفت.



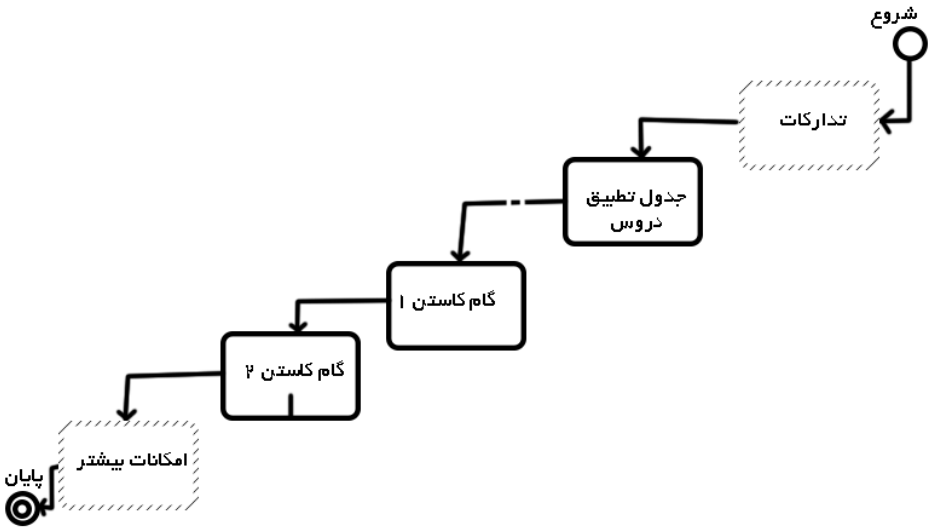
شکل ۲-٤ شمای کلی از مولفه های اصلی پروژه

حالا به صورت فشرده به سایر مولفه ها می‌پردازیم.

دانشگاه های مختلف الزاما تمام دروس موجود در برنامه درسی مصوب وزارت علوم را ارائه نمی‌دهند بعلاوه گاهی بجای یک درس موجود در برنامه مصوب وزارت علوم، درس معادلی را بجای آن ارائه می‌دهند ما در مواجه با این موضوع مولفه‌ای بنام **جدول تطبیق دروس** را به مولفه های نرم‌افزار اضافه کردیم. این جدول شامل سطرهایی با چهار ستون می‌باشد که این ستون ها عبارتند از شناسه درس، عنوان درس در برنامه درسی مصوب وزارت علوم ، نام درس در دانشگاه مقصد و کُد درس در دانشگاه مقصد.

از این پس به درس‌هایی از برنامه درسی مصوب وزارت علوم مرتبط با دانشجو که در دانشگاه مقصد(دانشگاهی که دانشجو با توجه به برنامه ارائه شده توسط آن دانشگاه انتخاب واحد می-کند) ارائه می‌شوند **درس‌های موجود** و به آنهایی که ارائه نمی‌شوند **درس‌های ناموجود** می‌گوییم.

علاوه بر مولفه جدول تطبیق دروس، دو مولفه بنام های مولفه **تدارکات** و مولفه **امکانات بیشتر** را نیز اضافه می‌کنیم. در واقع این دو مولفه را بعنوان جانگهدار برای قسمت هایی که ممکن است در تحلیل دیده نشده باشند به مولفه‌های پروژه افزودیم.



نمودار ۲-۵. تصویر کلی از مولفه های اصلی پروژه

پس از پیاده‌سازی، در نمونه آزمایشی پروژه، مولفه تدارکات شامل بخش‌های مربوط به **خانه** و **تعیین گرایش و تمرکز** شد و مولفه امکانات بیشتر نیز شامل بخش‌های **نمایش و افزودن اطلاعات بیشتر به برنامه های پیشنهادی و ذخیره سازی آنها** و **تنظیمات** شد.

برای قسمت هایی از پروژه مثلا برای تبدیل فایل های برنامه دروس ارائه شده در ترم به فایل قابل پردازش برای نرم افزار به مولفه ای مستقل از سایر بخش ها نیاز داریم که خدمات مربوط به فایل‌های مورد نیاز نرم افزار را فراهم کند. این مولفه را در عمل پیاده‌سازی می‌کنیم اما آنرا در شکل ها نشان نمی‌دهیم چرا که همانطور که گفتیم این بخش از سایر بخش ها مستقل پیاده‌سازی می‌شود. این مولفه را **فراهم کننده خدمات فایل ها** می‌نامیم.

۲-۵ خلاصه

در این فصل نتایج بدست آمده از تحلیل و طراحی پروژه را تشریح کردیم. در ابتدا سعی کردیم تاثیر و عملکرد الگوریتم پیشنهاد برنامه ترمی را شناسایی کنیم. در همین میان با ایده-ای بنام کاستن آشنا شدیم و بعد از آن بخش قابل توجهی از این فصل را به تشریح این ایده پرداختیم. در پایان نیز مولفه‌های دیگری از پروژه را شناسایی کرده و مولفه‌های پروژه را سازماندهی کردیم.

فصل سوم:
ساختارها، الگوریتم‌ها و روش‌های اصلی حل مسئله

۳-۱ مقدمه

در فصل قبل نتایج حاصل از تحلیل و طراحی پروژه را تشریح کردیم. در این فصل به مدلسازی داده‌ها و ارائه الگوریتم ها و روش‌های بکار گرفته شده برای حل مسائل اصلی پیش رو، می‌پردازیم.

۳-۲ ساختار توصیف کننده برنامه درسی مصوب وزارت علوم

یکی از مهمترین ورودی‌های مسئله ما برنامه درسی مصوب وزارت علومی است که با دانشجو در ارتباط است. این ارتباط ممکن است براساس مواردی از قبیل رشته تحصیلی، گرایش، سال ورود به دانشگاه و ... تعیین شود برای مثال دانشجوی مهندسی کامپیوتر ورودی ۹۳ باید از برنامه درسی مصوب سال ۹۲ وزارت علوم پیروی کند درحالیکه دانشجوی همین رشته اما ورودی ۹۷، با برنامه درسی مصوب سال ۹۶ آن، ارتباط دارد. تصمیم مهمی که باید اخذ می‌کردیم درباره چگونه مدلسازی کردن برنامه درسی مصوب وزارت علوم بود. در ادامه به تشریح مدلسازی خود می‌پردازیم.

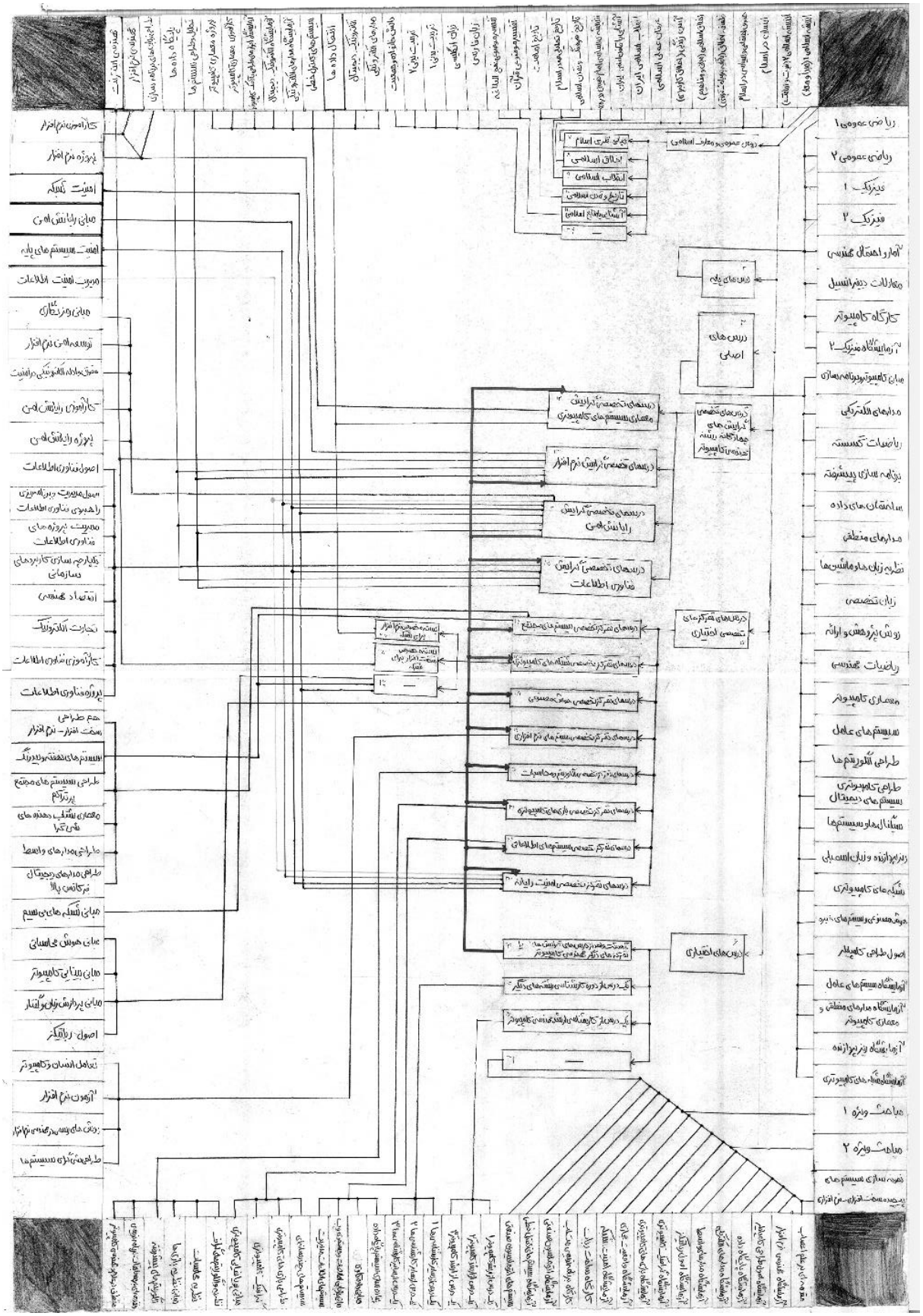
برنامه‌های درسی مصوب وزارت علوم شامل دو مولفه درس و محدودیت می‌باشند. درواقع هر برنامه درسی مصوب وزارت علوم شامل تعدادی درس است که وزارت علوم با محدودیت‌هایی که در برنامه درسی مصوب خود اعمال می‌کند شرایط اخذ هر درس را مشخص می‌کند. نکته دیگر اینکه این محدودیت‌ها انواع شناخته شده و مشخصی دارند که عبارتند از: پیشنیازی و هم‌نیازی دروس، حداقل ترم برای اخذ درس، حداقل تعداد واحد برای اخذ درس، حداقل و حداکثر واحد مجاز برای اخذ درس‌های موجود در یک گروه از درس‌ها یا انواعی از درس‌های یک گروه (مثلا از درس‌های پایه در برنامه مصوب ۹۲ کامپیوتر باید ۲۰ واحد گذرانده شود یا بعبارت دیگر حداقل و حداکثر ۲۰ واحد)، بعلاوه نوع دیگری از محدودیت‌ها که روی ارتباط دانشجو با برنامه درسی مصوب وزارت علوم اعمال می‌شود (مثلا برای دانشجو کامپیوتر ورودی ۹۳ این نوع از محدودیت را می‌توان، آن محدودیت هایی برای اخذ درس‌ها درنظر گرفت که به واسطه گرایش و تمرکز تخصصی دانشجو مشخص می‌شوند). می‌توان سه محدودیت اول نامبرده شده را بعنوان خصیصه های یک درس در نظر گرفت. به این ترتیب مدل ما باید توانایی این را داشته باشد که سایر محدودیت‌های وصف نشده را پشتیبانی کند و همچنین با محدودیت‌هایی که بعنوان خصیصه به درس‌ها انتساب داده شده‌اند سازگار باشد. با این اوصاف برای ساختار خود عناصری تعریف کردیم که با استفاده از آنها بتوان برنامه مصوب وزارت علوم را به شکل یا شکل‌هایی توصیف کرد. این عناصر عبارتند از:

- درس -> یک درس منبعی است که می‌تواند به یک یا چند گروه وصل باشد (ارتباط داشته باشد).
- گروه یا دسته درسی -> یک گروه یا دسته درسی، عنصری است که می‌تواند با دسته های درسی دیگر از طریق گیت‌های ورودی

یا خروجی ارتباط داشته باشد یا با درس های دیگر نیز ارتباط داشته باشد.

- کارت اعتباری -> ساختاری است که از آن برای تعیین اعتبار و تشخیص سطح دسترسی استفاده می‌شود.
- محدودیت -> ساختاری است که با آن می‌توان محدودیت‌های حداقل و حداکثر واحد مجاز برای اخذ درس‌های موجود در یک گروه از درس‌ها یا انواعی از درس‌های یک گروه را توصیف کرد.
- گواهینامه -> ساختاری است که می‌تواند شامل تعدادی محدودیت بعلاوه سطوح دسترسی معین که به کمک کارت‌های اعتباری قابل بیان است، باشد.
- گیت -> ساختاری است که می‌تواند تعدادی گواهینامه را بر روی دسته درسی مبدای به دسته درسی مقصدی مشخص کند.
- گیت ورودی و گیت خروجی -> گیت ورودی و گیت خروجی، مفاهیمی هستند که با توجه به جهت قراردادی، در حرکت از یک دسته درسی به دسته درسی دیگر شکل می‌گیرند. برای مثال فرض کنید بین دسته درسی الف و ب از طریق یک گیت، ارتباطی برقرار باشد و جهت قراردادی نیز، از الف به ب باشد در اینصورت مرجع گیت در الف گیت خروجی و مرجع گیت در ب گیت ورودی است.
- گروه یا دسته درسی فقط خروجی -> دسته درسی‌ای که همه گیت-های آن با توجه به جهت قراردادی، خروجی باشند را گروه یا دسته درسی فقط خروجی می‌نامیم.
- گروه یا دسته درسی فقط ورودی -> دسته درسی‌ای که همه گیت-های آن با توجه به جهت قراردادی، ورودی باشند را گروه یا دسته درسی فقط ورودی می‌نامیم.
- گروه یا دسته درسی هم ورودی هم خروجی -> دسته درسی‌ای که با توجه به جهت قراردادی هم گیت‌های ورودی و هم گیت‌های خروجی داشته باشد را گروه یا دسته درسی هم ورودی هم خروجی می‌نامیم.(در ساختار ما این نوعِ گروه نباید با هیچ درسی ارتباط باشد)
- ریشه -> گروه یا دسته درسی فقط خروجی را که ارجاع به هیچ درسی نداشته باشد ریشه می‌نامیم.(هر ساختار توصیفی ما باید دقیقا یک ریشه داشته باشد.)
- گروه یا دسته درسی سطح اول -> گروه‌ها یا دسته‌های درسی‌ای که با ریشه ارتباط داشته باشند را گروه‌ها یا دسته‌های درسی سطح اول می‌نامیم.
- نود پایانی یا لبه -> گروه‌ها یا دسته‌های درسی فقط ورودی را که با حداقل یک درس در ارتباط باشند، نود پایانی یا لبه می‌نامیم.

شاید درک این عناصر از روی توصیحات بالا کار دشواری باشد. در ادامه کار خود را با توضیح از روی یک مثال عملی ادامه می-دهیم.



تصویر ۳-۱ساختار توصیف کننده برنامه درسی وزارت علوم مصوب سال ۹۲ برای رشته مهندسی کامپیوتر

تصویر پیشین، تصویری است از ساختار توصیف کننده برنامه درسی مصوب سال ۹۲ وزارت علوم برای رشته مهندسی کامپیوتر. البته در این شکل همه عناصر ساختاری که پیش‌تر معرفی کرده بودیم نمایش داده نشده اند برای مثال در این تصویر عناصر مربوط به گواهینامه نمایش داده نشده است.

در چهارگوشه تصویر عناصر **درس** قرار گرفته اند که هر کدام از روی عنوان نوشته شده روی آنها قابل تشخیص می‌باشند.

جهت قراردادی ما از سمت **راست به چپ** می‌باشد و بنابراین جعبه سیاه رنگی که بالا و سمت راست تصویر قرار دارد و گروهی است فقط خروجی، نمایانگر **ریشه** می‌باشد. (سایر جعبه‌های سیاه صرفا جنبه نمایشی دارند)

جعبه‌هایی که در وسط صفحه قرار گرفته‌اند (یعنی همه جعبه ها بجز درس‌ها و سه جعبه تیره رنگ صرفا نمایشی)، **گروه**ها هستند.

گروه‌های دروس عمومی و معارف، درس‌های تخصصی گرایش‌های چهارگانه رشته مهندسی کامپیوتر، درس‌های تمرکز تخصصی اختیاری و دروس اختیاری، درس‌های تمرکز تخصصی شبکه‌های کامپیوتری و تا هشت واحد از درس‌های گرایش‌ها یا تمرکزهای دیگر مهندسی کامپیوتر **گروه‌های هم ورودی هم خروجی** هستند.

گروه‌های دروس عمومی و معارف، درس‌های پایه، درس‌های اصلی، درس‌های تخصصی گرایش‌های چهارگانه رشته مهندسی کامپیوتر، درس‌های تمرکز تخصصی اختیاری و دروس اختیاری گروه‌های **سطح اول** هستند.

