به نام پروردگار دانایی

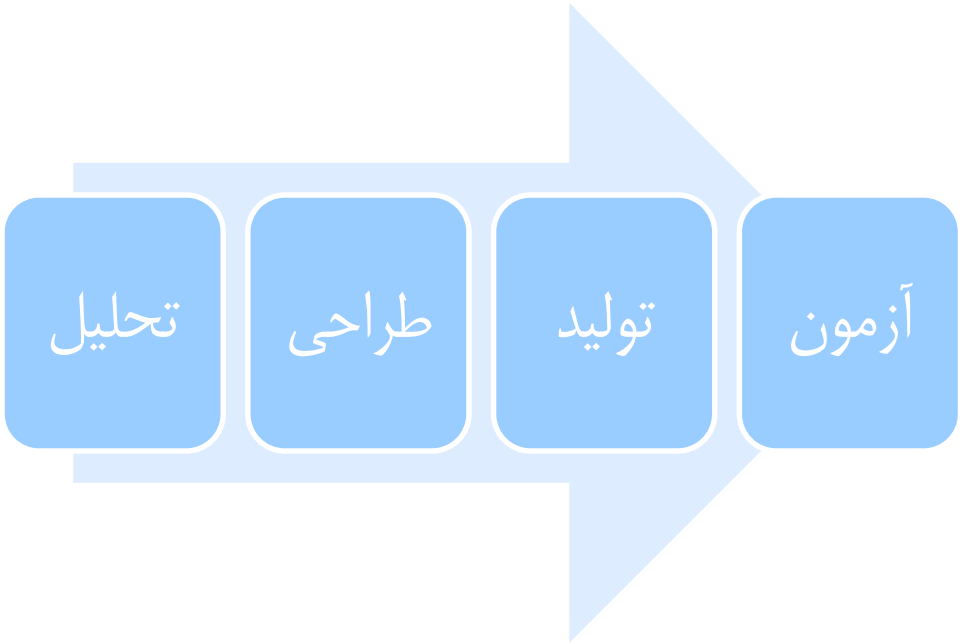
طراحی سیستم‌های شی‌گرا

مقدمه‌ای بر اصول شی‌گرایی – بخش دوم

سید کاوه احمدی

- اصول اساسی شی‌گرایی

توسعه برنامه کاربردی



- با نگرش شی‌گرا:
  - تحلیل شی‌گرا (OOA)
  - طراحی شی گرا (OOD)
  - برنامه‌نویسی شی‌گرا (OOP)

اصول اساسی شی گرایی

- تجرید (Abstraction)
- واحدبندی (Modularity)
- محصور سازی (Encapsulation)
- سلسله مراتب (Hierarchy)

مفاهیم کلیدی در برنامه‌نویسی شی‌گرا

- اشیا و کلاس‌ها (Objects & Classes)
- محصور سازی (Encapsulation)
- وراثت (Inheritance)
- چند ریختی (Polymorphism)

تجرید

- فرآیند متمرکز شدن روی ویژگی‌های اصلی یک پدیده از یک زاویه دید مشخص.
  - تمرکز روی بخش‌هایی که برای ناظر مهم است نه اینکه واقعا مهم باشد.
- نادیده گرفتن ویژگی‌های موقت و غیر مهم آن پدیده.
  - نادیده گرفتن جزئیات برای متمرکز شدن توجه بر سطح بالاتری از یک مسئله.
- نادیده گرفتن جزئیات در زمان مناسب
  - فکر کردن به آنچه یک متد/شی/کلاس انجام می‌دهد، نه چگونگی انجام آن.
  - برخورد کردن با اجزای سیستم به عنوان یک `black box`

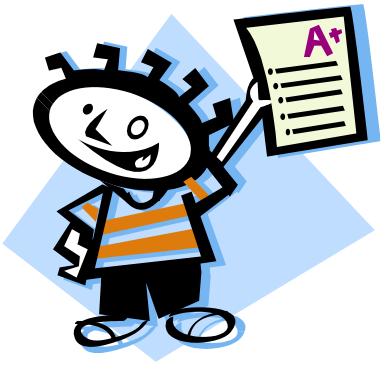
تجرید

- **یک مثال:** بدن انسان
  - سیستم گردش خون یک تجرید از بدن انسان است.
    - نگاه به بدن انسان از یک زاویه دید مشخص
    - اسکلت بندی بدن انسان مهم است و در تجرید گردش خون آنرا در نظر می‌گیریم اما با جزئیات آن کاری نداریم (در اینجا برای ما یک `black box` است).

