تاریخچه زبان های برنامه نویسی

باز نویسی و ویراش : سید احمد الهی

ایمیل : Ahmad@GoMail.ir

وب سایت ناشر :www.ProgForum.ir

سال انتشار : 1392

فهرست مطالب

مقدمه

برنامه نویسی چیست ؟

تاریخچه آغاز برنامه نویسی

نسل های زبان های برنامه نویسی

تاریخچه ی پیدایش زبان های برنامه نویسی شی گرا

تکامل زبانهای برنامه نویسی

منابع

مقدمه

زبان‌های برنامه‌نویسی ساختارهای زبانی دستورمداری در رایانه‌ها هستند که به‌وسیلهٔ آنها می‌توان یک الگوریتم را به‌وسیلهٔ ساختارهای دستوری متفاوت برای اجرای رایانه توصیف کرد و با این روش امکان نوشتن برنامه جهت تولید نرم‌افزارهای جدید بوجود می‌آید.

معمولاً هر زبان برنامه‌نویسی دارای یک محیط نرم‌افزاری برای وارد کردن متن برنامه، اجرا، همگردانی و رفع اشکال آن هستند. عموماً زبانهای برنامه نویسی را به پنج نسل تقسیم می‌کنند:

نسل اول زبان ماشین - زبان صفرو یک
نسل دوم زبانهایی مانند اسمبلی -قابل فهم تر برای انسان
نسل سوم زبانهایی مانند کوبول و پی ال وان و... -دستورات قابل فهم تر برای انسان و نیاز به کمپایلرها
نسل چهارم مثل زبانهای اوراکل و فاکس پرو و اس کیو الها - نزدیک به محاوره‌های انسانی
نسل پنج زبانهایی مانند - ops5 , prolog تمرکز بر حل مسئله و استفاده از الگوریتمهای نوشته شده توسط برنامه نویس
یک زبان برنامه نویسی یک زبان مصنوعی است که برای بیان محاسباتی که توسط یک ماشین (مخصوصا رایانه) قابل انجام است، طراحی شده‌است.

زبان‌های برنامه نویسی برای ایجاد برنامه‌هایی به کار می‌روند که رفتار یک ماشین را مشخص می‌کنند، الگوریتم دقیق را بیان می‌کنند، و یا روشی برای ارتباط انسانند.

بسیاری از زبان‌های برنامه نویسی تعدادی قالب از ویژگی‌های نوشته شده دستوری (syntax)و معناشناسی (semantics) دارند، چرا که رایانه‌ها دستورات دقیقا مشخص نیاز دارند.

برخی توسط سند خصوصیات (specification document) تعیین شده‌اند. (برای مثال یک استاندارد (ISO، در حالی که برخی دیگر دارای پیاده سازی غالبی می‌باشند.(مانند (Perl اولین زبان برنامه نویسی به قبل از اختراع رایانه باز می‌گردد، و برای هدایت رفتار ماشین‌هایی مانند دستگاه‌های نساجی اتوماتیک و نوازنده‌های پیانو به کار می‌رفت.

هزاران زبان برنامه نویسی خلق شده‌اند، بیشتر در زمینهٔ رایانه، زمینه‌ای که هر ساله بسیاری دیگر ایجاد می‌شوند.

برنامه نویسی چیست؟

برنامه نویسی را می توان به یک بازی هوش تشبیه کرد ، بازی بر روی داده ها و متغیرها با استفاده از دستورات و ابزارهای برنامه نویسی که در اختیار ما گذاشته شده است .

در این بازی ابتدا باید ابزراهای مورد نیاز خود ، جهت نوشتن برنامه ای خاص را انتخاب کرده سپس باید ابزارها را به گونه ای در کنار هم قرار دهیم و به گونه ای با ابزارها بر روی داده ها و متغیرها کار کنیم تا به هدف مورد نظر برسیم .