گروه‌های درس‌های پایه، درس‌های اصلی، مبانی نظری اسلام، اخلاق اسلامی، انقلاب اسلامی، تاریخ و تمدن اسلامی، آشنایی با منابع اسلامی، خط تیره دروس عمومی و معارف اسلامی، درس‌های تخصصی گرایش معماری سیستم‌های کامپیوتری، درس‌های تخصصی گرایش نرم‌افزار، درس‌های تخصصی گرایش رایانش امن، درس‌های تخصصی گرایش فناوری اطلاعات، درس‌های تمرکز تخصصی سیستم‌های مجتمع، درس‌های تمرکز تخصصی هوش مصنوعی، درس‌های تمرکز تخصصی سیستم‌های نرم‌افزاری، درس‌های تمرکز تخصصی الگوریتم و محاسبات، درس‌های تمرکز تخصصی بازی‌های کامپیوتری، درس‌های تمرکز تخصصی سیستم‌های اطلاعاتی، درس‌های تمرکز تخصصی امنیت رایانه، یک درس از دوره کارشناسی رشته‌های دیگر، یک درس از کارشناسی ارشد مهندسی کامپیوتر، خط تیره دروس اختیاری، بسته مخصوص نرم‌افزار برای شبکه، بسته مخصوص سخت‌افزار برای شبکه و خط تیره درس‌های تمرکز تخصصی شبکه‌های کامپیوتری گروه‌های فقط ورودی هستند و **لبه**‌های این ساختار را تشکیل می‌دهند.

همانطور که قبلتر نیز اشاره کردیم در این تصویر همه‌عناصر نمایش داده‌نشده‌اند اما برای آنکه یک دید کلی از اینکه این عناصر چگونه و کجا هستند پیدا کنید در این رابطه چند مثال ارائه می‌کنیم- یالی که با گیت ورودی ریشه به گیت خروجی دروس اصلی مشخص شده را در نظر بگیرید، بروی این یال گواهینامه‌ای

قرار دارد که طبق آن حداقل و حداکثر ۵۹ واحد درسی می‌تواند از آن عبور کند (محدودیت) و اخذ درس‌های آن برای همه دانشجویان مجاز می‌باشد (کارت اعتباری) در واقع فرض کنید یک دانشجو کامپیوتر که با این برنامه درسی ارتباط دارد با گرایش و تمرکز مخصوص به خود از ریشه شروع به حرکت به سمت لبه ها که منابع (درس های مختلف) آنجا قرار دارند می‌کند، در این مسیر دانشجو باید بتواند از یال‌ها بدون نقض کردن شرایط آنها عبور کند، مثلا اگر دانشجو تا به حال ۳واحد از درس‌های اصلی را گذرانده باشد می‌تواند از ریشه به لبه‌ی درس‌های اصلی برود و از آنجا برای خود یکی از دروسی که مجاز به اخذ آنها است را انتخاب کند- بعنوان مثالی دیگر دانشجویی با گرایش مهندسی نرم افزار که شش واحد از درس‌های تخصصی و دو واحد آزمایشگاه از دروس اختیاری خود را گذرانده است در نظر بگیرید. در اینصورت این دانشجو اگر از مسیر ریشه به درس‌های تخصصی گرایش‌های چهارگانه مهندسی کامپیوتر حرکت کند، فقط مجاز به اخذ درس از درس‌های تخصصی گرایش نرم افزار می‌شود چرا که مجوز دسترسی به سه گروه تمرکز تخصصی دیگر را نداشته، تنها مجوز در این مسیر برای او مربوط به درس‌های تمرکز تخصصی گرایش نرم افزار می‌باشد و از آنجا که تا به حال شش واحد از دروس این گروه را گذرانده بدون نقض کردن محدودیت‌ها می‌تواند به این گروه رسیده از آن درس یا درس‌هایی مجاز برای خود اخذ کند. این در حالیست که همین دانشجو اگر بخواهد از ریشه به دروس اختیاری و سپس به دروس تخصصی گرایش نرم افزار برسد مجوز لازم برای اینکار را نداشته ولی مجوز دسترسی به دروس تخصصی سایر گرایش‌ها را دارد که اگر بتواند از میان درس‌های مجاز درسی را بدون نقض کردن محدودیت های موجود در طول مسیر بیابد می‌تواند آنرا اخذ کند. برای مثال اگر دانشجو درس مدارهای الکترونیکی را نگذرانده باشد و مجاز به اخذ این درس هم باشد (مثلا پیشنیازی همنیازی برقرار باشد)، می‌تواند این درس را اخذ کند چرا که هیچ محدودیتی را در طول مسیر نقض نمی‌کند؛ حالا فرض کنید این دانشجو درس آزمایشگاه الکترونیک دیجیتال را نگذرانده باشد و مجاز به اخذ آن نیز باشد، دراینصورت اما دانشجو نمی‌تواند این درس را اخذ کند چرا که پیشتر گفته بودیم دانشجو دو واحد آزمایشگاه از دروس اختیاری خود را گذرانده است و اگر بخواهد این درس را اخذ کند موجب نقض محدودیت در طول مسیر می‌شود (برای مثال روی یال ریشه به دروس اختیاری محدودیت حداقل و حداکثر دو واحد آزمایشگاه قرار داده شده است).

۱-۲-۳ نقص ها و چالش‌ها

ما در پیاده‌سازی خود از همین ساختار برای توصیف برنامه درسی مصوب وزارت علوم استفاده کرده‌ایم اما این ساختار در حالت کلی ضعف‌هایی دارد و همچنین با چالش‌هایی مواجه است.در ادامه به تشریح این موارد می‌پردازیم.

در برنامه درسی مصوب سال ۹۲ وزارت علوم برای رشته

کامپیوتر، ایراداتی وجود داشت مثلا در جایی نام درس، گسسته و در جایی دیگر از نام ساختمان گسسته استفاده شده است در حالیکه منظور از هر دو همان گسسته بود و غیره. هنگامی که من این برنامه را تحلیل می‌کردم مشاهده کردم پیشنیازها یا همنیازهای یک درس در جاهای مختلف، متفاوت است برای مثال، در این برنامه درسی برای درس پایگاه داده‌ها در گروه تخصصی نرم-افزار و همچنین رایانش امن، درس ساختمان‌های داده به عنوان پیشنیاز در نظر گرفته شده است در حالیکه برای همین درس در گروه تخصصی فناوری اطلاعات، درس تحلیل و طراحی سیستم‌ها بعنوان پیشنیاز ذکر شده است. من همانطور که گفتم می‌دانستم این برنامه دارای ایراداتی است به همین خاطر، هنگام تحلیل این قسمت گمان کردم که این هم یک اشتباه از طرف وزارت علوم است و مدلسازی خود را با همین فرض انجام دادم اما در حقیقت چنین موردی یک اشتباه از طرف وزارت علوم نبود؛ یعنی پیشنیازی و همنیازی یک درس از خصوصیات آن درس نمی‌باشد(که این همان کاری است که ما در تحلیل و پیاده‌سازی خود کردیم) بلکه پیشنیازی و همنیازی، در رابطه یک درس با یک گروه قابل تعریف است. بنابراین این یکی از ضعف‌های تحلیل و در پی آن پیاده‌سازی ما می‌باشد که برخی خصایص را بجای آنکه به رابطه بین درس و گروه اختصاص دهیم به اشتباه آنها را بعنوان خصایص خود درس در نظر گرفتیم. البته ما برای آنکه برنامه حتی در مواجه با چنین موردی به مشکل نخورد راهکاری ارائه کردیم که در آینده درباره آن توضیح می‌دهیم اما به هر حال این یک ضعف مهم در تحلیل و سپس پیاده‌سازی ما محسوب می‌شود.

حال به یک چالش بسیار مهم می‌پردازیم، در ساختار فعلی ممکن است با یک کارت اعتباری مشخص بتوان به بعضی از درس‌ها از طریق چند گروه دسترسی پیدا کرد برای مثال یک دانشجوی کامپیوتر مرتبط با برنامه درسی مصوب سال ۹۲ وزارت علوم با گرایش مهندسی نرم‌افزار و تمرکز الگوریتم و محاسبات را در نظر بگیرید، این دانشجو از سه گروه درس‌های تخصصی نرم‌افزار، فناوری اطلاعات و رایانش امن به درس پایگاه داده‌ها دسترسی دارد، در اینگونه موارد باید مشکل را با **استنتاج منطقی** حل کرد. برای نمونه در مواجه با مثالی که بیان شد، می‌توانیم اینگونه استنتاج کنیم که دانشجو باید حداقل و حداکثر ۱۹ واحد از درس‌های تخصصی گرایش نرم‌افزار اخذ کند و چون اگر پایگاه داده را از این گروه اخذ نکند(یعنی اگر این درس را از یکی از دو گروه دیگر اخذ کند) محدودیت‌های این گروه دسترسی پذیر ارضا نخواهد شد، پس دانشجو باید این درس را از این گروه اخذ کند و نه گروه دیگری. اما مسلئه همیشه به این راحتی نیست. در مثال قبل هم استنتاج ساده بود و هم نتیجه آن به یک امر اجباری منتهی شد یعنی دانشجو باید درس پایگاه داده را از گروه خاصی اخذ می‌کرد، اما می‌توان برنامه‌های درسی‌ای را متصور شد که استنتاج در آنها به نتیجه اختیاری (نه اجبار دانشجو) منتهی شود که در این صورت باید بدانیم یک درس را از چه مسیری اخذ می‌کنیم بعلاوه اینکه استنتاج ممکن است بسیار پیچیده‌تر باشد

مثلا با اخذ یک درس بدون اینکه هیچ محدودیتی در طول مسیر نقض شود دانشجو در شرایطی قرار بگیرد که امکان فارغ‌التحصیلی را از دست بدهد. با این اوصاف اگر بخواهیم مسئله را با همین مدلسازی برای حالت عمومی حل کنیم دست کم باید به پروژه خود یک **عامل استنتاج کننده** اضافه کنیم. اضافه کردن یک عامل استنتاج کننده برای من در این وهله از نظر فنی کار آسانی نبود بعلاوه اینکه نمی‌دانستم، مسئله چقدر قرار است پیچیده‌تر شود. بنابراین راهکار دیگری برای مواجه با این چالش انتخاب کردم.
بجای افزودن عامل استنتاج کننده، به هنگام تعیین شدن کارت اعتباری دانشجو، فرآیندی بنام **اصلاح ساختار برنامه درسی** را معرفی کردم که در آن پس از تعیین شدن کارت اعتباری دانشجو، ساختار برنامه درسی با توجه به استنتاج‌های از قبل انجام شده توسط طراح، خود را به گونه‌ای تغییر می‌دهد که دانشجو به هر درس، حداکثر یک مسیر قابل دسترس داشته باشد. برای نمونه دانشجوی مثال قبل را با همان کارت اعتباری در نظر بگیرید، وقتی ساختار، کارت اعتباری دانشجو را دریافت کرد اتصال بین درس پایگاه داده‌ها و گروه‌های فناوری اطلاعات و رایانش امن را حذف می‌کند.
راهکاری که ارائه شد موجب می‌شود دامنه برنامه‌هایی که پروژه ما قادر به پشتیبانی از آنها است کاهش پیدا کند و در حالت کلی راه‌کار مطلوبی نیست با این حال باید در نظر داشت که برنامه‌های درسی فعلی وزارت علوم هم چندان برنامه‌های پیچیده‌ای نیستند و همین راه‌کار می‌تواند در حال حاضر پاسخگوی بسیاری از نیازهای ما باشد.

۳-۳ الگوریتم گام کاستن ۱

در فصل قبل با گام کاستن ۱ آشنا شدیم. حالا بعد از آشنا شدن با ساختار توصیف کننده برنامه درسی مصوب وزارت علوم می‌خواهیم الگوریتم گام کاستن ۱ را تشریح کنیم.

در گام کاستن ۱ ما بعنوان ورودی با یک ساختار برنامه درسی مصوب وزارت علوم که با توجه به جدول تطبیق دروس و با استفاده از تاریخچه دانشجو پر شده است مواجهیم و همچنین اطلاعات کارت اعتباری دانشجو و یک سری اطلاعات جانبی نیز در اختیار داریم. در این الگوریتم باید به ازای هر درس موجود در ساختار بررسی کنیم که آیا پس از اخذ شدن آن درس، آیا مسیری دسترس پذیر از درس در حال بررسی به ریشه به گونه‌ای که در طول مسیر محدودیتی نقض نشود وجود دارد یا خیر؟ فقط در صورتیکه پاسخ مثبت باشد، درس مورد بررسی را در لیست نجات ذخیره می‌کنیم. پس از اتمام بررسی برای همه درس‌ها، درس‌های موجود در لیست نجات را بعنوان خروجی الگوریتم برمی‌گردانیم.

```
List<Course> Reduce1(Curriculum curriculum,CreditCard studCredit,
                     int cntPassedUnits, int currentTermNumber)
```

```
{
            bool[] visited = new bool[curriculum.Courses.Count];
            bool[] achivable = new bool[visited.Length];

            List<Course> lst = new List<Course>();
            curriculum.Courses.ForEach(course =>
            {
                if (Acheivable(visited, achivable, curriculum,studCredit,
                    course, cntPassedUnits, currentTermNumber))
                {
                    lst.Add(course);
                }
            });
            return lst;
}

bool Acheivable(bool[] visited, bool[] achivable,
                        Curriculum curriculum,CreditCard studCredit,
                        Course c, int cntPassedUnits,
                        int currentTermNumber){
            //memoization :=> dynamic programming
            if (visited[c.Id])
            {
                return achivable[c.Id];
            }


            if (!c.IsAvailable())
            {
                return false;
            }
            else if (c.IsPassed)
            {
                return false;
            }
            else if (c.MinRequireTerm > currentTermNumber)
            {
                return false;
            }
            else if (c.MinReuireUnits > cntPassedUnits)
            {
                return false;
            }

            for (int i = 0; i < c.PrerequisiteCourses.Count; i++)
            {
                var pr = c.PrerequisiteCourses[0];
                if (!pr.IsPassed)
                {
                    if (pr.NumberOfFailed > 1)
                    {
                        if (!Acheivable(visited, achivable, curriculum,
                            studCredit, pr, cntPassedUnits,
                            currentTermNumber))
                            return false;
                    }
                    else
                    {
                        return false;
                    }

                }

            }

            for (int i = 0; i < c.RequisiteCourses.Count; i++)
            {
                var r = c.RequisiteCourses[i];

                if (!r.IsAvailable())
                    return false;
                if (r.IsPassed == false && !Acheivable(visited, achivable,
                    curriculum, studCredit, r, cntPassedUnits,
                    currentTermNumber))
```

```
                return false;

            }


            var res = if exists a path from c to root then return true
                      else return false:inputs(curriculum,studCredit,c);

            //memoization
            visited[c.Id] = true;
            achivable[c.Id] = res;

            return res;
 }
```

۳-۴ الگوریتم مرحله ۱ از گام کاستن ۲

در گام کاستن ۱ تعدادی از درس‌ها نجات پیدا کردند اما ممکن است همه‌ی این درس‌های نجات پیدا کرده در ترم جاری توسط دانشگاه ارائه نشده باشند. در این مرحله لیستی از دورس نجات پیدا کرده از مرحله قبل را به همراه برنامه دروس ارائه شده توسط دانشگاه در ترم جاری و به همراه اطلاعات جانبی به الگوریتم ارسال می‌کنیم. سپس الگوریتم سطرهایی از سطرهای ارائه شده از جدول برنامه دروس ارائه شده توسط دانشگاه را که درس متناظر آن در لیست دروس نجات یافته از مرحله قبل موجود باشد، برای این مرحله در لیست نجات مخصوص خود ذخیره می‌کند و در پایان، الگوریتم همه سطرهای نجات پیدا کرده را برمی-گرداند.

توجه به این نکته نیز ممکن است خالی از لطف نباشد که ما در کدها یا شبه‌کدهایی که می‌آوریم هم برای مفهوم لیست و هم برای مفهوم مجموعه از نمایش List<T> کمک می‌گیریم. در واقع مجموعه را لیستی با عناصر متمایز در نظر می‌گیریم که تعریف درستی نیست اما همین برای کار ما کفایت می‌کند.

```
List<OfferedCoursesRow> Reduce2(List<Course> courses, List<OfferedCoursesRow>
offeredCoursesRows, Gender studGender
,bool capacityFiltering=false)
{
   var groups = from r in offeredCoursesRows
                    where ((r.Gender == studGender
                            || r.Gender == Gender.COEDUCATIONAL) &&
                           (capacityFiltering?CalculateCapacity(r):true))
                         group r by r.CodeInDesUni;
   var newLst = new List<OfferedCoursesRow>();
   foreach (var group in groups)
   {
     var q = courses.FirstOrDefault(c => c.CodeInDesUni == group.Key);
     if (q != null)
     {
        foreach (var row in group)
        {
          newLst.Add(row);
        }
     }
   }

   return newLst;
}
```

۴-۵ ساختارها و الگوریتم‌های اصلی مرحله۲ از گام کاستن۲

بعد از مرحله۱ از گام کاستن۲ ما با تعدادی سطر از جدول دروس ارائه شده در ترم توسط دانشگاه مواجه‌ایم که هنوز احتمال انتخاب شدن برای آنها وجود دارد. در فصل قبل گفتیم در این مرحله می‌خواهیم با دریافت اولویت‌های کاربر سعی کنیم بازهم فضای مسئله را کاهش دهیم. حال در اینجا باید تصمیم بگیریم که چگونه می‌توانیم اولویت‌های لیست شده را از کاربر دریافت کنیم و چگونه این کاهش را باید صورت داد؟! در ادامه به این موضوعات می‌پردازیم.