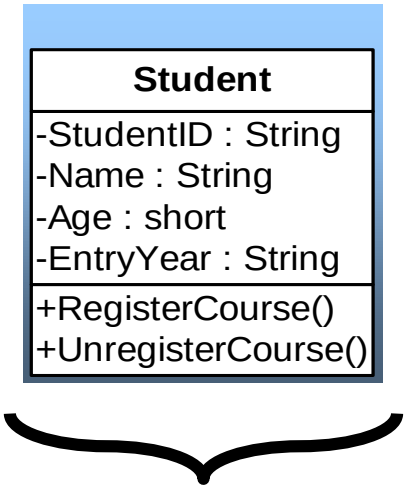
انواع تجرید

- تجرید موجودیت (Entity Abstraction)

Real Object: Student



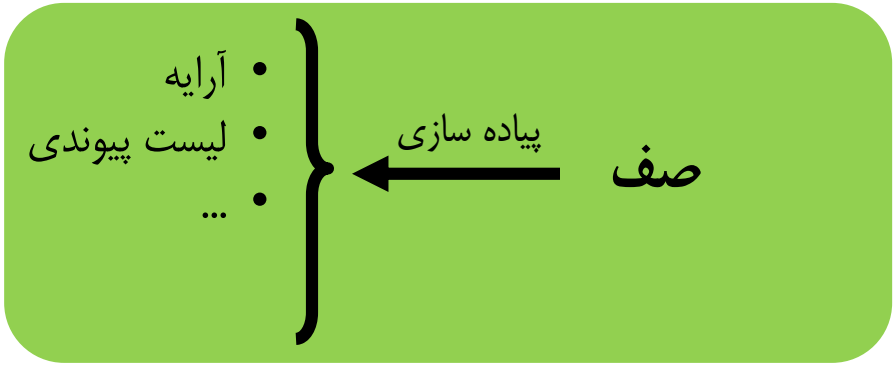
Abstraction: Student



بخشی از ویژگی‌ها و رفتارها از بیرون دیده می‌شود. نحوه پیاده‌سازی رفتارها را نمی‌دانیم.

انواع تجرید

- تجرید رفتار (Behavioral Abstraction)
  - به نحوه‌ی پیاده‌سازی صف کاری ندارم. برایمان اهمیتی هم ندارد. مهم این است که صف باشد!



ویژگی‌های تجرید

- تجرید با نمود خارجی یک شی سروکار دارد. به بیان دیگر تجرید آن بخش‌هایی را نگاه می‌کند که یک ناظر بیرونی آنرا می‌بیند.
  - می‌داند سیستم چه کاری انجام می‌دهد اما نمی‌داند چگونه
  - می‌داند سیستم از چه بخش‌هایی تشکیل شده.
  - رفتار از بیرون سیستم – مثال استفاده از اتومبیل
- تجرید داری سطوح مختلفی است. (میزان پرداختن به جزئیات)
  - سطح تجرید بالاتر: جزئیات کمتر – وسعت دید بیشتر
  - مثال هواپیما و ارتفاع

تجرید و پیچیدگی

- یکی از ابزارهای اصلی کنترل و تسلط بر پیچیدگی
  - جزئیات بی شماری درباره یک پدیده مطرح است
  - تنها ابعاد اساسی پدیده مد نظر قرار خواهد گرفت
  - به اجزای مختلف به صورت `black box` نگاه می‌شوند که با استفاده از واسطهای تعریف شده با هم تعامل دارند

