به نام خدا

درس طراحی و پیاده‌سازی زبان‌های برنامه‌سازی

موضوع: **آشنایی با زبان‌های برنامه‌سازی**

ارائه: صادق سلیمانی
www.Bioinformation.ir

جلسه‌ی اول

عناوین جلسه

- تعریف زبان‌های برنامه‌سازی
- چرا زبان‌های برنامه‌سازی را مطالعه می‌کنیم
- علت تعدد زبان‌های برنامه‌سازی
- تاریخچه
- الگوهای زبان
- مشخصه‌های یک زبان خوب
- انواع زبان از نظر کاربرد

تعریف زبان‌های برنامه‌سازی

- هر تلاشی برای ارائه‌ی (نمایش) الگوریتم‌ها و ساختمان داده‌ها
  - تلاش‌های پیاده‌سازی شده

چرا زبان‌های برنامه‌سازی را مطالعه می‌کنیم؟

1. افزایش درک از زبان مورد استفاده
2. آشنایی با امکانات مفید سایر زبان‌ها و استفاده از آن‌ها
3. تسهیل فراگیری زبان جدید
4. تسهیل و تسریع در انتخاب (زبان برای یک پروژه‌ی خاص)
5. طراحی یک زبان جدید
6. قطعه‌ایی از پازل مهندسی و علوم نرم‌افزار

علت تعدد زبان‌های برنامه‌سازی

1. Evolution (تکامل): یافتن راه‌حل‌های جدید برای انجام کارها

- 1960-1970: Structured Programming Language
  - Fortran, Cobol, Basic
- 1980: Nested Block Structure
  - Algol, Pascal, Ada
- 1967 to now: Object Oriented Programming
  - C++,Smalltalk

- علل تکامل زبان‌ها
  1. توسعه سخت‌افزار و سیستم‌عامل
  2. پیدایش روش‌های جدید (برنامه‌سازی یا پیاده‌سازی)
  3. مطالعات نظری و ریاضی
  4. نیاز به استانداردسازی

علت تعدد زبان‌های برنامه‌سازی

2. خاص منظورگی (Special Purpose)
   - هربرنامه برای مسأله‌های خاصی طراحی شده است
   - Lisp: List processing
   - SNOBOL: Character String
   - C: Low Level Sys. Programming
   - Prolog: Logical Relationship among data

3. ذائقه شخصی (Personal Preference)
   - تفکر Recursive در مقابل تفکر Iterative
   - استفاده یا عدم استفاده از اشاره‌گر

تاریخچه

- نسل اول
  - زبان ماشین (0و 1)
  - از اواخر 1930 تا 1940
  - بیشتر برای محاسبات
  - پیچیدگی و جزییات

01111111 01000101
01001100 01000110
00000001 00000001
00000000 00000000
...

- نسل دوم
  - زبان‌های نمادی (اسمبلی)
  - استفاده از چند نماد به جای رشته‌های دودویی
  - علاوه بر وابستگی به ماشین، به یک مترجم هم وابسته هستند

تاریخچه

- نسل سوم
  - زبان‌های سطح بالا (ساخت‌یافته) و شیءگرا
  - نیاز به مترجم جهت ترجمه به زبان اسمبلی
  - نزدیکتر به زبان طبیعی
  - شروع تحول 1955 تا 1957 با فرترن
  - بعدها: پاسکال، الگول، کوبول، C
  - ساختارهایی مانند Loop، Procedure، Condition

تاریخچه

- نسل چهارم
  - زبان‌های خیلی سطح بالا
  - VC، VB، VC#
  - عرضه به همراه IDE
- نسل پنجم
  - زبان‌های مدل‌سازی
  - UML

الگوهای زبان

- انواع مدل‌های محاسباتی برای زبان‌ها

1. زبان‌های دستوری (Imperative):
   - موسوم به زبان‌های Von Neuman هم
   - به شکل دنباله‌ایی از دستورات
   - کار عمده: تغییر متغییرها
   - مانند: C،Pascal ، Basic، Ada

```
Statement 1;
Statement 2;
...
```

الگوهای زبان

2. زبان‌های تابعی (Functional):
   - مدل محاسباتی مبتنی بر تعریف بازگشتی
   - الهام گرفته از محاسبات Lambda
     - یک برنامه، یک تابع است که از توابع کوچکتری تشکیل شده است
     - مانند Haskell، ML، Lisp

$$function_n(...function_2(function_1(data))...)$$

   - مثال از لیسپ

- Car (a b c (d e) f)
- Car (Cdr (a b c d (e (f g)) h)

الگوهای زبان

3. زبان‌های مبتنی بر قاعده (Rule Based یا Logic Baed):
   - شرایطی بررسی می‌شود و در صورت برقراری کار انجام می‌شود
   - مشابه زبان‌های ترتیبی اما دستورات ترتیبی نیستند
     - مانند Prolog، YACC

$$enabling\ condition_1 \rightarrow action_1$$
$$enabling\ condition_2 \rightarrow action_2$$
.
.
.
$$enabling\ condition_n \rightarrow action_n$$

الگوهای زبان

3. زبان‌های مبتنی بر قاعده (Rule Based یا Logic Based):

Father (x , y)
Grand_Father (x , y) :- Father (x , z) , Father (z , y)

Father (ali , reza)
Father (ali , kazem)
Father (reza , bahram)