۴-۵-۱ درس‌ها و سطرهای رنگی

برای آنکه کاربر را قادر سازیم تا بتواند الویت‌های انتخاب شده در فصل قبل را بیان کند سه رنگ سبز، سفید و قرمز را در نظر می‌گیریم که هر درس و همچنین هر سطر می‌تواند یکی از این سه رنگ را داشته باشد. رنگ سبز برای یک سطر، به معنای اینست که کاربر می‌خواهد آن را حتمی اخذ نماید. رنگ سفید برای یک سطر به معنای اینست که کاربر در رابطه با اخذ کردن یا اخذ نکردن آن سطر نظر قاطعی ندارد و رنگ قرمز به معنای اینست که کاربر به هیچ وجه نمی‌خواهد سطر فعلی را اخذ نماید. به طور مشابه درس سبز به معنای اینست که کاربر می‌خواهد درس فعلی حتمی در برنامه ترم جاری‌اش وجود داشته باشد، درس سفید یعنی که در این رابطه نظر قاطعی ندارد و درس قرمز هم یعنی این درس به هیچ عنوان در ترم جاری اخذ نشود. با استفاده از مفهوم درس‌ها و سطرهای رنگی می‌توان همه اولویت‌های انتخاب شده فصل قبل را توصیف کرد.

۴-۵-۲ الگوریتم های مرحله۲ از گام کاستن۲

در حین توصیف اولویت‌ها توسط کاربر ما باید، بررسی کنیم که اولویت‌های کاربر مجاز بوده و با هم در تناقض نباشند. در آخر پس از اتمام کار توصیف اولویت‌های کاربر نیز باید ورودی‌ها را برای الگوریتم مهیا کنیم. یکی از مهمترین بخش‌های این اعتبارسنجی اینست که مجموعه درس‌های سبز انتخاب شده توسط کاربر باید ساختار را در وضعیت معتبر نگه دارد بعلاوه اینکه باید حداقل یک برنامه ترمی مجاز و ممکن برای این مجموعه وجود داشته باشد. در این بخش به این موضوع که چطور می‌توانیم متوجه شویم پس از سبز کردن تعدادی درس آیا در وضعیت معتبر قرار داریم یا نه می‌پردازیم اما توضیح اینکه چطور می‌توانیم متوجه شویم برای یک مجموعه درس، برنامه ترمی مجاز و ممکنی وجود دارد یا نه را تا رسیدن به بخش مربوط به الگوریتم پیشنهاد برنامه ترمی به تاخیر می‌اندازیم.

ما بعنوان ورودی، ساختار برنامه ترمی‌ای که با توجه به جدول تطبیق دروس و تاریخچه دانشجو پر شده است را بعلاوه کارت اعتباری دانشجو و مجموعه دروس سبز رنگ درخواستی دانشجو دریافت می‌کنیم. برای آنکه بتوانیم تشخیص دهیم که آیا با اخذ

کردن مجموعه دروس مشخص شده، ساختار در وضعیت معتبری است یا خیر کافی است بدانیم پس از اخذ همه این دروس، قانونی نقض شده است یا نه. اگر قانونی نقض نشده باشد یعنی در وضعیت معتبری قرار داریم و در واقع مجاز به اخذ آن مجموعه دروس می‌باشیم. اما در غیرانصورت در وضعیت نامعتبر بوده مجاز به اخذ آن مجموعه از دروس نمی‌باشیم. اما پیش از آنکه راهکاری برای این اعتبارسنجی پیشنهاد کنیم باید قوانین را بشناسیم و بدانیم چگونه باید آنها را بررسی کنیم.

قوانین را می‌توان از زوایای مختلف به شکل‌های گوناگون تقسیم بندی کرد ما در اینجا آنها را به دو نوع **قوانین پایه** و **قوانین ثانویه** تقسیم‌بندی کردیم. قوانین پایه، قوانینی هستند که نسبت به برنامه درسی مصوب وزارت علوم ثابت اند مثلا چون در برنامه درسی مصوب سال ۹۲ وزارت علوم برای رشته مهندسی کامپیوتر، مفهومی مثل پیشنیازی و همینازی دروس وجود دارد و از آنجا که در این برنامه ذکر شده که: "این برنامه از تاریخ تصویب به مدت پنج سال قابل اجراء است و پس از آن نیازمند بازنگری است." ما انتظار داریم پیشنیازی و همنیازی و همچنین بصورت کلی قوانین مربوط به این برنامه در طول پنج سال عمر آن رعایت شده و ثابت بمانند. اما در کنار این قوانین، مواردی هستند که در طول عمر برنامه درسی مصوب وزارت علوم ممکن است تغییر کنند(برای مثالی از این قانون، مجاز بودن اخذ حداکثر یک درس معارف در یک ترم تحصیلی را در نظر بگیرید). پس ما کافی است این دو نوع از قوانین را برای تشخیص معتبر بودن یا نبودن وضعیت پس از انتخاب درس‌های سبز بررسی کنیم. بررسی کردن قوانین ثانویه معمولا سریع‌تر انجام می‌شود از اینرو در بررسی-های خود ترجیح دادیم ابتدا این قوانین را بررسی کنیم.

```
bool IsValidState(Curriculum curriculum,CreditCard studCredit, List<int> takenCoursesId)
{
  //secondary rules
  bool o = SecondaryRulesStateValidator.IsValidState(curriculum,
                                studCredit, takenCoursesId);
  if (!o) return false;

  //basic rules
  return BasicRulesStateValidator.IsValidState(curriculum,studCredit,
                                                  takenCoursesId);
}
```

یکی از قوانینی که در قسمت اعتبارسنجی قوانین پایه باید صورت پذیرد، بررسی اینست که آیا پس از اخذ مجموعه دروس سبز ورودی، محدودیت قابل دسترسی نقض می‌شود یا نه. برای بررسی این مورد شبه کدی ارائه نمی‌کنیم درعوض عملکرد کلی آن را شرح می‌دهیم. برای تشخیص این موضوع،ابتدا با استفاده از دروس سبز ورودی باید اطلاعات در ارتباط با ساختار توصیف کننده برنامه درسی مصوب وزارت علوم را بروز رسانی کرد سپس باید از ریشه شروع به جستجو کرده و از همه گروه‌های دسترسی پذیر عبور کرد، چنانچه در مسیر جستجو موردی مشاهده شد که دسترسی به آن مجاز بود اما محدودیتی نیز به واسطه ان دسترسی نقض می‌شود، یعنی وضعیت ساختار برنامه درسی نامعتبر است و اگر تا پایان جستجو چنین موردی یافت نشد یعنی در وضعیت معتبر قرار دارد.

۴-۶ الگوریتم پیشنهاد برنامه‌های ترمی

در این بخش می‌خواهیم شما را با الگوریتم اصلی برنامه که همان الگوریتم پیشنهاد برنامه(های) ترمی است آشنا کنیم. اما از آنجا که خود الگوریتم از زیرساخت‌ها و حل مسئله‌های دیگر بهره می‌برد ما ابتدا به این موضوعات پرداخته سپس به خود الگوریتم می‌پردازیم.

۴-۶-۱ زیرساخت اول

ما پیشتر در این فصل هنگام مواجه شدن با مسئله‌ای برای اعتبارسنجی درس‌های سبز، گفتیم که درباره آن بعد تر صحبت می‌کنیم؛ اکنون زمان آن رسیده است. جدول برنامه درسی ارائه شده در ترم را که توسط دانشگاه ارائه می‌شود به یاد بیاورید. گفته بودیم که این جدول شامل سطرهایی است که هر سطر آن نیز شامل ستون‌هایی است. بعلاوه برخی از ستون‌ها را نام برده بودیم. یک ستون از ستون‌های هر سطر، مربوط به شماره و گروه درس(که از آن می‌توان کد درس را نیز بدست آورد) می‌باشد.

حال فرض کنید یک مجموعه از درس‌های سبز را بعنوان ورودی به شما می‌دهند و می‌پرسند با فرض این که اخذ این دروس، سایر قوانین را نقض نکرده باشد، همه برنامه‌های ترمی مجاز و ممکن را برای آن‌ها بعنوان خروجی برگردان.

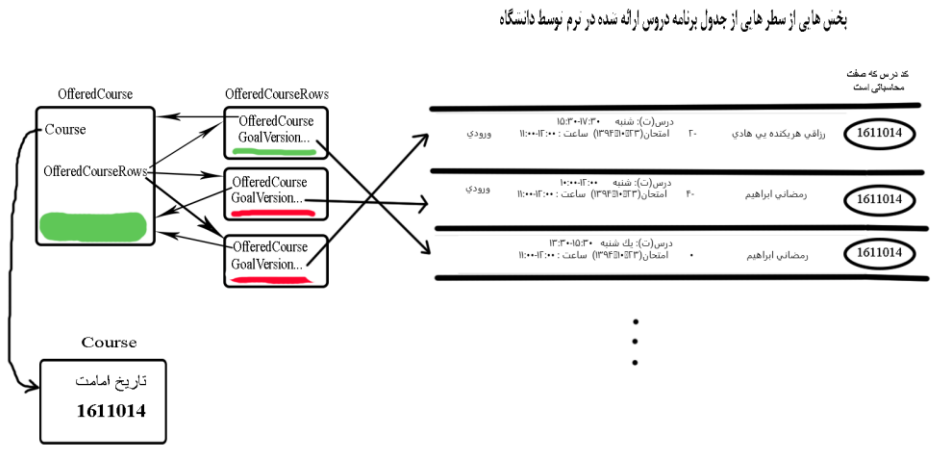
حال فرض کنید ما با پیش پردازش از یک نوع مدلسازی خاص استفاده کرده‌باشیم که در آن هر درس به همه سطرهای موجود در ترم آن دسترسی دارد(همه سطرهای جدول برنامه ترمی ارائه شده در ترم توسط دانشگاه که مربوط به آن درس است) و هر سطر هم به درسی که به آن ارجاع می‌کند، ارجاع دارد.

```
class OfferedCourse{
  Course Course;
  List<OfferedCourseRow> OfferedCourseRows;
  Color Color;
}
class OfferedCourseRow{
  OfferedCourse OfferedCourse;
  Main.OfferedCoursesRow GoalVersionOfferedCourseRow;
  Color Color;
}
```

در این مدلسازی OfferedCourse نماینده یک درس می‌باشد و دارای فیلدهای Course، OfferedCourseRows و Color است. Course نماینده یک درس می‌باشد(درسی مشابه آنچه از ساختار توصیف کننده برنامه درسی مصوب وزارت علوم می‌شناسیم)، OfferedCourseRows به سطرهایی ارجاع دارد که GoalVersionOfferedCourseRow آنها به سطری از جدول برنامه دروس ارائه شده در ترم توسط دانشگاه ارجاع داشته باشد که کد درس آن با کد درس Course در OfferedCourse یکسان باشد و Color همان سه رنگ سبز، سفید و قرمز می‌تواند باشد.

OfferedCourseRow نماینده یک سطر است که شامل OfferedCourse، GoalVersionOfferedCourseRow و Color می‌شود. خصوصیت OfferedCourse به نمونه ای از کلاس OfferedCourse که به این سطر اجاع دارد اشاره

می‌کند. GoalVersionOfferedCourseRow از نوع Main.OfferedCourseRow تعریف شده است. منظور ما از این نوع همان OfferedCourseRow ایست که در شبه-کدهای بخش های قبلی استفاده کردیم که نماینده یک سطر از جدول دروس ارائه شده در ترم توسط دانشگاه بود و بخاطر اینکه بتوانیم آنرا از نوع OfferedCourseRow ای که در اینجا تعریف کردیم تفکیک کنیم از این نوع نمایش استفاده شده است. Color نیز مانند قبل می‌تواند یکی از سه مقدار سبز، سفید و قرمز را داشته باشد.

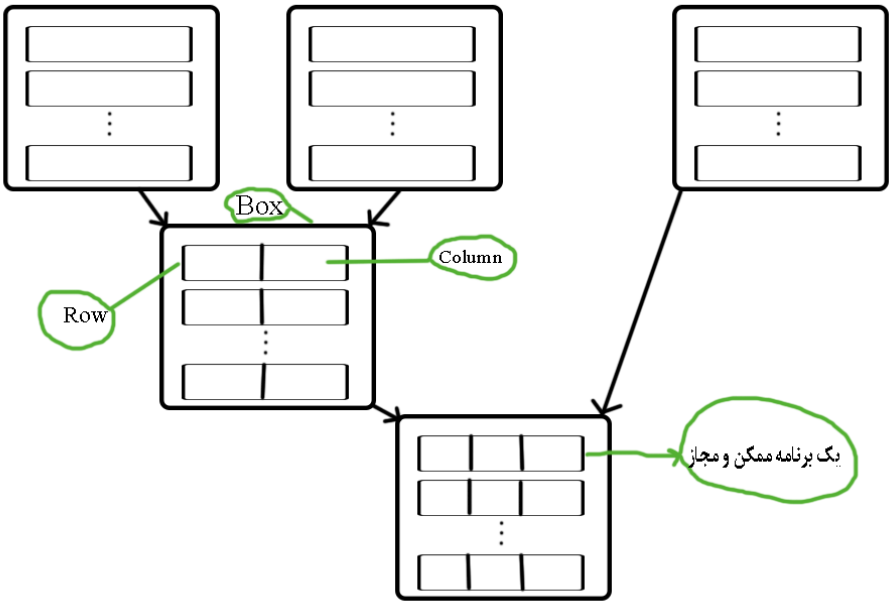


شکل ۲-۳ مدل گرافیکی نمونه‌ای از ارتباطات *Course*، *OfferedCourse*، *OfferedCourseRow* **و** *Main.OfferedCourseRow*

علاوه بر این ساختارها، دو ساختار دیگر نیز معرفی می‌کنیم که عبارتند از Box و Row و بصورت زیر قابل توصیف هستند:

```
class Row{
  List<OfferedCourseRow> Columns;
}
class Box{
  List<Row> Rows;
}
```

ما به ازای هر درس از مجموعه دروس ورودی یک Box می‌سازیم و سپس بصورت درخت مانند هر مرحله، دو به دو همه برنامه‌های ممکن و مجاز، یعنی آن Boxهایی که Columnهای آنها با هم تداخل زمانی ندارند را محاسبه می‌کنیم. این کار را تا بررسی آخرین Box با تعداد Column برابر یک ادامه می‌دهیم. اگر در طول محاسبات مرحله‌ای، به Box ای برخوردیم که طول Rows آن پس از انجام محاسبه صفر بود یعنی برنامه ممکن و مجازی برای این مجموعه دروس وجود ندارد و اگر هرگز به چنین Boxای برنخوردیم، آخرین Boxای که محاسبه می‌کنیم شامل تمام برنامه‌های مجاز و ممکن خواهد بود.



شکل ۳-۳ شیوه محاسبه کردن تمام برنامه‌های ممکن و مجاز برای یک مجموعه درس ورودی سه تایی

```
Box CreateBoxForOfferedCourse(OfferedCourse offeredCourse)
{
   Box box = new Box();

   offeredCourse.OfferedCourseRows.ForEach(r =>
   {
     if (r.Color != RED)
     {
        Row row = new Row();
        row.Columns.Add(r);
        box.Rows.Add(row);
     }
   });

   return box;
}
```

تابع CreateBoxForOfferedCourse یکی از دروس موجود در مجموعه دروس ورودی را بعنوان آرگومان گرفته Box متناظر با آن را ایجاد کرده، برمی‌گرداند.

```
List<Box> Validate(List<Box> boxes, bool examCollideCheck = false)
{
    if (boxes.Count == 1 && boxes[0].Rows.Count == 0)
       return null;
    List<Box> bxs = new List<Box>();
    bxs.Add(boxes[0]);

    bool con0 = true;
    for (int i = 1; i < boxes.Count; i++)
    {
      var b1 = bxs[0];
      var b2 = boxes[i];

      Box newBox = new Box();

      for (int j = 0; j < b2.Rows.Count; j++)
```

```
      {
        for (int k = 0; k < b1.Rows.Count; k++)
        {
           bool con = true;

           Row newRow = new Row();


           for (int m = 0; m < b1.Rows[k].Columns.Count; m++)
           {
             bool collide = DoTheyCollide(b1.Rows[k].Columns[m]
                                          .GoalVersionOfferedCourseRow
                                           .TimeAndSitesAndExam
                                            .TimeAndSites,
                                       b2.Rows[j].Columns[0]
                                          .GoalVersionOfferedCourseRow
                                           .TimeAndSitesAndExam
                                            .TimeAndSites);

             if (collide){
                con = false;
                break;
              }

             if (examCollideCheck &&
                 DoTheyExamCollide(b1.Rows[k].Columns[m]
                                    .GoalVersionOfferedCourseRow
                                      .TimeAndSitesAndExam.Exam,
                                b2.Rows[j].Columns[0]
                                    .GoalVersionOfferedCourseRow
                                       .TimeAndSitesAndExam.Exam)){
                   con = false;
                   break;
              }

             newRow.Columns.Add(b1.Rows[k].Columns[m]);
             }

             newRow.Columns.Add(b2.Rows[j].Columns[0]);

             if (con) newBox.Rows.Add(newRow);
          }
        }

        if (newBox.Rows.Count == 0){
          con0 = false;
          break;
        }
        else bxs[0] = newBox;
      }

      if (con0) return bxs;
      else return null;

}
```

تابع Validate بعنوان آرگومان یک لیست از Boxها را گرفته، با بررسی دو به دو آنها، در آخر اگر برنامه(های) ممکن و مجازی وجود داشته باشد، همه برنامه‌های ممکن و مجاز را محاسبه کرده برمی‌گرداند و درغیراینصورت null خروجی تابع خواهد بود.

```
Validate(List<OfferedCourse> offeredCoursesList, bool examCollideCheck = false)
{
  List<Box> lst = new List<Box>();
  for (int i = 0; i < offeredCoursesList.Count; i++)
    lst.Add(CreateBoxForOfferedCourse(offeredCoursesList[i]));
  return Validate(lst, examCollideCheck);
}
```

این تابع Validate، مجموعه دروسی که می‌خواستیم از اول این بحث، برنامه‌های ممکن و مجاز را برایشان محاسبه کنیم بعنوان

ورودی گرفته Boxهای متناظر آنها را ایجاد کرده در لیستی ذخیره می‌کند سپس با فراخوانی تابع Validateای که قبل تر بررسی‌اش کردیم که لیستی از Boxها را بعنوان ورودی می‌گیرد، همه برنامه-های ممکن و مجاز را در صورتیکه حداقل یک برنامه ممکن و مجاز وجود داشته باشد برمی‌گرداند و چنانچه چنین برنامه‌ای وجود نداشته باشد null بعنوان خروجی تابع در نظر گرفته می‌شود.