محصور سازی

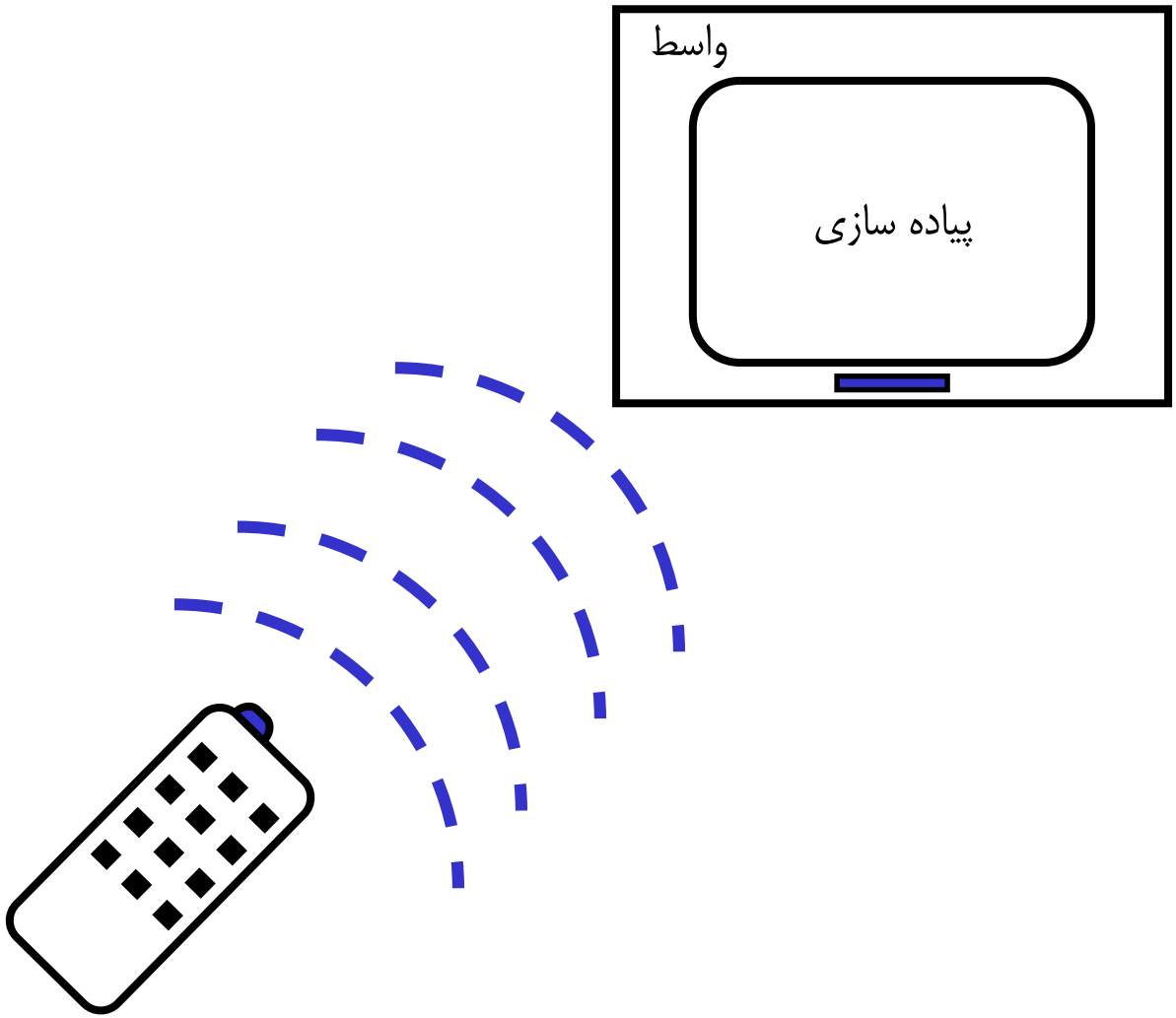
- محصور سازی عبارتست از محدود کردن روش‌های دسترسی یا استفاده از یک شی به طوری که از تاثیرات ناخواسته (Side Effects) یا کنترل نشده جلوگیری شود.
- اصل محصور سازی بر پایه اصل مخفی کردن اطلاعات است.
  - این اصل می‌گوید هر شی باید آن میزان از اطلاعات را در اختیار داشته باشد که به آن نیاز دارد.

محصور سازی

- از نظر محصور سازی هر شی به دو بخش اساسی تقسیم می‌شود:
  - واسطها (Interface)
  - ساختار داخلی
- سایر اشیا نمی‌دانند واسطهای شی چگونه پیاده‌سازی شده و کارهای مورد نظر را چگونه انجام می‌دهد.
- هر شی برای تعامل با سایر اشیا از واسطهای آن استفاده می‌کند.

محصور سازی

- ارتباط بین اشیا تنها از راه واسطها



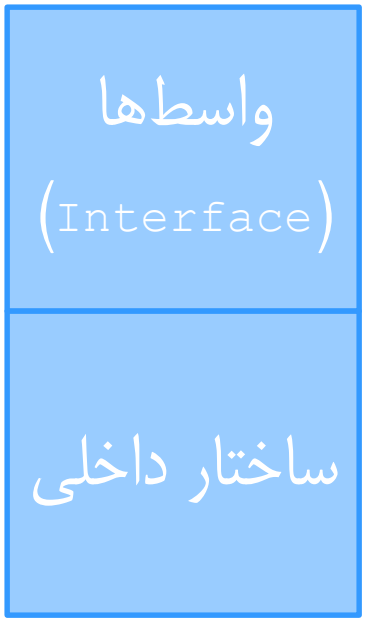
محصور سازی و پیچیدگی

- هر ماژول برای تعریف کننده آن باز و برای استفاده کننده آن بسته است.
  - جلوگیری از خرابکاری‌های احتمالی و محلی کردن گستره خطاها در شی
- با ثبات داشتن واسط یک شئ، می‌توان هر تغییری در پیاده سازی آن شئ انجام گیرد
  - هر ماژول برای توسعه باز و برای تغییر بسته است.
    - با تغییر واسطها، اشیای استفاده کننده از کلاس نیز برای سازگاری از واسطهای جدید نیاز به تغییر پیدا می‌کنند.
      - تغییر در بخش منجر به تغییر در سایر بخش‌ها خواهد شد!