بنابراین باید در ابتدا ، شناخت کافی بر روی ابزارها و کاربرد آنها داشته باشیم که تجربه نشان داده این یادگیری برای دانشجویان ، چندان دشوار نیست و اکثر دانشجویان قادر به درک ابزارها و دستوراتی مانند if یا while و غیره می باشند .

اما نکته مهم این است که پس از تسلط بر روی ابزارها ، تازه بازی آغاز می شود و ما تنها بر قوانین بازی و ابزارهای موجود شناخت پیدا کرده ایم .

اینجاست که برخی دانشجویان دچار مشکل می شوند و قادر به استفاده مناسب از ابزارها و بسط دادن آنها در کنار یکدیگر جهت رسیدن به هدف نهایی برنامه نمی باشند .

در این مرحله دانشجویان باید دارای یک روحیه الگوریتمی شوند بدین معنی که توانایی تفکیک مراحل ، جهت رسیدن به پاسخ را داشته باشند .

کمتر دانشجویی است که بدون تمرین و ممارست به این روحیه دست پیدا کند .

از علائم ورود یک دانشجو به دنیای برنامه نویسی شوق و اشتیاق او جهت یافتن تمرینهای جدید برنامه نویسی است و چنین دانشجویی با پشتکار خود می تواند یک برنامه نویس حرفه ای شود.

تاریخچه آغاز برنامه نویسی

قدیمیترین نمونه عملی از برنامه نویسی به سال 1801 در کشور فرانسه توسط شخصی بنام جکارد برمی گردد.

او یک دستگاه بافندگی طراحی کرده بود که می توانست اعمال خاصی از بافندگی را که روی کارتهای سوراخ شده (Punched card) ثبت شده بودند به ترتیب انجام دهد.

این تکنولوژی اجازه تولید بافت های پیچیده و با کیفیت تر را به کارگران معمولی می داد. اکنون به جای کارگران متعدد و ماهر، فقط یک نفر برای مدیریت کل دستگاه کافی بود. روند تقریبا مشابهی نیز در خلال انقلاب صنعتی در انگلیس به وقوع پیوست.

به دنبال استفاده از ماشینهای خودکار و کاهش نیروی کار انسانی، جنبشی تحت عنوان لودیت ها به راه افتاد.

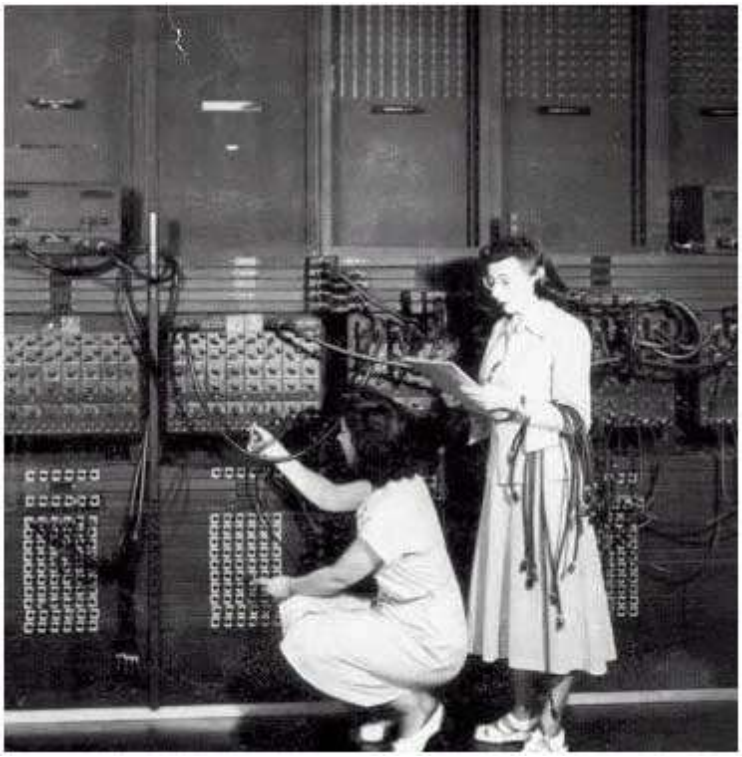
این جنبش متشکل بود از افرادی که مخالف توسعه تکنولوژی بودند و برای جایگاه شغلی، نان و رزق و روزی خود با تکنولوژی می جنگیدند.