۲-۶-۴ بخش اصلی الگوریتم

الگوریتم پیشنهاد برنامه(های) ترمی بعنوان ورودی به ترتیب لیستی از امتیاز دروس،لیستی از درس‌های سفید، لیستی از درس-های سبز، حداقل واحد مجاز برای برنامه ترمی، حداکثر واحد مجاز برای برنامه ترمی، مجموع تعداد واحدهای درس‌های سبز، مجموع امتیازات دروس سفید، ساختار توصیف کننده برنامه ترمی مصوب وزارت علومی که با اطلاعات مناسب پر شده است، کارت اعتباری دانشجو، همه برنامه‌های ممکن و مجاز برای دروس سبز، حالت بررسی تصادم امتحانات و تایم اجرا الگوریتم را بعنوان ورودی دریافت می‌کند.

الگوریتم ابتدا از میان درس‌های سفید ورودی، بصورت تصادفی با احتمال **امتیاز درس ضرب در صد تقسیم بر مجموع امتیازات دروس سفید** درس‌هایی را جهت اخذ شدن انتخاب می‌کند. اگر مجموع واحدهای دروس انتخاب شده در این مرحله بعلاوه مجموع واحدهای دروس سبز، در محدوده مجاز حداقل و حداکثر واحد مجاز یک برنامه ترمی قرار نداشت این مرحله تکرار می‌شود. در غیر اینصورت الگوریتم چک می‌کند آیا با اخذ مجموعه دروسی شامل درس‌های سبز و درس‌های سفیدِ انتخابیِ فعلی، آیا ساختار در وضعیت معتبری قرار دارد یا خیر؟ اگر پاسخ منفی باشد الگوریتم مجدادا باید دنبال مجموعه دروس سفید جدیدی برای بررسی بگردد اما اگر پاسخ مثبت بود الگوریتم همه برنامه‌های ممکن و مجاز از مجموعه شامل دروس سبز و سفید انتخابی فعلی را در صورتیکه وجود داشته باشند، می‌یابد. پس از آن هر کدام از این برنامه‌ها به نمونه‌ای بنام _ChoosedWeeklyProgramManager، درخواستی برای قرار گرفتن در لیست نهایی می‌دهند. _ChoosedWeeklyProgramManager وظیفه دارد برای هر برنامه‌درخواستی ارزشی براساس تابع ارزیابی در نظر بگیرد. بعلاوه وظیفه دیگر آن گزینش تعداد مشخصی از بهترین برنامه‌های غیر تکراری یافته شده است. الگوریتم همه این تکرار ها را در مدت زمان مشخصی انجام می‌دهد و پس از آن، تعداد مشخصی از بهترین برنامه ترمی‌هایی را که یافته باز می‌گرداند.

```
Algo(double[] courseScore, List<OfferedCourse> whiteCourses, List<OfferedCourse> greenCourses,
int minUnits, int maxUnits, int greenCoursesUnits, double whiteCoursesTotalScore,Curriculum
curriculum, CreditCard studCredit,List<Box> GreenCoursesBoxes, bool examCollideCheck, int
timeout)
{
 var _ChoosedWeeklyProgramManager= new AlgorithmTopWeeklyProgramManager();

 Execute.For(timeout){

   Random random = new Random();
   int CurrentSelectedUnits=0;
   bool[] Selected;
```

```
   List<int> takenCoursesId = new List<int>();

   while (true){
    while (true){
     do{
        Selected = new bool[whiteCourses.Count];
        CurrentSelectedUnits = 0;
        takenCoursesId.Clear();

        //init and assigning values to Selected array
        for (int i = 0; i < Selected.Length; i++){

           var offeredCourse = whiteCourses[i];
           var course = whiteCourses[i].Course;

           var pc = random.Next(100);

           if (CurrentSelectedUnits + course.Units + greenCoursesUnits >
               maxUnits) continue;

           if (pc <= courseScore[course.Id] * 100.0 /
               whiteCoursesTotalScore){

                Selected[i] = true;
                CurrentSelectedUnits += course.Units;
                takenCoursesId.Add(course.Id);
           }
        }

      }while (minUnits > CurrentSelectedUnits + greenCoursesUnits);

      greenCourses.ForEach(gc => takenCoursesId.Insert(0, gc.Course.Id));
      //two step validation
      bool o = IsValidState(curriculum,studCredit,takenCoursesId);
      if (o) break;
    }

   List<Box> boxes = new List<Box>();
   GreenCoursesBoxes.ForEach(b => boxes.Add(b));
   for (int i = 0; i < Selected.Length; i++){
      if (Selected[i]){
         Box b = CreateBoxForOfferedCourse(whiteCourses[i]);
         boxes.Add(b);
      }
   }

   List<Box> res = Validate(boxes, examCollideCheck);

   if (res != null) _ChoosedWeeklyProgramManager.TryAddNewWeeklyProgram(res[0].Rows);

  }
 }
 return _ChoosedWeeklyProgramManager;
}
```

همانطور که دیدیم الگوریتم بالا ورودی‌های خاصی دارد که به نظر نمی‌رسد مهیا کردن همه آنها توسط مشتری(کسی که از الگوریتم استفاده می‌کند یا آن را فراخوانی می‌کند) بصورت مستقیم کار معقولی باشد. در واقع نیز قرار نیست همه این ورودی ها را مشتری تهیه کند بلکه وظیفه مشتری فقط تهیه ورودی‌های ضروری است، در این قسمت ما شبه‌کد دیگری ارائه می‌کنیم که در آن اطلاعات مورد نیاز برای الگوریتم، با توجه به ورودی‌های ضروری مهیا شده از طرف مشتری تولید می‌شود و سپس با فراخوانی آنچه در شبه‌کد قبلی دیدیم در یک بازه زمانی مشخص، سعی می‌کنیم تعدادی برنامه ترمی مناسب به کاربر پیشنهاد دهیم.

```
MainAlgo(List<OfferedCourse> inputs, Curriculum curriculum
        ,CreditCard studCredit, int minUnits, int maxUnits
        , bool examCollideCheck, int timeout){
```

```
double[] courseScore = new double[curriculum.Courses.Count];

double numberOfGreenUnits = 0;

List<OfferedCourse> greenCourses = new List<OfferedCourse>();

//white courses that have at least one white row
List<OfferedCourse> whiteCourses = new List<OfferedCourse>();

int greenUnits = 0;
for (int i = 0; i < inputs.Count; i++){
   var offeredCourse = inputs[i];

   if (offeredCourse.Color == Green){
      greenCourses.Add(offeredCourse);
      greenUnits += offeredCourse.Course.Units;
   }
   else if (offeredCourse.Color == WHITE){
       for (int r = 0; r < offeredCourse.OfferedCourseRows.Count; r++){
          var offeredCourseRow = offeredCourse.OfferedCourseRows[r];
          if (offeredCourseRow.Color == WHITE){
            whiteCourses.Add(offeredCourse);
            break;
          }
       }
    }

    var course = offeredCourse.Course;

    courseScore[course.Id] += (course.Units * 2 - 1);

    for (int c = 0; c < course.PrerequisiteCourses.Count; c++){
       var preCourse = course.PrerequisiteCourses[c];

       if (!preCourse.IsPassed && preCourse.IsAvailable()){
         if (preCourse.NumberOfFailed > 1)
             courseScore[preCourse.Id] += (course.Units * 2 - 1) * 0.20;
         else
             courseScore[preCourse.Id] += (course.Units * 2 - 1) * 0.25;
       }
     }

     for (int c = 0; c < course.RequisiteCourses.Count; c++){
         var reqCourse = course.RequisiteCourses[c];

         if (!reqCourse.IsPassed && reqCourse.IsAvailable())
            courseScore[reqCourse.Id] += (course.Units * 2 - 1) * 0.20;
      }

 }

 double whiteCoursesTotalScore = 0;

 whiteCourses.ForEach(i =>{whiteCoursesTotalScore +=
                      courseScore[i.Course.Id];});

 List<Box> greenCoursesBoxes = new List<Box>();

 greenCourses.ForEach(gc =>{
              numberOfGreenUnits += gc.Course.Units;
              Box b = CreateBoxForOfferedCourse(gc);
              greenCoursesBoxes.Add(b);
 });
 if (greenCoursesBoxes.Count > 0){
    List<Box> res = Validate(greenCoursesBoxes, examCollideCheck);
    if (res != null){
       greenCoursesBoxes.Clear();
       greenCoursesBoxes.Add(res[0]);
    }
 }
return  Algo(courseScore, whiteCourses, greenCourses, minUnits,
        maxUnits, greenUnits, whiteCoursesTotalScore, curriculum
        ,studCredit, greenCoursesBoxes, examCollideCheck,timeout);
```

}

امتیاز دادن به درس‌ها بسته به شرایط مسئله یا نظر طراح می‌تواند متفاوت باشد. ساده‌ترین راهکار اینست که امتیاز همه درس‌ها را یکسان درنظر بگیریم در اینصورت همه درس‌ها برای انتخاب شدن در برنامه شانس برابری خواهند داشت اما بدون توضیح اضافه این راهکار خوبی بنظر نمی‌رسد. در عوض استفاده از یک تابع اکتشافی جذابتر بنظر می‌رسد. تابع اکتشافی ما سعی می‌کند معیاری برای خوب بودن یک درس ارائه کرده و احتمال انتخاب شدن درس‌های بهتر را بیشتر کند و این درحالیست که برای درس‌های بدتر هم شانس انتخاب شدن قائل می‌شود. همانطور که از شبه‌کد هم قابل برداشت است ما امتیاز هر درس را به صورت زیر تعریف کردیم:

```
+ ( 1 - 2 * تعداد واحد درس )
( 0.25 * تعداد واحدهای دروس موجود پاس نشده‌ای که این درس در حال حاضر پیشنیاز آنها است )
( 0.20 * تعداد واحدهای دروس موجود پاس نشده‌ای که در حال حاضر این درس همنیاز آنها است ) +
```

در طراحی این تابع فرض ما این اینست که پیشنیازی و همنیازی دروس قانونی پایه‌ای است که برای همه برنامه‌های مصوب وزارت‌علوم خواه برنامه‌های فعلی باشد خواه برنامه‌های آینده ثابت و موجود است و به همین دلیل وابسته کردن فرمول بالا به این قانون، ایرادی ندارد. بعلاوه معیاری که ما برای امتیاز دهی در نظر گرفتیم مثل این است که بگوئیم **درسی بهتر است که تعداد واحدهای بیشتری داشته باشد و با اخذ یا پاس‌کردن آن بتوان تعداد دروس بیشتری را قابل اخذ کرد.** اما همانطور که گفته بودیم این فرمول بسته به سلیقه طراح می‌تواند متفاوت باشد.

تا به اینجا بخش‌های مهمی از الگوریتم را تشریح کردیم اما هنوز معیاری برای ارزش‌گذاری برنامه‌های ترمی ارائه نکردیم. در رابطه با این موضوع نیز باید گفت معیاری که برای این ارزش‌گذاری انتخاب می‌شود بسته به سلیقه طراح می‌تواند متفاوت باشد.

معیار اصلی‌ای که ما در اینجا از آن استفاده کردیم، حداقل فاصله بین کلاس‌ها است. البته این تنها معیاری نیست که در نظر گرفتیم چرا که در اینصورت برنامه‌هایی که تعداد واحد بیشتری دارند در مقایسه با برنامه‌های با تعداد واحد کمتر تقریبا هیچ شانسی برای گزینش پیدا نمی‌کنند و این درحالیست که کاربر حداقل و حداکثر واحدی را برای برنامه‌ترمی پیشنهادی مشخص می‌کند. بنابراین معیاری که از آن استفاده کردیم **حداقل فاصله بین کلاس‌ها به ازای هر واحد** است.

```
CalculateWeeklyProgramValue(List<OfferedCourseRow> lst)
{
   double res = 0;
   int units = 0;

   var rows = new SortedList<int,TimeAndSite>();
   foreach (var item in lst){

    if (item.GoalVersionOfferedCourseRow.TimeAndSitesAndExam
         != null && item.GoalVersionOfferedCourseRow
           .TimeAndSitesAndExam.TimeAndSites != null){

            foreach (var timeAndSite in
```

```
                        item.GoalVersionOfferedCourseRow.
                          TimeAndSitesAndExam.TimeAndSites){

                    rows.Add((((int)timeAndSite.Day) * 60 * 24 +
                              timeAndSite.StartTime.Hour * 60
                              + timeAndSite.StartTime.Minute)
                              , timeAndSite);
            }
    }
   units += item.OfferedCourse.Course.Units;
  }

  for (int i = 1; i < rows.Count; i++){
   int dt = (((int)rows.Values[i].Day) * 60 * 24 +
             rows.Values[i].StartTime.Hour * 60
             + rows.Values[i].StartTime.Minute) -
             (((int)rows.Values[i - 1].Day) * 60 * 24 +
             rows.Values[i - 1].FinishTime.Hour * 60
             + rows.Values[i - 1].FinishTime.Minute);
   res += dt;
  }
          return (res * -1.0 * 1.3) / units;
 }
```

هر چه ارزش محاسبه شده برای یک برنامه ترمی کمتر منفی باشد آن برنامه بهتر است در واقع اینطور در نظر گرفتیم که بهترین حالت وقتی است که همه زمان دانشجو در طول هفته برای خودش باشد.

۳-۷ ارائه ترتیبی قابل اخذ برای سطرهای یک برنامه ترمی

در فصل اول در رابطه با سیستم گلستان گفتیم ،در حال حاضر در هنگام انتخاب واحد در این سیستم شما باید واحد ها را به ترتیبی ثبت کنید که همواره برنامه ترمی شما در وضعیت معتبری قرار داشته باشد یعنی برای مثال جهت اخذ درسی که همنیازی دارد ابتدا باید همنیاز آن درس را وارد کرده سپس خود درس را وارد کنید. حالا که الگوریتم پیشنهاد برنامه ترمی را معرفی کردیم وقت آن رسیده بررسی کنیم چگونه می‌توانیم سطرهای یک برنامه ترمی را با ترتیبی قابل اخذ برای سیستم گلستان مرتب کنیم.

راه‌حل بسیار ساده است، تنها کافیست یک گراف بر مبنای پیشنیازی و همنیازی مجموعه دروس موجود در برنامه ترمی ایجاد کنید و سپس با اجرای الگوریتمی که مرتب سازی توپولوژیکی روی این گراف انجام دهد، شما به یک ترتیب قابل اخذ برای آن دروس دست خواهید یافت.

۳-۸ خلاصه

در این فصل بحث را با معرفی ساختار توصیف کننده برنامه درسی مصوب وزارت علوم شروع کردیم. پس از معرفی این ساختار نقض‌ها و چالش‌هایی که ساختار معرفی شده با آنها مواجه بود را بررسی کرده و خط مشی که در این خصوص اتخاذ کردیم را بیان نمودیم. در ادامه الگوریتم‌های مربوط به ایده کاستن را که در

فصل قبل با آنها آشنا شدیم شرح دادیم و در طول این کار با مفاهیمی مانند درس‌ها و سطرهای رنگی آشنا شدیم. بعد از این مرحله به الگوریتم اصلی پروژه خود یعنی الگوریتم پیشنهاد برنامه‌های ترمی رسیدیم و پس از تشریح زیرساخت‌های آن به تشریح خود الگوریتم پرداختیم. در آخر نیز راهکاری برای دستیابی به ترتیبی قابل اخذ برای دروس یک برنامه ترمی پیشنهاد دادیم.