تجرید و محصور سازی

- تجرید و محصور سازی دو مفهوم مکمل هستند
  - تجرید: رفتارهای قابل مشاهده یک شی (نمود خارجی)
  - محصور سازی: پیاده سازی رفتارها (نمود داخلی)
- تجرید مکانیزم تعیین جزئیاتی است که باید پنهان شود.
- محصورسازی، فرایند پنهان سازی جزئیات و کنترل دسترسی به آن است.
- محصور سازی یک مفهوم نسبی.

تجرید و محصور سازی



- برای تعامل با سایر اشیا
- اشیا دیگر دسترسی دارند
- **تجرید با آن درگیر**

- پیاده‌سازی رابط‌ها
- اشیا دیگر دسترسی ندارند
- **محصور سازی با آن درگیر**

یک واسط، چند پیاده سازی:
ویژگی‌ای که به یک رابط امکان می‌دهد تا **برای گروهی از عملیات** مورد استفاده قرار گیرند.

چندریختی (Polymorphism)

مفهوم یک واسط، چند پیاده سازی امکان استفاده مجدد را بالامی برد

واحدبندی

- پروسه تقسیم‌بندی یک مسئله به قسمت‌های خوش تعریف است به نحوی که بتوانند به طور جداگانه تولید و آزمایش شوند و به روش‌های خوش تعریفی بتوانند تعامل برقرار کنند.
- سیستمی را واحدبندی شده می‌گویند که به مجموعه‌ای از واحدهای منسجم و معنی‌دار که وابستگی بین آنها حداقل است تجزیه شده باشد.
- ماژول‌ها واحد تشکیل دهنده ساختار فیزیکی سیستم نرم افزاری هستند.

واحدبندی

- واحدها باید ویژگی‌های Building Blocks را داشته باشند:
  - استقلال (Independence)
  - واسطهای خوش تعریف (Well-defined Interfaces)
- یک مجموعه از تجریدهای مرتبط
- واحدبندی قابلیت استفاده مجدد را بالا می‌برد.

واحدبندی

- انسجام (Cohesion): درجه ارتباط عملکردهای عناصر داخلی یک ماژول دارای است.
- وابستگی (Coupling): درجه ارتباط واحدهای گوناگون به یکدیگر.
- درون سیستمی قوی – برون سیستمی ضعیف

واحدبندی

می‌خواهید برای تعطیلات به اسپانیا بروید

- می‌توانید مسئله را به چند واحد تقسیم کنید:
  - رفتن به فرودگاه
  - پرواز به اسپانیا
  - رفتن از فرودگاه در اسپانیا به هتل
- هر واحد می‌تواند به صورت جداگانه حل شود. این کار مسائل را ساده‌تر می‌کند.
  - پیدا کردن یک تاکسی تلفنی برای رفتن به فرودگاه، یک آژانس هواپیمایی برای رفتن به اسپانیا و یک اتوبوس شاتل برای رفتن به هتل

واحدبندی و پیچیدگی

- شکستن مساله به اجزائی کوچکتر یکی از راههای کارا برای مقابله با پیچیدگی است.
- مثال: اگر مسئله P را به زیر مسئله‌های P_1، P_2 و P_3 تقسیم کنیم آنگاه
  - $C(P) > C(P_1) + C(P_2) + C(P_3)$
  - $E(P) > E(P_1) + E(P_2) + E(P_3)$
  - C: Complexity
  - E: Solving Energy

واحدبندی و استفاده مجدد

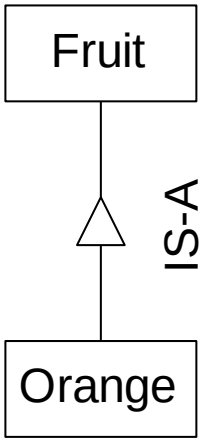
- اگر شرایط بیان شده در تعریف واحد بندی رعایت گردد، ماژول‌های با قابلیت استفاده مجدد بالایی می‌توان ایجاد کرد.
  - استفاده مجدد به معنی `Copy` و `Paste` نیست!
- تعیین معیار شکستن یک مساله مهمترین عامل برای موفقیت استفاده از این ویژگی است.