در این دوره، برنامه نویسی به موازات ساخت ماشین های جدید توسط پاسکال، بابیج، شوالتز و ... توسعه می یافت. برای حل هر مشکلی یک ماشین جدید ساخته می شود که چرخ دنده ها، سیم ها، میله ها و دستورات جدیدتری نسبت به قبل داشت.

در سال 1847 یک ریاضیدارن انگلیسی بنام George Boole با ارائه جبر منطقی، ارتباط میان علم ریاضیات و منطق را اثبات کرد. برای اولین بار او اثبات کرد که علم منطق بیشتر بخشی از ریاضیات است تا فلسفه! این عمل که یک گام بزرگ در تفکر بشریت به حساب می آید تقریبا صد سال (تا سال 1940) طول کشید تا در محاسبات کامپیوتری به کار گرفته شود.

در دهه 1940 اولین کامپیوتر الکترونیکی ساخته شد.

حافظه محدود و سرعت بسیار پایین (به نسبت سرعت کامپیوترهای امروزی) از ویژگی های بارز کامپیوترهای الکترونیکی اولیه بودند. در این سالها که به عصر اطلاعات معروف بود، اجرای برنامه ها در کامپیوترها مستلزم تنظیم کلیدها، سوییچ ها و اتصال سیم های مختلف بود که منطق برنامه را مشخص می کردند.



در چنین حالتی برنامه نویسی چیزی به جز تنظیم تعداد بسیار زیادی سیم نبود .یک محاسبات چند دقیقه ای نیازمند صرف روزها وقت برای اتصال میان سیم ها، کلیدها و پورت ها بود.

برای هر عملی سخت افزار خاصی طراحی شده بود که می بایست توسط سیم ها و پورتها به یکدیگر متصل می شدند تا برنامه اصلی را شکل دهند.

جان وان نیومن در سال 1945 هنگامی که در انیستیتو تحقیقات عالی مشغول بکار بود، 2 اصل اساسی را ارائه کرد که تاثیر مستقیمی بر مسیر توسعه برنامه نویسی گذاشت؛

- اولین اصل"*تکنیک برنامه مشترک* (Shared-program technique) "نام داشت. بر طبق این اصل سخت افزار کامپیوترها باید بسیار ساده باشند و نیازی نیست که برای هر برنامه به صورت دستی سیم کشی و تجهیز شوند. در عوض، دستورات نرم افزاری باید از ترکیب همان دستورات ساده سخت افزاری تشکیل شوند و یک دستور نرم افزار مرکب و پیچیده تر را بسازند. این اصل سخت افزارها را ساده و دستورات نرم افزاری را پیچیده می کرد .

- وان نیومن اصل دوم خود را" *انتقال کنترل به صورت شرطی* (Conditional Control) " (Transfer)نامید. بر طبق این اصل که تاثیر عمیقی بر توسعه برنامه نویسی گذاشته است، برنامه کامپیوتری می تواند شامل بلاک های کوچکتری از دستورات بنام زیرروال (Subroutine) شود که می توانند در هر جای برنامه اصلی جای گیرند. این بلاک ها می توانند هر چندبار که لازم است در هر کجای برنامه اصلی استفاده شوند. بخش مهم دیگری از این اصل، کنترل روند برنامه کامپیوتری توسط دستورات منطقی مانند IF…Then و حلقه هایی مانند FOR را امکانپذیر می کرد .