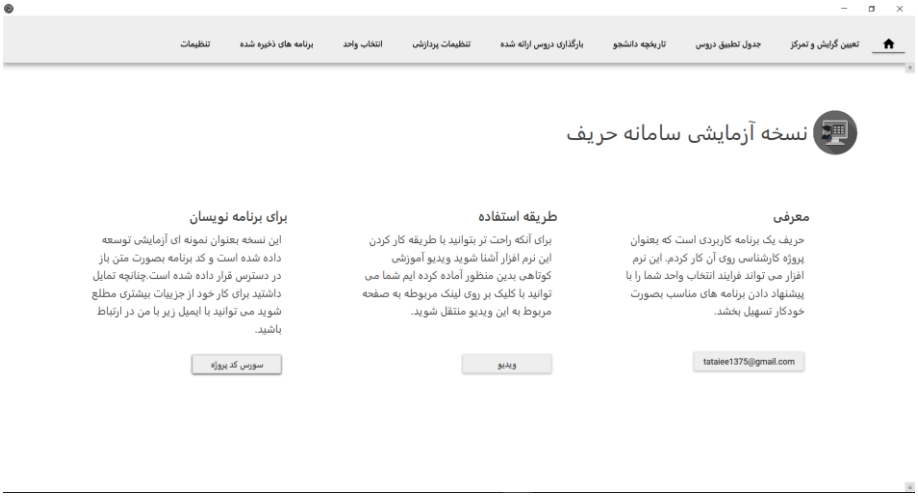
فصل چهارم:
نسخه آزمایشی و نتایج آزمایشات

۴-۱ مقدمه

در این فصل ابتدا شما را با یک نسخه آزمایشی توسعه داده شده بر مبنای برنامه درسی مصوب سال ۹۲ وزارت علوم برای رشته مهندسی کامپیوتر آشنا خواهیم کرد. سپس با اشاره به تعدادی از برنامه‌های ترمی پیشنهادی توسط نسخه آزمایشی که آنها را در طول آشنایی با این نسخه مطرح می‌کنیم، برخی از نکات و توضیحات را در رابطه با نتایج حاصل از آزمایشات بیان می‌کنیم.

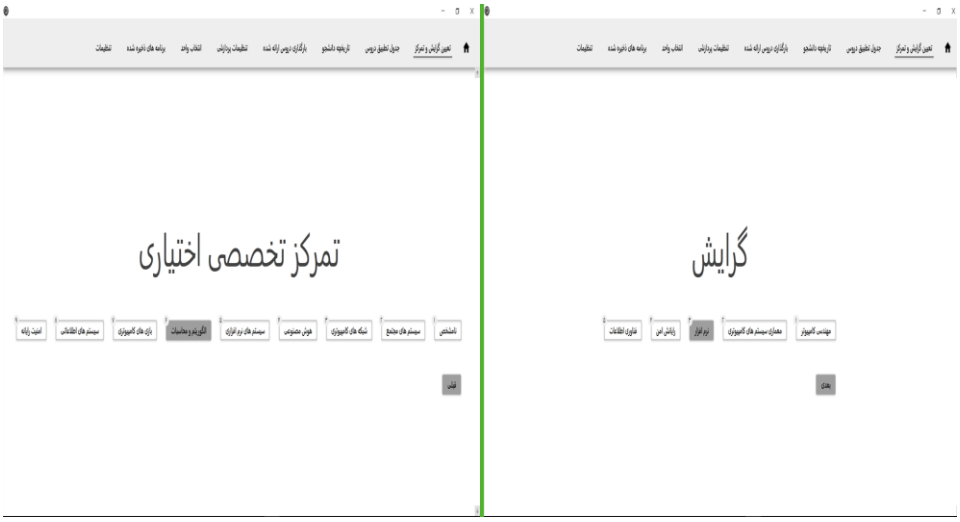
۴-۲ آشنایی با نسخه آزمایشی

نسخه آزمایشی، برنامه‌ای است که بعنوان نمونه با توجه به آنچه در فصل های قبل بیان کردیم توسعه داده شده است. این نسخه بر مبنای برنامه درسی سال ۹۲ مصوب وزارت علوم برای رشته مهندسی کامپیوتر با زبان سی شارپ پیاده‌سازی شده است. در ادامه شما را با این نسخه آزمایشی آشنا می‌کنیم.



شکل ٤-١. تصویری از صفحه خانه نرم‌افزار

برنامه بجز صفحه خانه که صرفا جنبه نمایشی دارد شامل بخش‌هایی بنام‌های تعیین گرایش و تمرکز، جدول تطبیق دروس، تاریخچه دانشجو، بارگذاری دروس ارائه شده، تنظیمات پردازشی، انتخاب واحد، برنامه‌های ذخیره شده و تنظیمات می‌باشد.



شکل ٤-٢ تصویری از قسمت‌های تعین گرایش و تمرکز تخصصی

در شکل ۴-۲ بخش تعیین گرایش و تمرکز نشان داده شده است. در تصویر سمت راست کاربر گرایش تخصصی نرم‌افزار و در تصویر سمت چپ کاربر تمرکز تخصصی الگوریتم و محاسبات را انتخاب کرده است.

شکل ٤-٣ تصویری از بخش جدول تطبیق دروس

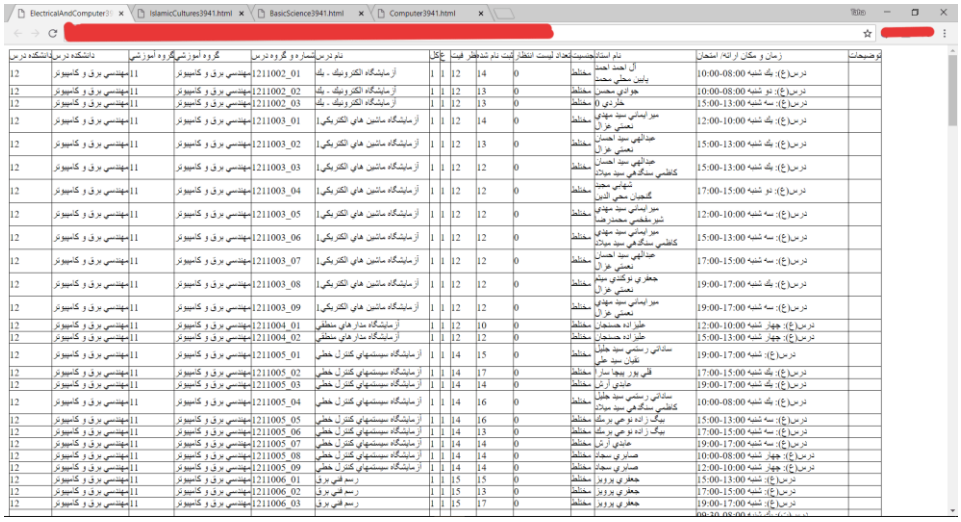
در شکل ۴-۳ بخش جدول تطبیق دروس نرم‌افزار نشان داده شده است و داده‌های آن بر اساس اطلاعات دانشگاه صنعتی نوشیروانی بابل پر شده است. خوب است به این نکته اشاره کنیم که جدول تطبیق دروس به تنهایی نمی‌تواند ضامن سازگاری این نسخه در دانشگاه‌های دیگر باشد چرا که همه دانشگاه‌ها خود را ملزم به رعایت پیشنیازها و همنیازهای برنامه درسی مصوب وزارت علوم نمی‌دانند بنابراین برای آنکه بتوانید این نسخه را در دانشگاهی که چنین شرایطی دارد استفاده کنید، باید داخل کد برنامه، قسمتی که ساختار توصیف کننده برنامه درسی مصوب وزارت علوم توصیف شده است، پیشنازها و همنیازها را بر اساس شرایط دانشگاه مورد نظر اصلاح کرد.

شکل ٤-٤ تصویری از بخش تاریخچه دانشجو

در بخش تاریخچه دانشجو درس‌های سبز، درس‌هایی هستند که دانشجو آنها را گذرانده است. درس‌های زرد به این معنی هستند که دروسی را که این درس پیشنیاز آنها است می‌توان با آن همنیاز کرد (مثلا دانشجویی که دوبار درسی را افتاده طبق قوانین فعلی مجاز به انجام چنین کاری می‌باشد) و سایر درس‌ها سفید هستند.

شکل ٤-٥ تصویری از بخش بارگذاری دروس ارئه شده

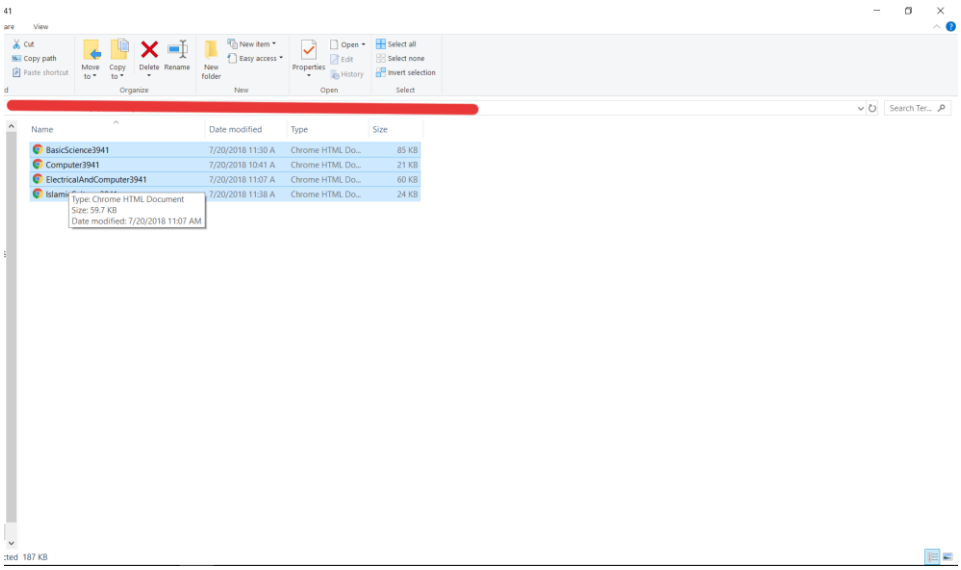
در بخش بارگذاری دروس ارائه شده، می‌توانید فایل‌های مربوط به برنامه دروس ارائه شده در ترم توسط دانشگاه را که در سیستم گلستان قابل دریافت است برای پردازش بارگذاری کنید.



دانشکده درس	دانشگاه درس	گروه آموزشی	گروه آموزشی	شماره گروه درس	نام درس	ظرفیت	ثبت نام شده	ظرفیت	تکمیل	تعداد لیست انتظار	جنسیت	نام استاد	زمان و مکان ارائه امتحان	توضیحات
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211002_01	آزمایشگاه الکترونیک ۱ - یک	1	12	14	1	0	مختلط	آل احمد احمد یاسین مختبی محمد	درس(ع): یک شنبه 10:00-08:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211002_02	آزمایشگاه الکترونیک ۱ - یک	1	12	13	1	0	مختلط	جوادی محسن	درس(ع): دو شنبه 10:00-08:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211002_03	آزمایشگاه الکترونیک ۱ - یک	1	12	13	1	0	مختلط	جلوزی فا	درس(ع): سه شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_01	آزمایشگاه ماشین های الکتریکی۱	1	12	14	1	0	مختلط	موایمانی سید مهدی نعمتی عزال	درس(ع): یک شنبه 12:00-10:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_02	آزمایشگاه ماشین های الکتریکی۱	1	12	13	1	0	مختلط	عبدالهی سید احسان نعمتی عزال	درس(ع): یک شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_03	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	عبدالهی سید احسان کاظمی سنگدهی سید میلاد	درس(ع): یک شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_04	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	شهابی محمد گنجیان محی الدین	درس(ع): دو شنبه 17:00-15:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_05	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	موایمانی سید مهدی شیرمظفری محمدرضا	درس(ع): سه شنبه 12:00-10:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_06	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	موایمانی سید مهدی کاظمی سنگدهی سید میلاد	درس(ع): سه شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_07	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	عبدالهی سید احسان نعمتی عزال	درس(ع): سه شنبه 17:00-15:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_08	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	جعفری نوکندی میثم نعمتی عزال	درس(ع): یک شنبه 19:00-17:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211003_09	آزمایشگاه ماشین های الکتریکی۱	1	12	12	1	0	مختلط	موایمانی سید مهدی نعمتی عزال	درس(ع): سه شنبه 19:00-17:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211004_01	آزمایشگاه مدار های منطقی	1	12	10	1	0	مختلط	علوانده هستیجان	درس(ع): چهار شنبه 12:00-10:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211004_02	آزمایشگاه مدار های منطقی	1	12	12	1	0	مختلط	علوانده هستیجان	درس(ع): چهار شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_01	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	سادائی رستمی سید جلیل نقیان سید علی	درس(ع): شنبه 19:00-17:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_02	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	قلی پور بیطا سارا	درس(ع): یک شنبه 17:00-15:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_03	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	هندی آرش	درس(ع): یک شنبه 19:00-17:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_04	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	سادائی رستمی سید جلیل کاظمی سنگدهی سید میلاد	درس(ع): یک شنبه 10:00-08:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_05	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	بیگ زاده نو عی برمک	درس(ع): سه شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_06	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	بیگ زاده نو عی برمک	درس(ع): سه شنبه 17:00-15:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_07	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	هندی آرش	درس(ع): سه شنبه 19:00-17:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_08	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	صابری سجاد	درس(ع): چهار شنبه 10:00-08:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211005_09	آزمایشگاه سیستمهای کنترل خطی	1	12	14	1	0	مختلط	صابری سجاد	درس(ع): چهار شنبه 12:00-10:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211006_01	رسم فنی برق	1	12	15	1	0	مختلط	جعفری پرویز	درس(ع): شنبه 15:00-13:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211006_02	رسم فنی برق	1	12	15	1	0	مختلط	جعفری پرویز	درس(ع): شنبه 17:00-15:00	
12	مهندسی برق و کامپیوتر	11	مهندسی برق و کامپیوتر	1211006_03	رسم فنی برق	1	12	15	1	0	مختلط	جعفری پرویز	درس(ع): شنبه 19:00-17:00	

شکل ٤-٦ تصویری از نمونه فایل‌های قابل دستیابی در سیستم گلستان

ابتدا باید برنامه‌های مورد نیاز خود را در سیستم خود بصورت فایل‌های اچ تی ام ال ای ذخیره کنید. برای مثال ما در اینجا چهار فایل مربوط به دروس ارائه شده در ترم توسط برق و کامپیوتر، کامپیوتر، علوم پایه و معارف اسلامی را ذخیره کردیم.



شکل ٤-٧ تصویری از فایل‌های ذخیره شده مربوط به دروس ارائه شده در ترم توسط دانشگاه

سپس همه فایل‌های مورد نیاز خود را انتخاب کرده با drag and drop آنها را به داخل شکل ابر مانند موجود در بخش بارگذاری دروس ارائه شده کشیده، رها کنید و در آخر بر روی گزینه تهیه فایل هدف کلیک کنید.

شکل ٤-٨ تصویری از بخش بارگذاری دروس ارائه شده پس از تهیه موفقیت آمیز فایل هدف

چنانچه فایل هدف با موفقیت تولید شود بخش کشیدن و رها کردن به رنگ سبز در می‌آید.

شکل ٤-٩تصویری از بخش تنظیمات پردازشی

در بخش تنظیمات پردازشی، یک فرم تعبیه شده است که طی آن برخی از اطلاعات مورد نیاز برای اجرای الگوریتم را مشخص می‌کند. در قسمت ترم می‌توانید مشخص کنید بعنوان دانشجوی ترم چندم می‌خواهید انتخاب واحد کنید. در قسمت حداقل و حداکثر واحد می‌توانید مشخص کنید برنامه‌هایی که می‌خواهید الگوریتم به شما پیشنهاد دهد، حداقل و حداکثر چند واحد باشد (اگر دقیقا تعداد خاصی واحد را مد نظر دارید کافیست حداقل و جداکثر واحد را روی همان واحد مد نظرتان تنظیم کنید). فیلد بعدی مشخص می‌کند برای اعتبارسنجی درس‌های سبز حداکثر چقدر زمان بر حسب میلی ثانیه در نظر گرفته شود. فیلد زمان پردازش الگوریتم پیشنهاد برنامه ترمی مشخص می‌کند پردازش الگوریتم بر حسب میلی ثانیه چقدر طول بکشد. فیلد حداکثر تعداد برنامه‌های پیشنهادی مشخص می‌کند الگوریتم حداکثر چند برنامه ترمی به شما پیشنهاد

دهد. گزینه فیلتر کردن سطرهای بدون ظرفیت مشخص می‌کند آیا سطرهایی که ظرفیت آنها پر شده است فیلتر شوند یا خیر. گزینه بررسی تداخل زمانی امتحانات مشخص می‌کند آیا علاوه بر تداخل نداشتن کلاس‌ها تداخل نداشتن زمان امتحانات نیز بررسی شود یا خیر. در آخر با توجه به جنسیت می‌توانیم سطرهایی که از نظر جنسیت برای شما مجاز نمی‌باشند را فیلتر کنیم.

شکل ٤-١٠ تصویری از بخش انتخاب واحد

در بخش انتخاب واحد درس‌هایی که امکان اخذ آنها را دارید به همراه سطرهایی که ممکن است آنها را اخذ نمایید نمایش داده می‌شود. نام این دروس به همراه کد آنها در سمت راست این بخش نمایش داده می‌شوند. بر روی هر درس که کلیک کنید، رنگ دکمه آن درس خاکستری می‌شود و سطرهای آن به نمایش در می‌آید.