سلسله مراتب

- سلسله مراتب عبارت از مرتب ساختن تجریدها در سطوح مختلف.
- برای یک سیستم می‌توان سلسله مراتب مختلفی را در نظر گرفت. سلسله مراتب درک ما را از سیستم بالا می‌برد.
- انواع سلسله مراتب :
  - سلسله مراتب ساختار کلاس (IS-A)
  - سلسله مراتب ساختار شی (PART-OF)

سلسله مراتب IS-A

- سلسله مراتب ساختار کلاس.
- رابطه‌ی زیر نوع بین انواع مختلف اشیا را نشان می‌دهد.
- نام دیگر این نوع سلسله مراتب تخصیص / تعمیم (Generalization/Specialization) می‌باشد.



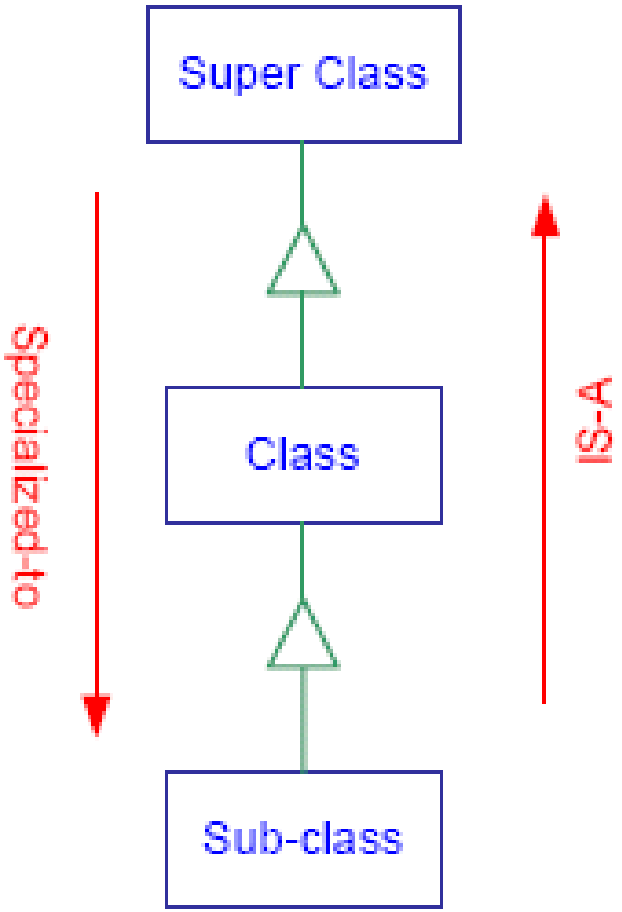
Orange IS-A Fruit

وراثت (Inheritance)

- وراثت مهمترین شکل سلسله مراتب IS-A است.