سال 1948 همزمان بود با ارائه *تئوری ریاضی ارتباطات* توسط کلود شانن (Claude Shannon, 1916-2001).ارائه تز تئوری ریاضی ارتباطات او اساس تئوری اطلاعات (Information Theory) را شکل داد. این تئوری نحوه استفاده از منطق باینری (اعداد در مبنای دو، همان صفر و یک ها) در محاسبات نرم افزاری در کامپیوترها را بیان می کرد. تز ارائه شده توسط شانن دنیای صفر و یک ها)باینری) که اساس و پایه کامپیوترهای الکترونیکی امروزی هستند را شکل می داد. بعدها اولین کامپیوتر باینری در حین جنگ جهانی دوم توسط یک آلمانی بنام زوس ساخته شد.

در سال 1949، پس از ارائه 2اصل وان نیومن، زبان برنامه نویسی بنام کد کوتاه (Short Code) به وجود آمد. کد کوتاه که دستورات آن به صورت یک سری صفر و یک تشکیل شده بود، اولین زبان برنامه نویسی برای کامپیوترهای الکترونیکی به حساب می آید. ساخت این زبان برنامه نویسی به عنوان اولین قدم در جهت ساخت دنیای برنامه نویسی امروز به حساب می آید.

در سال 1951، گریس هاپر اولین کامپایلر دنیای کامپیوتر را به نام A-0 نوشت .کامپایلر به برنامه ای گفته می شود که دستورات برنامه نویسی را به دستورات صفر و یک (در آن زمان دستورات زبان کد کوتاه) تبدیل می کند. از آن پس برنامه نویس نیازی به ورود کدهای صفر و یک نداشت و توسعه دنیای برنامه نویسی سرعتی چند برابر یافت.

پس از آن فورترن (FORTRAN) به عنوان اولین زبان برنامه نویسی به شکل امروزی در سال 1957 توسط شرکت IBM ساخته شد. نام آن مخفف سیستم تبدیل فرمول (FORmula TRANslating system) بود .دستورات آن بسیار ساده بودند و فقط شامل دستورات IF ، DOو GOTO می شدند .هرچند دستورات این زبان امروزه بسیار محدود و ناکارآمد تلقی می شود اما در آن زبان قدم بسیار بزرگی رو به جلو بود. انواع داده ای نیز برای اولین بار در زبان FORTRAN ارائه شدند. داده هایی مانند متغیرهای منطقی، صحیح، حقیقی و اعداد با دقت مضاعف در FORTRAN قابل تعریف و استفاده بودند.

اگرچه زبان FORTRAN در کار با اعداد و داده ها بسیار خوب عمل می کرد اما در دریافت ورودی و صدور خروجی بسیار محدودیت داشت. با شروع استفاده از کامپیوتر در دنیای تجارت، زبانهایی مانند COBOLنیز پا به عرصه ظهور گذاشتند COBOL .که از ابتدا به منظور استفاده در حوزه برنامه های تجاری طراحی شده بود، فقط شامل انواع داده ای عددی و رشته متنی می شد. امکان گروه بندی داده ها در آرایه ها و رکوردها، مدیریت داده ها را تسهیل می کرد. این امکانات فضا را برای رشد نرم افزارهای ذخیره و بازیابی اطلاعات فراهم کرد.
تا سال 1972 بیش از 200 زبان برنامه نویسی مختلف ظهور کرد. اکثر این زبانها به منظور کاربرد خاصی توسعه یافته بودند، برخی دیگر نیز ویرایش جدیدی از زبانهای قبلی بودند.