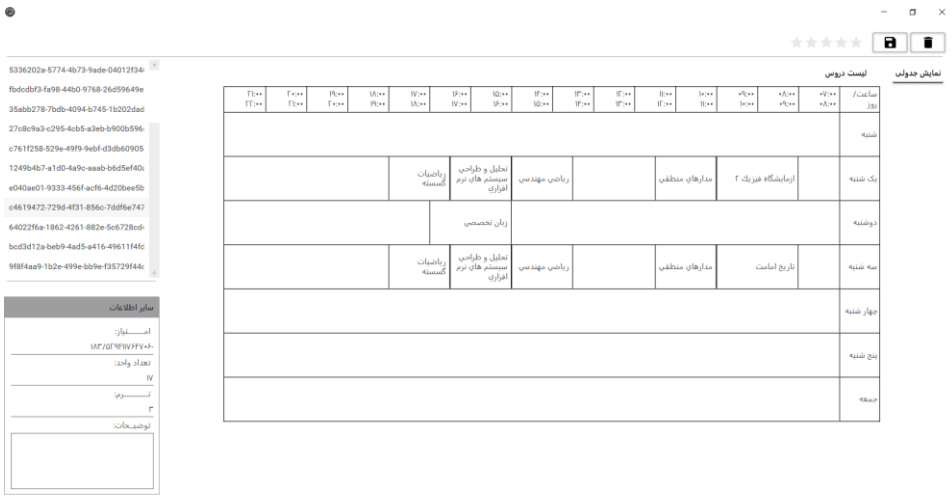
در بالای صفحه سمت راست، سه مربع قرار گرفته اند. با کلیک بر روی مربع اول از سمت راست می‌توانید مشخص کنید درس سبز باشد(در برنامه های پیشنهادی حتمی این درس وجود داشته باشد) یا خیر(بسته به شرایط درس بتواند اخذ شود یا نشود). با کلیک بر روی مربع قرمز رنگ می‌توانید همه سطرهای سفید رنگ یک درس غیر سبز را به رنگ قرمز درآورید. و با مربع کناری می‌توانید همه سطرهای قرمز یک درس را سفید کنید. علاوه بر این با دوباره کلیک بر روی هر سطر می‌توانید در صورت مجاز بودن، رنگ سطر را از سفید به قرمز یا از قرمز به سفید تغییر دهید. سطرهای قرمز به این معنی هستند که در برنامه‌های پیشنهادی الگوریتم نباید این سطرها حضور داشته باشند. در بخش مربوط به فیلتر زمانی می‌توانید رنگ کردن سطرها را با شیوه‌ای متفاوت انجام دهید. برای مثال فرض کنید بدلایلی مثلا کار نیمه وقت نمی‌خواهید در زمان‌های خاصی کلاس داشته باشید در اینصورت می‌توانید بازه‌های زمانی و روزهایی که نمی‌خواهید کلاس داشته باشید را در این بخش مشخص کرده و رنگ آنها را به کمک این ابزار قرمز کنید. اگر از بخش انتخاب واحد به بخش دیگری منتقل شوید یا نرم‌افزار را بسته مجددا اجرا کنید اطلاعات این بخش از دست خواهد رفت چنانچه می‌خواهید در مراجعات بعدی به این بخش اطلاعات شما

ذخیره بماند(تغییراتی که در رنگ سطرها و درس‌ها داده‌اید) می‌توانید بر روی گزینه موجود در قسمت ذخیره‌سازی کلیک کنید تا با این کار اطلاعات شما ذخیره شود. پس از خاتمه اجرا می‌توانید برنامه‌های پیشنهادی را با کلیک بر روی دکمه سمت چپ همین قسمت مشاهده کنید.

وقتی تنظیمات مورد نظر خود را انجام دادید، می‌توانید از قسمت پیشنهاد برنامه درسی، بر روی دکمه سمت راست آن کلیک کنید تا الگوریتم پیشنهاد برنامه‌های ترمی اجرا شود.

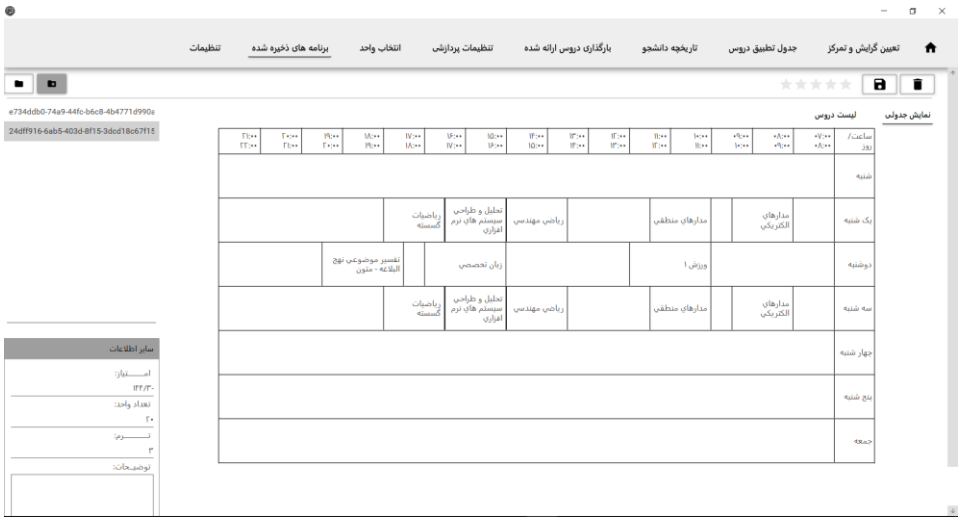
شکل ٤-١١تصویری از یک نمونه برنامه ترمی پیشنهاد شده توسط الگوریتم در نمایش لیست دروس

الگوریتم‌های پیشنهاد شده توسط الگورتیم با دو شیوه نمایش جدولی و لیست دروس قابل نمایش هستند. در شیوه لیست دروس، درس‌ها با ترتیبی قابل اخذ برای سیستم گلستان نمایش داده می‌شوند.



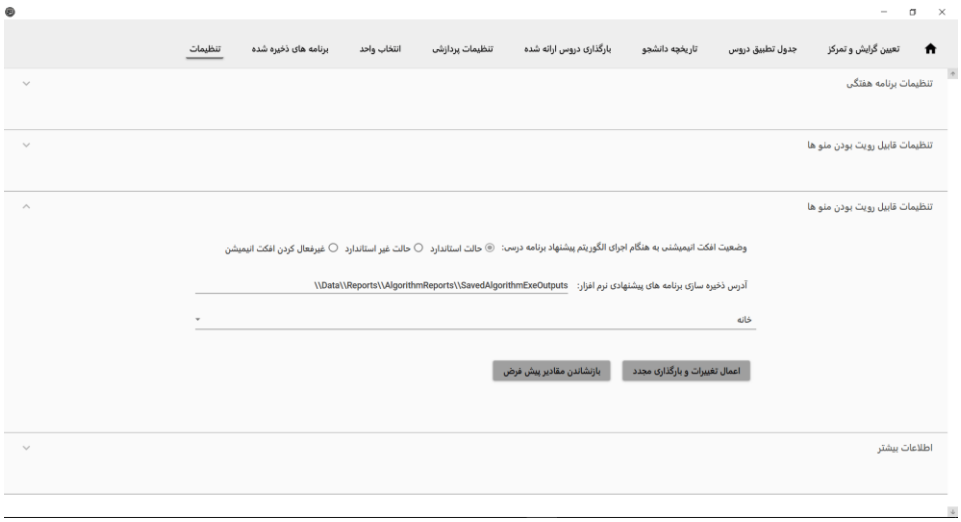
شکل ٤-١٢ تصویری از یک نمونه برنامه ترمی پیشنهاد شده توسط الگوریتم در نمایش جدولی

در شیوه نمایش جدولی هم، برنامه پیشنهادی در قالب جدول برنامه هفتگی به نمایش در می‌آید.



شکل ٤-١٣تصویری از بخش برنامه‌های ذخیره شده

در بخش برنامه‌های ذخیره شده، می‌توانید هم به برنامه‌های پیشنهادی در آخرین اجرای الگوریتم و هم به برنامه‌هایی که برای خود ذخیره کرده‌اید دسترسی پیدا کنید.



شکل ٤-١٤ تصویری از بخش تنظیمات

علاوه بر آنچه تا به اینجا گفته شد، این نسخه بخشی بنام تنظیمات دارد که از طریق آن می‌توانید تنظیمات مربوط به نرم-افزار را انجام دهید.

۴-۳ نتایج حاصل از آزمایشات

در شکل‌های ۴-۱۲ و ۴-۱۳ دو نمونه از برنامه‌های ترمی پیشنهادی توسط الگوریتم مشاهده می‌شوند. در اینجا نمونه‌های بیشتری را ارائه نمی‌کنیم اما در مورد برخی از نکات در رابطه با نمونه‌هایی که با آزمایش برای دوره تحصیلی یکی از دانشجویان در هر ترم بدست آمده، بحث می‌کنیم.

برنامه‌های پیشنهاد شده معمولا در عمل به گونه‌ای بودند که،

روز یا روزهایی در برنامه هفتگی آزاد می‌ماند(کلاسی نداشته باشند).

اغلب در عمل وقتی یک محدوده حداقل و حداکثر واحدی برای الگوریتم انتخاب می‌کنیم بیشتر جواب‌هایی که پیدا می‌شوند متمایل به یکی از واحدها هستند مثلا اگر حداقل و حداکثر واحد را به ترتیب ۱۷ و ۱۹ در نظر گرفته و تعداد برنامه پیشنهادی را ۱۵ در نظر بگیریم در ۱۵ برنامه پیشنهادی تعداد زیادی از برنامه‌ها مربوط به یکی از واحدها مثلا ۱۸ هستند و برنامه‌های پیشنهادی از سایر واحدها در اقلیت قرار می‌گیرند. در فصل قبل وقتی معیاری را برای ارزیابی ارزش هر برنامه ترمی معرفی می‌کردیم خاطر نشان کردیم که می‌خواهیم برنامه‌های پیشنهادی صرفا برای یک واحد خاص نباشند که این امر محقق شد و همینکه تعدادی اقلیت از واحدهای دیگر هم در برنامه‌ها می‌آیند نشانگر همین موضوع است. با این حال این اقلیت بودن خود شک برانگیز است چرا که این سوال را پیش می‌آورد که در اکثریت چشمگیر قرار گرفتن یکی از واحدها در برنامه‌های پیشنهادی به چه دلیل است؟ بنظر می‌رسد پاسخ را باید در سه احتمال جستجو کرد. اول آنکه ممکن است معیار ارزیابی ارائه شده موجب این امر شده باشد. دوم اینکه ممکن است خود الگوریتم و شیوه گزینش آن موجب چنین امری شود. سوم این احتمال وجود دارد که در عمل محدودیت‌هایی که توسط برنامه دروس ارائه شده توسط دانشگاه در ترم و شرایط دانشجو اعمال می‌شوند واقعا ما را در وضعیتی قرار می‌دهند که تعداد بخصوصی از واحد برای اخذ در آن ترم مناسب تر باشد. ممکن هم هست که ترکیبی از این موارد باعث چنین امری شده باشند. به هر حال تا به اینجا من نتوانسته‌ام نتیجه گیری کنم که دلیل این اتفاق چیست و آیا اصلا این خوب است یا نه.

علاوه بر آنچه تا به اینجا گفته شد خوب است یادآور شویم که الگوریتم پیشنهاد برنامه‌های ترمی براساس معیاری که طراح آن را مناسب می‌دانسته توسعه یافته ولی شکی نیست که دیدگاه‌ها و سلایق مختلف می‌توانند معیارهای دیگری را برای یک برنامه، خوب بدانند. در آخر ارزیابی من از برنامه‌های پیشنهادی ایسنت که فکر می‌کنم براساس معیارهایی که من برای یک برنامه ترمی خوب می‌دانم، الگوریتم برنامه‌های بسیار خوب و همچنین متنوعی را در نتیجه آزمایشات انجام شده ارائه داده است.

۴-۴ خلاصه

در این فصل ابتدا با یک نسخه آزمایشی توسعه داده شده آشنا شدیم و بخش‌های مختلف آنرا به همراه چگونگی بکار بردنشان توضیح دادیم. سپس به تشریح نتایجی که از برنامه‌های پیشنهاد شده توسط الگوریتم بدست آمده بودند پرداختیم.

فصل پنجم:
نتیجه گیری‌ها و پیشنهادها

۵-۱ نتیجه گیری‌ها

در این پروژه پس از مطالعه برنامه‌های درسی مصوب وزارت علوم به تحلیل و طراحی جهت ارائه راهکاری عملی و کارا برای پیشنهاد برنامه‌های ترمی جهت انتخاب واحد دانشجویان مبادرت گردید. سپس نسخه‌ای آزمایشی در همین جهت پیاده‌سازی شد که در عمل توانست برنامه‌های ترمی مناسبی را در زمانی کوتاه و قابل تنظیم ارائه دهد. سعی شد تا قسمت تحلیل و طراحی اولیه، بسیار دقیق و جامع باشد که فکر می‌کنم این امر محقق شد با این وجود اما ساختاری که می‌خواستم بعنوان توصیف کننده برنامه‌های درسی ارائه دهم در حالت کلی شکست خورد و ساختار به ساختار توصیف کننده برنامه‌های درسی وزارت علوم محدود شد. اما در کل فکر می‌کنم پروژه، خوب از آب در آمد و علیرغم ضعف‌هایی که داشت توانست آنچه به عنوان خروجی از آن انتظار می‌رفت، بخوبی برآورده کند.

۵-۲ پیشنهادها

برخی از پیشنهادهایی که برای ادامه این پروژه می‌توان ارائه داد را در ادامه لیست کرده‌ایم:

- ارائه ساختار توصیف کننده برنامه‌های درسی برای حالت عمومی و یا اصلاح همین ساختار فعلی
- افزودن سایر الویت‌های باید و شاید
- گنجاندن امکان اصلاح پیشنیازها و همنیازهای دروس در برنامه کاربردی
- توسعه یک برنامه کاربردی که تولید ساختار توصیف کننده برنامه‌های درسی را بصورت ویژوال تسهیل نماید.
- اضافه کردن امکانی جهت افزودن اسکریپت‌ها یا تکه برنامه‌ها در برنامه کاربردی با این ایده که هر شخص بتواند معیارهای خود در مورد برنامه ترمی خوب، توصیف کند.
- اضافه کردن امکان انتخاب واحد خودکار که در آن دانشجو برنامه‌ای از برنامه‌های پیشنهادی را انتخاب کند و برنامه کاربردی بصورت خودکار پس از تائید توسط دانشجو آن برنامه را در سیستم برایش اخذ نماید.

پیوست‌ها

جمهوری اسلامی ایران

وزارت علوم، تحقیقات و فناوری

برنامه درسی

(بازنگری شده)

مقطع کارشناسی

مهندسی کامپیوتر

با ۴ گرایش: معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات

گروه فنی و مهندسی

کمیته کامپیوتر

مصوبه هشتصد و بیست و هشتمین جلسه شورای برنامه‌ریزی آموزش عالی

وزارت علوم، تحقیقات و فناوری مورخ ۹۲/۲/۸

بسم الله الرّحمن الرّحیم

برنامه درسی مقطع کارشناسی رشته مهندسی کامپیوتر

گروه: فنی و مهندسی — کمیته تخصصی: مهندسی کامپیوتر

رشته: مهندسی کامپیوتر — گرایش: معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات

مقطع: کارشناسی — کد رشته:

شورای برنامه‌ریزی آموزش عالی، در هشتصد و بیست و هشتمین جلسه مورخ ۹۲/۲/۸، برنامه درسی بازنگری شده مقطع کارشناسی رشته مهندسی کامپیوتر با ٤ گرایش معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات را به شرح زیر تصویب کرد:

ماده ۱: برنامه درسی بازنگری شده مقطع کارشناسی رشته مهندسی کامپیوتر با ٤ گرایش معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات از تاریخ تصویب برای کلیه دانشگاهها و مؤسسات آموزش عالی کشور که مشخصات زیر را دارند، لازم‌الاجراء است:

الف) دانشگاهها و مؤسسات آموزش عالی که زیر نظر وزارت علوم، تحقیقات و فنّاوری اداره می‌شوند.

ب) مؤسساتی که با اجازه رسمی وزارت علوم، تحقیقات و فنّاوری و بر اساس قوانین تأسیس می‌شوند و تابع مصوبات شورای گسترش آموزش عالی هستند.

ماده ۲: این برنامه بازنگری شده از تاریخ ۹۲/۲/۸ جایگزین برنامه درسی مقطع کارشناسی رشته مهندسی کامپیوتر با ٤ گرایش معماری سیستم های نرم‌افزاری، هوش مصنوعی، معماری کامپیوتر، نظریه‌های محاسبات و الگوریتم مصوب ۸۰یصد و هشتادمین جلسه شورای برنامه‌ریزی آموزش عالی مورخ ۷۸/۶/۲۸ شد و برای دانشجویانی که از این تاریخ به بعد وارد دانشگاه می‌شوند، لازم‌الاجراء است.

ماده ۳: برنامه درسی بازنگری شده مقطع کارشناسی رشته مهندسی کامپیوتر با ٤ گرایش معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات در سه فصل: مشخصات کلی، جداول دروس و سرفصل دروس برای اجراء به دانشگاهها و مؤسسات آموزش عالی ابلاغ می‌شود.

رأی صادرهٔ هشتصد و بیست و هشتمین جلسه شورای برنامه‌ریزی آموزش عالی مورخ ۹۲/۲/۸ درخصوص برنامه درسی بازنگری شده مقطع کارشناسی رشته مهندسی کامپیوتر با ٤ گرایش معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات:

۱. برنامه درسی بازنگری شده مقطع کارشناسی رشته مهندسی کامپیوتر با ۴ گرایش معماری سیستم های کامپیوتری، نرم‌افزار، رایانش امن و فناوری اطلاعات که از سوی کمیته تخصصی مهندسی کامپیوتر پیشنهاد شده بود، با اکثریت آراء به تصویب رسید.