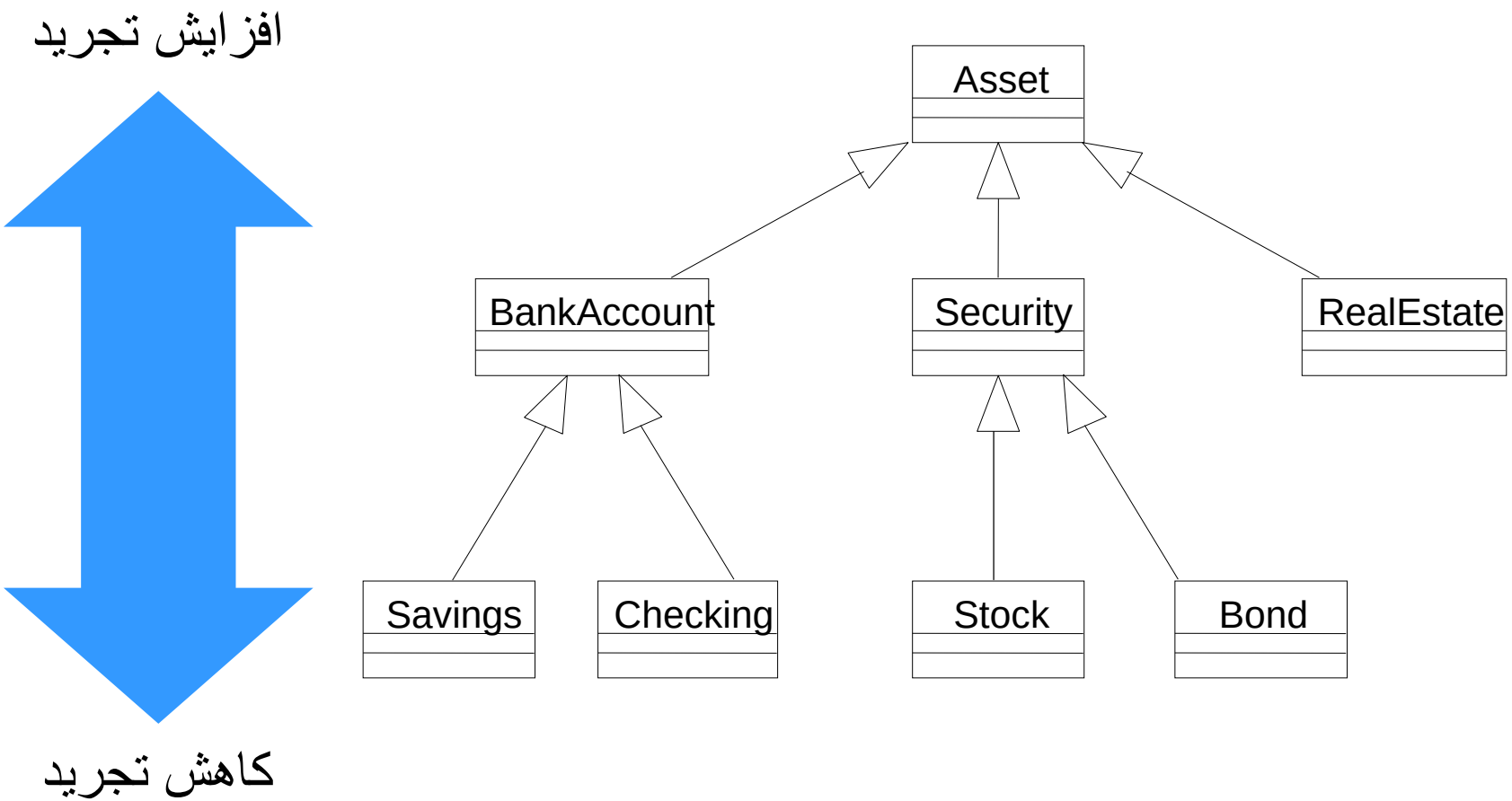
- وراثت عبارت است از رابطه بین چند کلاس که در آن یک کلاس در ساختار ، رفتار یا هر دو با یک کلاس (Single Inheritance) یا چند کلاس (Multiple Inheritance) دیگر شرکت دارد.
- کلاس فرزند یک تخصیص از کلاس عمومی‌تر (کلاس پدر) را نمایش می دهد.
- Type → Subtypes (Person / Student)
- کلاس‌های فرزند با اضافه کردن متدها و خصوصیات جدید، کلاس پدر را گسترش می‌دهند
- استفاده مجدد از کدها

وراثت



وراثت

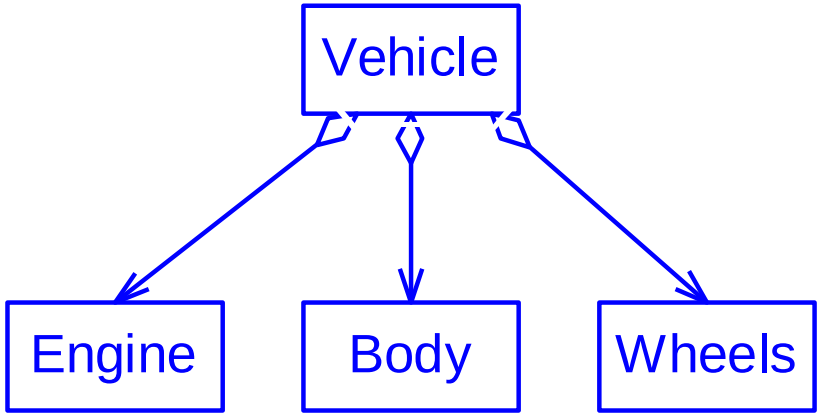
- سطوح تجرید متفاوت در سطوح مختلف سلسه مراتب نمایان می‌گردد.



سلسله مراتب PART-OF

- این سلسله مراتب اجزا تشکیل دهنده اشیا را نشان می‌دهد.
  - یک کلاس از یک یا چند کلاس دیگر تشکیل می‌شود.
- نام دیگر این نوع سلسله مراتب رابطه کل / جز (Whole/Part) یا رابطه تجمعی (Aggregation Relationship) است.