Grand_Father(ali,?) ➔ bahram

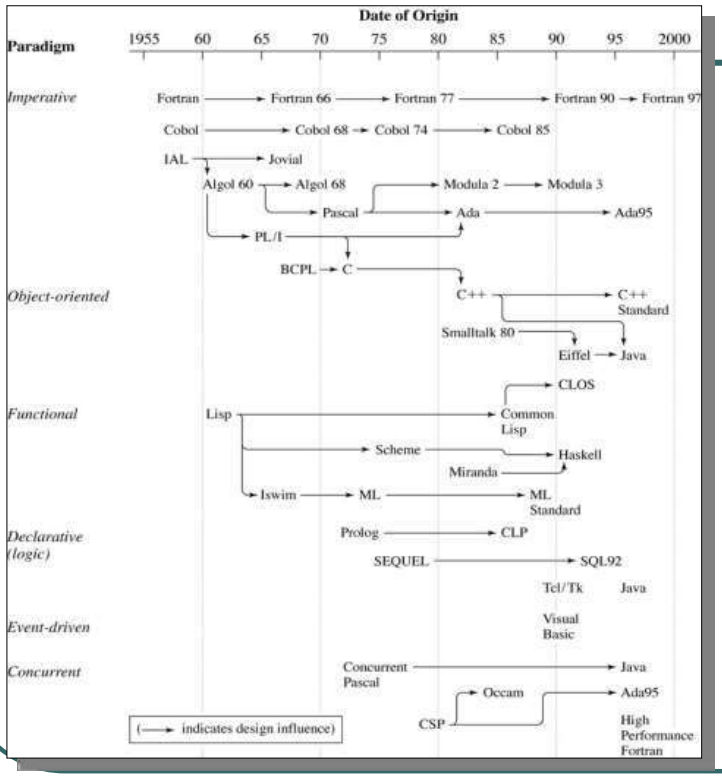
الگوهای زبان

4. زبان‌های شی‌گرا (Object Oriented):
   - تأکید بر سازماندهی سلسله‌مراتبی داده و متد
   - اساس آن کلاس‌ها و شیءهای ساخته شده از روی آن‌هاست
   - کلاس، شی، تجرید، محصور سازی، چند ریختی، وراثت
   - کلاس: رده‌ایی خاص از اشیاء که خصوصیات و رفتار مشابه دارند
     - مانند C#، C++، Java، ریشه‌ی آن‌ها Simula67
5. زبان‌های Data Flow
   - جریانی از اطلاعات بین ندها
   - معمولاً یک مدل موازی فراهم می‌کنند
   - با ورود داده، یک ند شروع به فعالیت می‌کند
     - مانند Val، Id، Arena، GPSS

الگوهای زبان

6. زبان‌های اسکریپتی (Scripting):
   - زیرمجموعه‌ای از زبان‌های دستوری، Rapid Prototyping
   - کنارهم گذاردن اجزایی (Components) که خود برنامه‌ی مستقلی به حساب می‌آیند
     - Batch Files < - Shell Scripting
     - Java Scripts ،PHP
     - Python ،Perl
     - و ...

مرورتوأم تاریخچه والگو



مشخصه‌های یک زبان خوب (بهترین زبان چه زبانی است؟)

1. Syntax Clarity وضوح ساختاری (نحوی)
   - مثال

```
While A>B do S=S+1

When A>B do S=S+1
```

2. Semantic Clarity وضوح مفهوم
   - زبان دارای مفاهیم ساده و روشنی باشد که به راحتی با ترکیب آن‌ها بتوان مفاهیم پیچیده‌تری ایجاد نمود
   - مثال

```
If  A<=B Goto L1
     S=S+1
L1:
```


مشخصه‌های یک زبان خوب

3. Portable
   - قابلیت انتقال بین Platformها
4. محیط برنامه‌سازی مناسب
   - وجود Editor یا نرم‌افزارهای Traceکننده و مستندات کامل
   - مشکل Free Sourceها
5. داشتن امکان Abstraction
   - تعریف تابع، روال، ...
6. تطبیق با کاربرد
   - Lisp برای پردازش لیست
   - C کار سیستمی

مشخصه‌های یک زبان خوب

7. سادگی
   - کم بودن تعداد دستورات و حالات و ...
   - سهولت در تغییر
8. هزینه
   - هزینه ایجاد
   - هزینه کامپایل
   - هزینه تست و غلط گیری
   - هزینه نگهداری
   - هزینه اجرا
   - هزینه‌ی کامپایل و اجرا به هم وابسته‌اند
     ثابت = هزینه‌ی اجرا×هزینه‌ی کامپایل

مشخصه‌های یک زبان خوب

9. Orthogonality (تعامد)
   - بتوان ویژگی‌های مختلفی از یک زبان را با هم ترکیب کرد و ترکیب حاصل هم با معنی باشد
   - مثال:

If (a+b>c+d) then

   - ترکیب دستور محاسباتی با if

انواع زبان از نظر کاربرد

1. زبان‌های پردازش تجاری
   - Cobol ،C#
2. زبان‌های کاربرد علمی
   - Matlab ،Fortran
3. زبان‌های سیستمی
   - اسمبلی، PL/1، C
4. زبان‌های هوش مصنوعی
   - Prolog
5. زبان‌های اسکریپتی
   - کامپایل نمی‌شوند بلکه بیشتر تفسیر می‌شوند: Perl ،Shell Script

پرسش؟