۲. این برنامه از تاریخ تصویب به مدت پنج سال قابل اجراء است و پس از آن نیازمند بازنگری است.

سعید قدیمی
دبیر شورای برنامه‌ریزی آموزش عالی

حسین نادری مقش
نایب رئیس شورای برنامه‌ریزی آموزش عالی

فصل اول

مشخصات کلی

دوره کارشناسی مهندسی کامپیوتر

مقدمه

۱. تعریف و اهداف

هدف از طراحی این دوره آموزشی تربیت انسانی است خود اتکا خودباور، مسلط به فناوری، معتقد به اینکه ماشین باید در خدمت و به فرمان انسانیت و ارزش‌های اسلامی-انسانی جامعه باشد؛ دارای غرور و خود باوری ملی؛ خود را هم‌سطح جوامع دیگر و با تلاش برای کسب ارزش‌های والای اخلاقی و اسلامی دارای قابلیت پیشتازی و هدایت‌گری می‌داند؛ معتقد به همکاری بین‌المللی است و نه برده‌گی، تابعیت و دنبال روی؛ معتقد و به دنبال رویه فنی اجرای بدیع و نوآورانه مأموریت‌های محوله؛ نوآور و پیشتاز در عرصه اقتصادی؛ معتقد به کسب اقتدار و قدرت و ثروت با تلاش و شایستگی و برتری اخلاقی-کاری-علمی؛ پیشتاز و الهام‌بخش و متعامل با دنیای اطراف برای کسب و پخش خیر کثیر علم و نیکی می‌باشد. دست‌یابی به این مهم از طریق تربیت مهندسین توانمند و آشنا و بلکه مسلط به اصول و پایه‌های علمی صورت می‌گیرد که با زمینه‌های کاربردی و مهندسی آشنایی نظری و تئوری لازم را با ترکیب متناسب فکر و عمل یافته‌اند.

با توجه به سرعت بالای تحولات علمی و توسعه فناوری در عرصه های مرتبط با مهندسی کامپیوتر و تاثیر مستقیم آن در همه رشته های علمی دیگر و در زندگی انسان امروز، این ضرورت احساس می‌شد که به روز شدن شکل و قالب دوره و تجدید نظر در درسها و محتوای آنها صورت پذیرد تا بدینوسیله هم رشته مهندسی کامپیوتر و هم تمامی رشته ها و عرصه های کاری صنعتی و خدماتی در کشور از این تحولات سریع به صورت نهادینه و نظام مند بهره مند گردند. در این بازنگری ضمن مراجعه و ارزیابی مقایسه ای برنامه های دانشگاههای معتبر دنیا از نظرات و مشورتهای بیش از صد نفر از اساتید متخصص زمینه های گوناگون مهندسی کامپیوتر و صاحب‌نظران صنعتی در کشور به صورت مستقیم بهره گرفته شده است و در عین حال این عزیزان همکار با سایر متخصصان و اساتید دانشگاههای سرتاسر ایران مشورت و نظرخواهی نموده اند. در تدوین این برنامه ضمن حرکت پایایای با

تحولات روز دنیا، جنبه های کاربردی و شکل گیری تفکر و نگاه نقادانه و مبتکرانه در میان دانش آموختگان و آماده سازی آنها برای راهبری بازارهای ملی با نگاه رقابت پذیری جهانی مد نظر قرار گرفته است.

در جریان آموزش‌های دوره دانشجویان با اصول و مبانی و کاربردهای مهندسی کامپیوتر و فناوری اطلاعات و با دانش و فناوری روز مرتبط با سیستم‌های کامپیوتری و سطح طراحی آنها آشنا می‌گردند و بر سطوح پیاده‌سازی، پشتیبانی و بهینه‌سازی سیستم‌های مهندسی کامپیوتری مورد نیاز جامعه مسلط می‌شوند و آمادگی برای انجام پژوهش و کسب قابلیت طراحی سیستم‌های جدید در دوره‌های تحصیلات تکمیلی را حاصل می‌کنند.

۲. طول دوره و شکل نظام

طول دوره و شکل نظام مطابق آیین نامه های مصوب وزارت علوم تحقیقات و فناوری می باشد. دوره کارشناسی مهندسی کامپیوتر دارای ۴ گرایش است. لازم است در طول دوره دانشجویان ۲۲ واحد درسهای عمومی، ۳۰ واحد درسهای پایه، ۵۹ واحد درسهای اصلی رشته مهندسی کامپیوتر و ۳۱ واحد درسهای تخصصی خود را در یکی از گرایش‌ها (و تمرکزهای مجاز تعریف شده در زیر) اختیار کنند و با اخذ ۸ واحد اختیاری مدرک کارشناسی مهندسی کامپیوتر که حداکثر با نام چهار گرایش همراه می‌گردند به ایشان اعطا می‌شود. این گرایش‌ها عبارتند از:

- معماری سیستم‌های کامپیوتری (۱۹ واحد تخصصی بعلاوه ۱۲ واحد از یکی از پنج تمرکز تخصصی سیستم‌های مجتمع، شبکه‌های کامپیوتری، هوش مصنوعی، بازی‌های کامپیوتری و امنیت رایانه و ۸ واحد از فهرست درسهای اختیاری با رعایت پیشنیاز)
- نرم‌افزار (۱۹ واحد تخصصی بعلاوه ۱۲ واحد از یکی از هفت تمرکز تخصصی سیستم‌های اطلاعاتی، الگوریتم و محاسبات، سیستم‌های نرم‌افزاری، امنیت رایانه، بازی‌های کامپیوتری، هوش مصنوعی و شبکه‌های کامپیوتری و ۸ واحد از فهرست درسهای اختیاری با رعایت پیشنیاز)
- رایانش امن (۳۱ واحد تخصصی بعلاوه ۸ واحد از فهرست درسهای اختیاری با رعایت پیشنیاز) این گرایش فعلا فقط در دانشگاه‌های بند ز ماده پنجاه قانون برنامه چهارم توسعه قابل عرضه است.
- فناوری اطلاعات (۳۱ واحد تخصصی بعلاوه ۸ واحد از فهرست درسهای اختیاری با رعایت پیشنیاز)

دانشجویان گرایش‌های معماری سیستم‌های کامپیوتری و نرم‌افزار می‌توانند ضمن گذراندن ۱۹ واحد تخصصی یکی از بسته‌های تمرکز تخصصی اختیاری ۱۲ واحدی را، که با توجه به قابلیتهای آموزشی، تخصص اساتید و اولویتهای بومی دانشکده محل تحصیل اجرای آن توسط دانشکده تصویب و عرضه می‌گردد را اخذ نمایند. دانشکده‌ها لازم است برای ارائه گرایش معماری سیستم های کامپیوتری و گرایش نرم‌افزار دوره مهندسی کامپیوتر حداقل یک تمرکز مجاز برای هر گرایش را عرضه نمایند و همچنین در صورت ارائه تمرکزهای متعدد مجاز هستند با توجه به امکانات خود برای ورود به هر تمرکز یک ظرفیت حداکثر تعیین نمایند. در برخی موارد ورود به یک هسته‌ی تمرکز ممکن است یک یا دو درس پیشنیاز لازم داشته باشد که لازم است از سهمیه باقیمانده درسهای اختیاری توسط دانشجو اخذ گردد. ۸ واحد باقیمانده درسهای اختیاری است که در میان آن‌ها اخذ دو واحد آزمایشگاه یا کارگاه با نظر دانشکده الزامی است. درسهای اختیاری می‌توانند از جدول درسهای اختیاری و از جمله از جدولهای درسهای تخصصی سایر گرایش‌ها و تمرکزها با رعایت درسهای پیشنیاز انتخاب گردند. بنا بر تشخیص دانشگاه محل پذیرش، دانشجویان ورودی دوره می‌توانند با گرایشهای جداگانه از کنکور ورودی انتخاب شوند و یا با عنوان دوره کارشناسی مهندسی کامپیوتر وارد گردند و پس از طی دو سال با توجه به تمایل دانشجویان و اولویت معدل تحصیلی آنان و ظرفیت گرایشها در دانشکده تعیین گرایش گردند. بسته‌های تمرکز تخصصی ۱۲ واحدی همگام با تحولات علم و فناوری می‌توانند به صورت پیوسته با پیشنهاد یک

دانشگاه و تصویب کمیته برنامه‌ریزی مهندسی کامپیوتر اضافه یا حذف یا اصلاح گردند و چون عنوان بسته‌های تمرکز تخصصی در عنوان مدرک کارشناسی ذکر نمی‌شود امکان دارد که به صورت سریع‌تری همگام با تحولات روز بهینه و اصلاح گردند. پس با این تعبیر تعداد درسهای اختیاری برخی گرایش‌های دوره کارشناسی مهندسی کامپیوتر به ۲۰ واحد ارتقا یافته است. بسته‌های تمرکز تخصصی اختیاری عبارتند از:

عناوین تمرکزهای تخصصی اختیاری:

۱. سیستم‌های مجتمع
۲. شبکه‌های کامپیوتری
۳. هوش مصنوعی
۴. امنیت رایانه
۵. بازی‌های کامپیوتری
۶. سیستم‌های نرم‌افزاری
۷. الگوریتم و محاسبات
۸. سیستم‌های اطلاعاتی

۳. واحدهای درسی

درسهای عمومی	۲۲ واحد
درسهای پایه	۲۰ واحد
درسهای اصلی	۵۹ واحد
درسهای تخصصی گرایش‌های معماری سیستم‌های کامپیوتری و نرم‌افزار	۱۹ واحد
درسهای تمرکزهای تخصصی اختیاری گرایش‌های معماری سیستم‌های کامپیوتری و نرم‌افزار	۱۲ واحد
درسهای تخصصی گرایش‌های فناوری اطلاعات و رایانش امن	۳۱ واحد
درسهای اختیاری	۸ واحد
جمع	۱۴۰ واحد

در جدول بالا مجموع درس های تخصصی گرایش های معماری سیستم های کامپیوتری و نرم افزار به علاوه درس های تمرکز های تخصصی این گرایش ها ۳۱ واحد است که معادل درس های تخصصی گرایش های فناوری اطلاعات و رایانش امن می باشد و در هریک از چهار گرایش تعریف شده مجموع کل واحد ها ۱۴۰ واحد است.

۴. توانمندی‌ها و قابلیتهای دانش آموختگان

۴.۱. توانمندیها

مهندس فارغ‌التحصیل رشته مهندسی کامپیوتر علاوه بر نگاه قوی تحلیلی و سیستمی و تسلط به مبانی علمی و فناوری روز دارای دید و تجربه عملی و تخصص کارگاهی و آزمایشگاهی و مهارت کاربردی برای زمینه‌های زیر است:

- آشنا و مسلط به اصول سیستمی، معماری، امنیتی سخت‌افزاری و نرم‌افزاری طراحی و بکارگیری سیستم‌های مهندسی کامپیوتری مدرن، مسلط در به‌کارگیری علمی یک زبان خارجی، آشنا با روش جستجو و بهره‌برداری از تازه‌ها و تحولات علم و فناوری، مسلط به دانش تحلیلی ریاضی و فیزیک مدرن و ریاضیات گسسته، مسلط به استفاده از زبان‌های برنامه‌نویسی و توصیف سخت‌افزار و سیستم دیجیتال، ساختارها و الگوریتم‌های ذخیره، بازیابی و به‌روزسازی ساختمان‌های داده، اصول سیستم‌های هوشمند، اصول مدارها و سیستم‌های پردازش و ذخیره الکترونیکی، اصول طراحی، برنامه‌نویسی، امنیت و بکارگیری شبکه‌های مخابراتی و کامپیوتری، امنیت داده‌ها و اطلاعات، طراحی سیستم‌های نهفته بهنگام و طراحی سیستم‌های قابل اطمینان کامپیوتری.
- همچنین در ابعاد کاربردی قادر به طرح سیستم‌ها و انتخاب سخت‌افزار و نرم‌افزار و راه‌اندازی سرورهای کامپیوتری شبکه‌ای برای کاربردهای اداری، آموزشی، اقتصادی، مالی، بهداشتی و دفاعی، طراحی و راه‌اندازی سخت‌افزار و نرم‌افزارهای بردهای کامپیوتری برای کاربردهای خاص نظیر اتوماسیون صنعتی، رباتیک، کنترل تردد، کنترل فرآیندهای صنعتی، سیستم‌های تصویربرداری صنعتی و پزشکی و ذخیره، پردازش و انتقال امن داده‌ها، طراحی و راه‌اندازی شبکه‌های بی‌سیم و بی‌سیم امن و مطمئن برای تبادل داده‌های چند رسانه‌ای، طراح سخت‌افزارهای برنامه‌پذیر و مدارهای مجتمع برای سیستم‌های کامپیوتری و طراحی نرم‌افزارهای مورد نیاز آنها، لحاظ کردن ملاحظات امنیت سیستم و شبکه و طراحی متناسب با آن.

۲-۱ درسهای عمومی (۲۲ واحد)

دروس عمومی و معارف اسلامی				
ردیف	گرایش	نام درس	تعداد واحد	ساعات تدریس
۱	مبانی نظری اسلام	اندیشه اسلامی ۱ (مبدأ و معاد)	۲	۳۲
		اندیشه اسلامی ۲ (نبوت و امامت)	۲	۳۲
		انسان در اسلام	۲	۳۲
		حقوق اجتماعی و سیاسی در اسلام	۲	۳۲
۲	اخلاق اسلامی	فلسفه اخلاق (با تکیه بر مباحث تربیتی)	۲	۳۲
		اخلاق اسلامی (مبانی و مفاهیم)	۲	۳۲
		آیین زندگی (اخلاق کاربردی)	۲	۳۲
		عرفان عملی اسلامی	۲	۳۲
۳	انقلاب اسلامی	انقلاب اسلامی ایران	۲	۳۲
		آشنایی با قانون اساسی جمهوری اسلامی ایران	۲	۳۲
		اندیشه سیاسی امام خمینی«ره»	۲	۳۲
۴	تاریخ و تمدن اسلامی	تاریخ فرهنگ و تمدن اسلامی	۲	۳۲
		تاریخ تحلیلی صدر اسلام	۲	۳۲
		تاریخ امامت	۲	۳۲
۵	آشنایی با منابع اسلامی	تفسیر موضوعی قرآن	۲	۳۲
		تفسیر موضوعی نهج‌البلاغه	۲	۳۲
۶	-	زبان فارسی	۳	۴۸
۷	-	زبان انگلیسی	۳	۴۸
۸	-	تربیت بدنی ۱	۱	۳۲
۹	-	تربیت بدنی ۲	۱	۳۲
۱۰	-	دانش خانواده و جمعیت	۲	۳۲
		جمع کل واحدهای عمومی	۲۲	

- دو درس به ارزش ۴ واحد از مجموعهٔ درسهای مبانی نظری اسلام
- یک درس به ارزش ۲ واحد از مجموعهٔ درسهای اخلاق اسلامی
- یک درس به ارزش ۲ واحد از مجموعهٔ درسهای انقلاب اسلامی
- یک درس به ارزش ۲ واحد از مجموعه درسهای تاریخ تمدن اسلامی
- یک درس به ارزش ۲ واحد از مجموعه درسهای آشنایی با منابع اسلامی

۹

۲-۱ درسهای عمومی (۲۲ واحد)

دروس عمومی و معارف اسلامی				
ردیف	گرایش	نام درس	تعداد واحد	ساعات تدریس
۱	مبانی نظری اسلام	اندیشه اسلامی ۱ (مبدأ و معاد)	۲	۳۲
		اندیشه اسلامی ۲ (نبوت و امامت)	۲	۳۲
		انسان در اسلام	۲	۳۲
		حقوق اجتماعی و سیاسی در اسلام	۲	۳۲
۲	اخلاق اسلامی	فلسفه اخلاق (با تکیه بر مباحث تربیتی)	۲	۳۲
		اخلاق اسلامی (مبانی و مفاهیم)	۲	۳۲
		آیین زندگی (اخلاق کاربردی)	۲	۳۲
		عرفان عملی اسلامی	۲	۳۲
۳	انقلاب اسلامی	انقلاب اسلامی ایران	۲	۳۲
		آشنایی با قانون اساسی جمهوری اسلامی ایران	۲	۳۲
		اندیشه سیاسی امام خمینی«ره»	۲	۳۲
۴	تاریخ و تمدن اسلامی	تاریخ فرهنگ و تمدن اسلامی	۲	۳۲
		تاریخ تحلیلی صدر اسلام	۲	۳۲
		تاریخ امامت	۲	۳۲
۵	آشنایی با منابع اسلامی	تفسیر موضوعی قرآن	۲	۳۲
		تفسیر موضوعی نهج‌البلاغه	۲	۳۲
۶	-	زبان فارسی	۳	۴۸
۷	-	زبان انگلیسی	۳	۴۸
۸	-	تربیت بدنی ۱	۱	۳۲
۹	-	تربیت بدنی ۲	۱	۳۲
۱۰	-	دانش خانواده و جمعیت	۲	۳۲
		جمع کل واحدهای عمومی	۲۲	

- دو درس به ارزش ۴ واحد از مجموعهٔ درسهای مبانی نظری اسلام
- یک درس به ارزش ۲ واحد از مجموعهٔ درسهای اخلاق اسلامی
- یک درس به ارزش ۲ واحد از مجموعهٔ درسهای انقلاب اسلامی
- یک درس به ارزش ۲ واحد از مجموعه درسهای تاریخ تمدن اسلامی
- یک درس به ارزش ۲ واحد از مجموعه درسهای آشنایی با منابع اسلامی