نسل های زبان های برنامه نویسی

با توجه به مطالعات پیشین در زمینه روند توسعه ابزارهای برنامه نویسی، می توان این ابزارها را به چند نسل مختلف تقسیم بندی کرد:

نسل اول

در سالهای دهه 1950 برنامه نویسی کامپیوترهای اولیه توسط تغییر سیم ها و تنظیم هزاران کلید و سوییچ انجام میشد. در برخی موارد این تنظیمات بر روی کاغذهای طومار گونه و یا کارت های سوراخشده نوشته می شدند که به کامپیوتر می گفتند چه کاری را(What) ، به چه صورت (How) و در چه زمانی (When) انجام دهد.
به منظور اجرای یک نرم افزار، برنامه نویس باید اطلاعات جامع و کاملی از کامپیوتر موردنظر می داشت. یک اشتباه کوچک منجر به شکست در کل برنامه کامپیوتری میشد.

نسل دوم

در این دوره افراد به دنبال ابزارهای سریعتر و راحتتری برای برنامه نویسی بودند. نتیجه این تلاشها تولد نسل دوم زبان های برنامه نویسی در اواسط دهه 1950 شد. در این نسل از نمادها به جای دستورات صفر و یک استفاده می شد.

نسل سوم

در اواخر دهه 1950 مفسرهای زبان های طبیعی و کامپایلرهای پا به عرصه ظهور گذاشتند. قدیمیترین زبان برنامه نویسی این نسل FORTRAN است که در سال 1953 توسط IBM ساخته شد. در سال 1959 زبان برنامه نویسی COBOL به منظور استفاده در دنیای نرم افزارهای تجاری عرضه گردید.
زبانهای سطح بالای برنامه نویسی مانند BASIC ، PASCAL، ALGOL، PL/Iو C در این دوره معرفی شدند .

نسل چهارم

زبانهای این نسل برنامه نویس را قادر می سازند تا کارهای سطح بالاتر و بیشتری را توسط کد کمتری انجام دهد. هر دستور از زبانهای این نسل معادل صدها دستور از زبانهای نسل سوم است. برنامه هایی که توسط این زبانهای نوشته می شوند، نیاز به یک محیط سخت افزاری خاص و امکانات خاص برای اجرا شدن دارند. در دهه 1990درخواست ها برای استفاده از این زبانها بسیار زیاد شد و کمپانی هایی مانند Oracle و SUN تلاشهایی در این زمینه انجام دادند.

نسل پنجم

این نسل از زبانهای برنامه نویسی هنوز در مرحله تئوری هستند و تا به امروز نمونه عملی از آنها ساخته نشده است. بسیاری از تلاشها به دلیل محدودیت سخت افزارها، عملیاتی نشده اند. استفاده از زبان طبیعی و روزمره برای تفهیم کارها به کامپیوترها از ویژگی های بارز این نسل به شمار می رود.
استفاده از شبکه های عصبی و هوش مصنوعی و همچنین استفاده از Agent ها به منظور انجام کارها در کامپیوتر از دیگر ویژگی های این نسل از زبان ها هستند.

تاریخچه ی پیدایش زبان های برنامه نویسی شی گرا

در ابتدای پیدایش علوم کامپیوتر، برنامه‌نویسان کدهایی در سطح ماشین می‌نوشتند.
به همین دلیل بیشتر توجه آنان معطوف به مجموعه دستورات ماشین بود. به تدریج زبان‌های سطح بالا ایجاد شد و در نتیجه توجه برنامه‌نویسان بیشتر به اصل مسئله معطوف گردید.
اکنون سطح انتزاعی بر روی کامپیوترهای مختلف ایجاد شده است. یعنی برنامه‌ی نوشته شده روی هر ماشین اجرا می‌شود .

در زبان‌های ساخت‌یافته ، برنامه را به تعدادی روال تقسیم می‌نمودند، بدین صورت که هر روال کار خاصی را انجام می‌داد. برنامه‌نویسی شی‌گرایی اجازه می‌دهد تا سیستمی دارای اشیای مرتبط و همکار داشته باشید.
کلاس ها این امکان را فراهم می‌کنند که جزییات پیاده‌سازی را پشت واسط برنامه‌نویسی پنهان نمایید.
چندشکلی یا چندریختی ، رفتار و واسط مشترکی را برای مفاهیم مشابه نشان می‌دهد.
بدین وسیله قادر خواهید بود تا پیمانه‌های خاص و جدیدی را بدون نیاز به دست‌کاری در پیاده‌سازی مفاهیم پایه ایجاد نمایید.