سلسله مراتب PART-OF



The Vehicle ***HAS-An*** Engine

The Engine is ***PART-OF*** Vehicle

سلسله مراتب و پیچیدگی

- با ساماندهی تجریدها در سلسله مراتب PART-OF و IS-A درک ما نسبت به سیستم افزایش می‌یابد.
- سلسله مراتب PART-OF روابط موجود بین اشیا و فعل و انفعالاتی که رخ می‌دهد را نمایان می‌سازد.
- سلسله مراتب IS-A افزونگی موجود در سیستم را مدیریت می‌کند (Economy of Expression)

محصور سازی و سلسله مراتب

استفاده از **وراثت** با **محصورسازی تام** تعارض دارد زیرا مستلزم دسترسی مستقیم کلاس فرزند به بعضی از اعمال و داده های اختصاصی کلاس پدر است.

– زبان‌های برنامه‌نویسی و شی‌گرایی

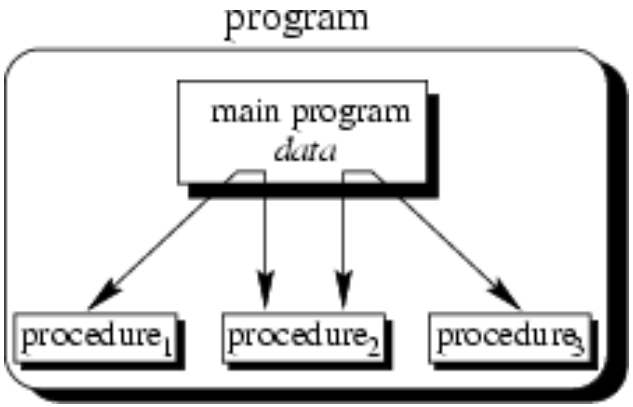
یک زبان برنامه‌سازی شی‌گرا

- بطور خلاصه می‌توان گفت که یک نرم‌افزار از مجموعه‌ای از داده‌ها و الگوریتم‌ها تشکیل شده است.
- یک زبان شی‌گرا باید
  - باید از اشیا به عنوان تجرید الگوریتم + تجرید داده حمایت کند.
  - هر شی دارای نوع باشد (از مفهوم کلاس پشتیبانی کند).
  - بتوان یک رابطه زیرنوعی را بین اشیا ایجاد کرد. (سلسله مراتب)

پارادایم‌های برنامه‌نویسی

- عمومی تر
  - **دستوری** (ساخت یافته – رویه‌ای)**:** این را انجام بده بعد آنرا
    - `Fortran, Algol, Pascal, Basic, C,…`
  - **شی‌گرا:** ارسال پیغام بین اشیا
    - `Simula, Smalltalk, C++, Java, C#,…`
- کمتر مورد استفاده
  - **تابعی:** ارزیابی یک الگو و ارسال آن
    - `Haskell, ML, Lisp, Scheme`
  - **منطقی:** بر پایه اظهارات منطقی و پاسخ به آنها
    - `Prolog`

پارادایم برنامه‌نویسی دستوری

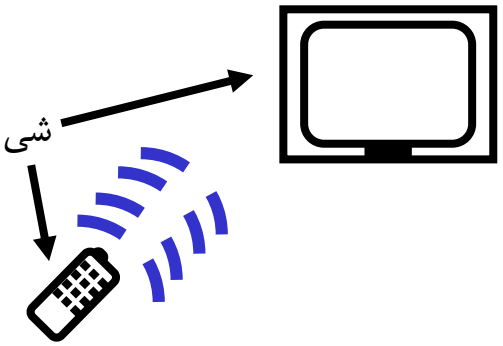


- قدیمی‌ترین و سرراست‌ترین روش برنامه‌نویسی
  - تمرکز بر عملکرد سیستم (قرار است چه کاری انجام شود)
- برنامه مجموعه‌ای از دستورالعمل‌ها است
- توابع تا حدودی اجازه واحدبندی، محصورسازی و استفاده مجدد کدها را فراهم می‌آورد.

پارادایم برنامه‌نویسی دستوری

- این خط کد را انجام بده.
- سپس این خط کد را.
- سپس این خط...