۲-۴ درسهای پایه (۲۰ واحد)

درسهای پایه					
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات ضریبی	پیشنیاز
۱	ریاضی عمومی ۱	۳	نظری	۴۸	-
۲	ریاضی عمومی ۲	۳	نظری	۴۸	ریاضی عمومی ۱
۳	فیزیک ۱	۳	نظری	۴۸	-
۴	فیزیک ۲	۳	نظری	۴۸	ریاضی عمومی ۱
۵	امار و احتمال مهندسی	۲	نظری	۴۸	ریاضی عمومی ۲
۶	معادلات دیفرانسیل	۳	نظری	۴۸	ریاضی عمومی ۱
۷	کارگاه کامپیوتر	۱	عملی	۴۸	مبانی کامپیوتر و برنامه سازی
۸	آزمایشگاه فیزیک ۲	۱	عملی	۳۲	فیزیک ۲
	جمع	۲۰			

۲-۳ درسهای اصلی (۵۹ واحد)

درسهای اصلی						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	مبانی کامپیوتر و برنامه‌سازی	۳	نظری	۴۸	-	
۲	مدارهای الکتریکی	۳	نظری	۴۸	معادلات دیفرانسیل	
۳	ریاضیات گسسته	۳	نظری	۴۸		ریاضی عمومی ۱ و مبانی کامپیوتر و برنامه‌سازی
۴	برنامه‌سازی پیشرفته	۳	نظری	۴۸	مبانی کامپیوتر و برنامه‌سازی	
۵	ساختمان‌های داده	۳	نظری	۴۸	ریاضیات گسسته و برنامه‌سازی پیشرفته	
۶	مدارهای منطقی	۳	نظری	۴۸		ریاضیات گسسته
۷	نظریه زبان‌ها و ماشین‌ها	۳	نظری	۴۸	ساختمان‌های داده	
۸	زبان تخصصی	۲	نظری	۳۲	زبان خارجی	
۹	روش پژوهش و ارائه	۲	نظری	۳۲	زبان تخصصی	
۱۰	ریاضیات مهندسی	۳	نظری	۴۸	ریاضی عمومی ۲ و معادلات دیفرانسیل	
۱۱	معماری کامپیوتر	۳	نظری	۴۸	مدارهای منطقی	
۱۲	سیستم‌های عامل	۳	نظری	۴۸	ساختمان‌های داده و معماری کامپیوتر	
۱۳	طراحی الگوریتم‌ها	۳	نظری	۴۸	ساختمان‌های داده	
۱۴	طراحی کامپیوتری سیستم‌های دیجیتال	۳	نظری	۴۸	معماری کامپیوتر	
۱۵	سیگنال‌ها و سیستم‌ها	۲	نظری	۴۸	ریاضیات مهندسی	
۱۶	ریزپردازنده و زبان اسمبلی	۳	نظری	۴۸	معماری کامپیوتر	
۱۷	شبکه‌های کامپیوتری	۳	نظری	۴۸	سیستم‌های عامل	
۱۸	هوش مصنوعی و سیستم‌های خبره	۳	نظری	۴۸	ساختمان‌های داده	
۱۹	اصول طراحی کامپایلر	۳	نظری	۴۸	ساختمان‌های داده	
۲۰	آزمایشگاه سیستم‌های عامل	۱	عملی	۳۲		سیستم‌های عامل
۲۱	آزمایشگاه مدارهای منطقی و معماری کامپیوتر	۱	عملی	۳۲	مدارهای منطقی	معماری کامپیوتر

درسهای اصلی						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۲۲	آزمایشگاه ریزپردازنده	۱	عملی	۳۲	ریزپردازنده و زبان اسمبلی	
۲۳	آزمایشگاه شبکه‌های کامپیوتری	۱	عملی	۳۲		شبکه‌های کامپیوتری
جمع		۵۹				

۱۲

۴-۲ درسهای تخصصی گرایش‌های چهارگانه رشته مهندسی کامپیوتر

- درسهای تخصصی گرایش معماری سیستم‌های کامپیوتری (۱۹ واحد)

درسهای تخصصی گرایش معماری سیستم‌های کامپیوتری						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	مدارهای الکترونیکی	۳	نظری	۴۸	مدارهای الکتریکی	
۲	الکترونیک دیجیتال	۳	نظری	۴۸	مدارهای الکترونیکی	
۳	انتقال داده‌ها	۲	نظری	۳۲	آمار و احتمال مهندسی و سیگنال‌ها و سیستم‌ها	
۴	سیستم‌های کنترل خطی	۳	نظری	۴۸	سیگنال‌ها و سیستم‌ها	
۵	آزمایشگاه مدارهای الکترونیکی	۱	عملی	۳۲		همنیاز مدارهای الکترونیکی
۶	آزمایشگاه الکترونیک دیجیتال	۱	عملی	۳۲	الکترونیک دیجیتال	
۷	آزمایشگاه ابزارهای طراحی با کمک کامپیوتر	۱	عملی	۳۲	طراحی کامپیوتری سیستم‌های دیجیتال	الکترونیک دیجیتال
۸	کارآموزی	۱	عملی		بعد از ۸۰ واحد	
۹	پروژه معماری کامپیوتر	۳	عملی		بعد از ۱۰۰ واحد	
جمع		۱۹				

- درسهای تخصصی گرایش نرم‌افزار (۱۹ واحد)

درسهای تخصصی گرایش نرم‌افزار						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	هم‌نیاز
۱	تحلیل و طراحی سیستم‌ها	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۲	پایگاه داده‌ها	۳	نظری	۴۸	ساختمان‌های داده	
۳	طراحی زبان‌های برنامه‌سازی	۳	نظری	۴۸	اصول طراحی کامپایلر	
۴	مهندسی نرم‌افزار	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
۵	مهندسی اینترنت	۳	نظری	۴۸	شبکه‌های کامپیوتری	پایگاه داده‌ها
۶	کارآموزی	۱	عملی		بعد از ۸۰ واحد	
۷	پروژه نرم‌افزار	۲	عملی		بعد از ۱۰۰ واحد	
جمع		۱۹				

- درسهای تخصصی گرایش رایانش امن (۳۱ واحد)

درسهای تخصصی گرایش رایانش امن					
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	پایگاه دادهها	۳	نظری	۴۸	ساختمان‌های داده	
۲	تحلیل و طراحی سیستم‌ها	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۳	امنیت شبکه	۳	نظری	۴۸	شبکه‌های کامپیوتری	
۴	مبانی رایانش امن	۳	نظری	۴۸	سیستم‌های عامل	شبکه‌های کامپیوتری
۵	امنیت سیستم‌های پایه	۳	نظری	۴۸	پایگاه دادهها سیستم‌های عامل	
۶	مدیریت امنیت اطلاعات	۳	نظری	۴۸	مبانی رایانش امن	
۷	مبانی رمزنگاری	۳	نظری	۴۸	مبانی رایانش امن	
۸	توسعه امن نرم‌افزار	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
۹	حقوق و ادله الکترونیکی در امنیت	۳	نظری	۴۸	امنیت شبکه، امنیت سیستم‌های پایه	
۱۰	کارآموزی (در زمینه رایانش امن)	۱	عملی		بعد از ۸۰ واحد	
۱۱	پروژه رایانش امن	۳	عملی		بعد از ۱۰۰ واحد	
	جمع	۳۱				

- درسهای تخصصی گرایش فناوری اطلاعات (۳۱ واحد)

درسهای تخصصی گرایش فناوری اطلاعات						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	تحلیل و طراحی سیستمها	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۲	پایگاه داده‌ها	۳	نظری	۴۸	تحلیل و طراحی سیستمها	
۳	اصول فناوری اطلاعات	۳	نظری	۴۸		
۴	اصول مدیریت و برنامه‌ریزی راهبردی فناوری اطلاعات	۳	نظری	۴۸		
۵	مدیریت پروژه‌های فناوری اطلاعات	۳	نظری	۴۸		
۶	یکپارچه‌سازی کاربردهای سازمانی	۳	نظری	۴۸	تحلیل و طراحی سیستمها، شبکه‌های کامپیوتری	
۷	مبانی رایانش امن	۳	نظری	۴۸	شبکه‌های کامپیوتری	
۸	اقتصاد مهندسی	۲	نظری	۴۸		
۹	تجارت الکترونیکی	۳	نظری	۴۸	اقتصاد مهندسی، شبکه‌های کامپیوتری	
۱۰	کارآموزی	۱	عملی		بعد از ۸۰ واحد	
۱۱	پروژه فناوری اطلاعات	۳	عملی		بعد از ۱۰۰ واحد	
جمع		۳۱				

۲-۵ درسهای تمرکزهای تخصصی اختیاری (۱۲ واحد تمرکز برای گرایش‌های با ۱۸ واحد تخصصی)

درسهای تمرکز تخصصی سیستم‌های مجتمع						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	همطراحی سخت‌افزار- نرم‌افزار	۳	نظری	۴۸	معماری کامپیوتر	
۲	سیستم‌های نهفته و بیدرنگ	۳	نظری	۴۸	سیستم‌های عامل و ریزپردازنده و زبان اسمبلی	
۳	طراحی سیستم‌های مجتمع پرتراکم	۳	نظری	۴۸	الکترونیک دیجیتال	
۴	معماری شتاب دهنده‌های شی‌گرا	۳	نظری	۴۸	معماری کامپیوتر و برنامه‌سازی پیشرفته	
۵	طراحی مدارهای واسط	۳	نظری	۴۸	ریزپردازنده و زبان اسمبلی	
۶	طراحی مدارهای دیجیتال فرکانس بالا	۳	نظری	۴۸	مدارهای الکتریکی	
	جمع	۱۲				
اخذ چهار درس از شش درس الزامی است.						

درسهای تمرکز تخصصی شبکه‌های کامپیوتری						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	امنیت شبکه	۳	نظری	۴۸	شبکه‌های کامپیوتری	
۲	سیستم‌های نهفته و بیدرنگ	۳	نظری	۴۸	سیستم‌های عامل و ریزپردازنده و زبان اسمبلی	
۳	*۱۵- یکی از دو درس زیر: مهندسی اینترنت یا انتقال داده	۳	نظری	۴۸	پیشنیاز تعیین شده هر یک	
۴	مبانی شبکه‌های بی‌سیم	۳	نظری	۴۸	انتقال داده‌ها	
	جمع	۱۲				

*۱۵- مهندسی اینترنت برای دانشجویان گرایش معماری سیستم‌های کامپیوتری و انتقال داده برای دانشجویان گرایش نرم‌افزار.

۱۷

درسهای تمرکز تخصصی هوش مصنوعی					
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	مبانی هوش محاسباتی	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۲	مبانی بینائی کامپیوتر	۳	نظری	۴۸	مبانی هوش محاسباتی	
۳	مبانی پردازش زبان و گفتار	۳	نظری	۴۸	آمار و احتمال مهندسی، سیگنال‌ها و سیستم‌ها	
۴	اصول رباتیکز	۳	نظری	۴۸	سیگنال‌ها و سیستم‌ها	
	جمع	۱۲				

درسهای تمرکز تخصصی سیستم‌های نرم‌افزاری					
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	تعامل انسان و کامپیوتر	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
۲	آزمون نرم‌افزار	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
۳	روش‌های رسمی در مهندسی نرم‌افزار	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
۴	طراحی شی‌گرای سیستم‌ها	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
	جمع	۱۲				

۱۸

درسهای تمرکز تخصصی الگوریتم و محاسبات						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	نظریه و الگوریتم‌های گراف	۳	نظری	۴۸	ریاضیات گسسته	
۲	نظریه محاسبات	۳	نظری	۴۸	نظریه زبان‌ها و ماشین‌ها	
۳	مبانی نظریه بازی‌ها	۳	نظری	۴۸	طراحی الگوریتم‌ها	
۴	الگوریتم‌های پیشرفته	۳	نظری	۴۸	طراحی الگوریتم‌ها	
۵	مقدمه‌ای بر مسابقات برنامه‌نویسی	۲	نظری	۴۸	طراحی الگوریتم‌ها	
۶	منطق در علوم و مهندسی کامپیوتر	۳	نظری	۴۸	ساختمان‌های گسسته و مبانی کامپیوتر و برنامه‌سازی	
جمع		۱۲				
اخذ چهار درس از شش درس الزامی است.						

درسهای تمرکز تخصصی بازی‌های کامپیوتری						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	سیستم‌های چند رسانه‌ای	۳	نظری	۴۸	آمار و احتمال مهندسی، سیگنال‌ها و سیستم‌ها	
۲	طراحی بازی‌های کامپیوتری	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۳	گرافیک کامپیوتری	۳	نظری	۴۸	برنامه‌سازی پیشرفته	
۴	مبانی پویا نمایی کامپیوتری	۳	نظری	۴۸	گرافیک کامپیوتری	
جمع		۱۲				

درسهای تمرکز تخصصی سیستم‌های اطلاعاتی						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	پیاده‌سازی سیستم پایگاه داده	۳	نظری	۴۸	اصول طراحی پایگاه داده	
۲	مبانی داده کاوی	۳	نظری	۴۸	اصول طراحی پایگاه داده، ساختمان‌های داده	
۳	مبانی بازیابی اطلاعات و جستجوی وب	۳	نظری	۴۸	طراحی الگوریتم‌ها	
۴	سیستم‌های اطلاعات مدیریت	۳	نظری	۴۸	تحلیل و طراحی سیستم‌ها	
جمع		۱۲				

درسهای تمرکز تخصصی امنیت رایانه						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات تدریس	پیشنیاز	همنیاز
۱	امنیت شبکه	۳	نظری	۴۸	شبکه‌های کامپیوتری	
۲	مبانی رایانش امن	۳	نظری	۴۸		
۳	امنیت سیستم‌های پایه	۳	نظری	۴۸	پایگاه داده‌ها و سیستم‌های عامل	
۴	مدیریت امنیت اطلاعات	۳	نظری	۴۸		مبانی رایانش امن
جمع		۱۲				

۶-۲ درسهای اختیاری: اخذ دو واحد آزمایشگاه یا کارگاه در میان درسهای اختیاری با تصویب گروه تخصصی الزامی است.

درسهای اختیاری همهٔ گرایش‌ها						
ردیف	عنوان	تعداد واحد	نوع واحد	ساعات درس	پیشنیاز	همنیاز
۱	یک درس از کارشناسی ارشد رشته مهندسی کامپیوتر	۳	نظری	۴۸		
۲	مباحث ویژه ۱	۳	نظری	۴۸	درسهای جدید و روزآمد با مجوز دانشکده در این قالب می‌تواند عرضه شود	
۳	مباحث ویژه ۲	۲	نظری	۴۸	درسهای جدید و روزآمد با مجوز دانشکده در این قالب می‌تواند عرضه شود	
۴	تا هشت واحد از درس‌های گرایش‌ها یا تمرکزهای دیگر مهندسی کامپیوتر	تا ۸ واحد			با رعایت پیشنیاز در هر مورد	
۵	یک درس از دوره کارشناسی دانشکده‌های دیگر	۳	نظری	۴۸		
۷	نمونه‌سازی سیستم‌های پیچیده سخت‌افزاری-نرم‌افزاری	۳	نظری	۴۸	معماری کامپیوتر و سیستم‌های عامل	
۱۱	مقدمه‌ای بر علم اعصاب	۳	نظری	۴۸		
۲۲	آزمایشگاه مهندسی نرم‌افزار	۱	عملی	۳۳	همنیاز درس تحلیل و طراحی سیستم‌ها	
۲۳	آزمایشگاه اصول طراحی کامپایلر	۱	عملی	۳۳	همنیاز درس	
۲۴	آزمایشگاه پایگاه داده	۱	عملی	۳۳	همنیاز درس	
۲۵	آزمایشگاه مدارهای الکتریکی	۱	عملی	۳۳	همنیاز درس	
۲۶	آزمایشگاه مدارهای واسط	۱	عملی	۳۳	طراحی مدارهای واسط	
۲۷	آزمایشگاه اصول رباتیکز	۱	عملی	۳۳	همنیاز درس	
۲۸	آزمایشگاه گرافیک کامپیوتری	۱	عملی	۳۳	همنیاز درس	
۲۹	آزمایشگاه بازی‌های کامپیوتری	۱	عملی	۳۳	همنیاز درس طراحی بازی‌های کامپیوتری	
۳۰	آزمایشگاه واقعیت مجازی	۱	عملی	۳۳	همنیاز درس	
۳۱	آزمایشگاه امنیت شبکه	۱	عملی	۳۳	همنیاز درس	
۳۳	کارگاه ساخت ربات	۱	عملی	۴۸	درس پایه جایگزین "کارگاه عمومی" به صورت اختیاری عرضه می‌شود	

۲۱

					سیستم‌ها	
۲۴	آزمایشگاه اتوماسیون صنعتی	۱	عملی	۲۴	سیستم‌های کنترل خطی	
۲۵	آزمایشگاه سیستم‌های کنترل خطی	۱	عملی	۲۴	سیستم‌های کنترل خطی	
۲۶	سیستم‌های اتوماسیون صنعتی	۳	نظری	۴۸	ریز پردازنده و زبان اسمبلی	
۲۷	علوم ومعارف دفاع مقدس	۲	نظری	۳۴	-	
	جمع واحدهای اختیاری	۸ واحد				

***لازم است مجموع درسهای اختیاری اخذ شده از این جدول ۸ واحد باشد.**