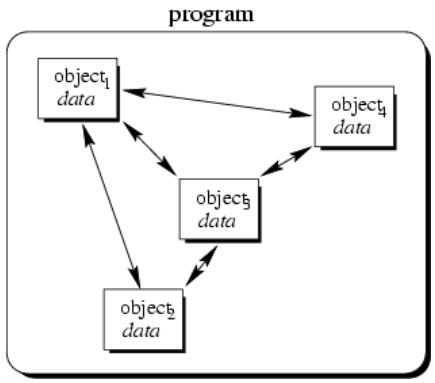
پارادایم برنامه‌نویسی شی‌گرا



- پارادایم رایج‌تر برنامه‌نویسی
- به جای تمرکز برآنچه سیستم باید انجام دهد،

تمرکز روی:

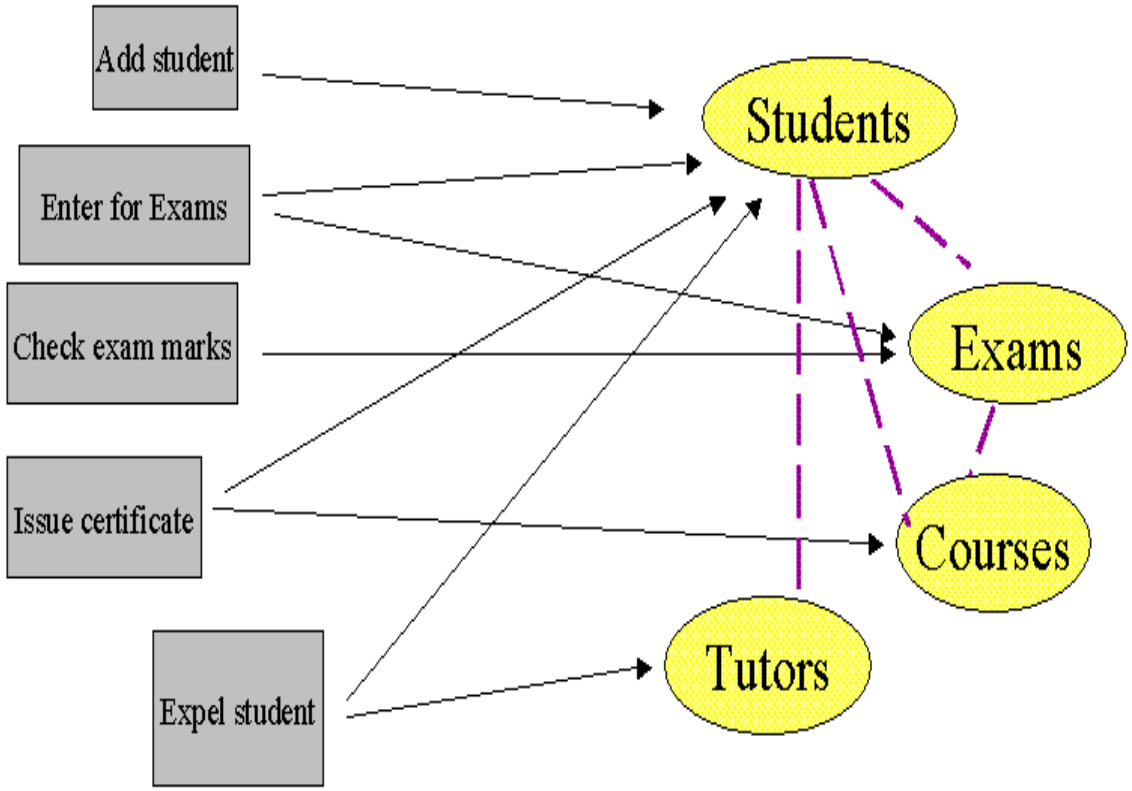
- سیستم شامل چه اشیایی است
- چگونه با هم تعامل دارند (پیغام)



پارادایم برنامه‌نویسی شی‌گرا

- کلید برق پیغامی را به الکتریسیته ارسال می‌کند (شی فعال – تاثیرگذاری روی سایر اشیا با ارسال پیام).
- الکتریسیته پیغامی را به لامپ می‌فرستد (شی عامل – تاثیرگذاری روی سایر اشیا و تاثیر پذیری از آنها).
- لامپ روشن می‌شود (شی غیر فعال – تاثیرپذیر از سایر اشیا)!
- برنامه‌نویسی دستوری به عنوان بخشی از برنامه‌نویسی شی‌گرا مورد نیاز است.

تفاوت رویکرد شی‌گرا و ساخت‌یافته



مقایسه فلسفه پارادایم‌ها

- ساخت‌یافته
  - درک مسئله
  - شکستن مسئله به چند زیر مسئله
  - نوشتن الگوریتم‌هایی برای حل زیر مسئله‌ها
  - نتایج: راه حل یکپارچه، غیر قابل استفاده مجدد
- شی‌گرا
  - درک کامل دامنه مسئله (Problem domain)
  - نوشتن یک سیستم که دامنه را مدل‌سازی می‌کند
  - نتایج: یک مجموعه از اجزا که می‌تواند در شکل‌های مختلف مورد استفاده قرار گیرد. (در این دامنه یا دامنه‌های دیگر)