در ابتدای پیدایش علوم کامپیوتر، برنامه‌نویسان کدهایی در سطح ماشین می‌نوشتند. به همین دلیل بیشتر توجه آنان معطوف به مجموعه دستورات ماشین بود. به تدریج زبان‌های سطح بالا ایجاد شد و در نتیجه توجه برنامه‌نویسان بیشتر به اصل مسئله معطوف گردید. اکنون سطح انتزاعی بر روی کامپیوترهای مختلف ایجاد شده است. یعنی برنامه‌ی نوشته شده روی هر ماشین اجرا می‌شود .

روش‌های برنامه‌نویسی و زبان‌ها در واقع راه ارتباط با ماشین را تعریف می‌کنند. هر روش جدید، شیوه‌های نو را برای تجزیه‌ی مساله ارائه می‌دهد که عبارتند از: کد ماشین، کد مستقل از ماشین، روال‌ها، کلاس‌ها و غیره.

هر شیوه‌ی جدید، نگرشی تازه جهت تبدیل نیازهای سیستم به زیرساخت‌های برنامه‌نویسی ارائه می‌دهد.

تکامل این نوع شیوه‌های برنامه‌نویسی امکانی را فراهم می‌نماید تا سیستم‌های پیچیده‌تری ایجاد کنید. عکس این مطلب نیز صادق می‌باشد. یعنی سیستم‌های پیچیده می‌توانند پیاده‌سازی شوند.

اکنون، برنامه‌نویسی شی‌گرا به عنوان روش ایجاد پروژه‌های نرم‌افزاری استفاده می‌شود. این شیوه قدرت خود را در مدل‌سازی رفتارهای معمولی نشان داده است. اما این روش به خوبی نمی‌تواند بر روی رفتارهایی که بین چندین پیمانه مشترک وجود دارند، کار کند. برعکس، شیوه‌ی جنبه‌گرا تا حد قابل توجهی این مشکل را برطرف می‌کند.

در سال 1972 پارانز مفهومی به نام جداسازی دغدغه‌ها را مطرح کرده که امروزه جزء مفاهیم اساسی در فرآیند مهندسی نرم‌افزار به شمار می‌آید. این مفهوم به صورت زیر تعریف شده است:

"قابلیت تشخیص، کپسوله‌سازی و کار با دغدغه، هدف و یا مقصود هستند"

دغدغه را می‌توان به عنوان محرکی برای تقسیم نرم‌افزار به بخش‌های قابل مدیریت درنظر گرفت. برای نمونه، یک وظیفه‌مندی خاص نرم افزار و مسائلی که به خواسته‌های غیروظیفه‌مندی مرتبط می‌شوند مانند ثبت وقایع، امنیت و غیره، همگی به عنوان دغدغه هستند، البته با توجه به جداسازی دغدغه‌ها آنها را در قالب واحدهای مستقل کپسوله کرده‌اند.

در سال 1997، مشهورترین رویکرد زبان جنبه‌گرا به نام AspectJ ابتدا توسط گروهی در Xerox PARC عمومیت یافت. این گروه روی پروتکل‌ها و ایده‌ی مدل‌سازی دغدغه‌های مشترک کار می‌کردند. در همان حال، گروهی در شرکت IBM برنامه‌نویسی موضوع‌گرا را مطرح کردند. برنامه‌نویسی موضوع‌گرا و عناوین بعدی آن، تحت نام "جداسازی چندبعدی دغدغه‌ها"، به جداسازی و ادغام پیمانه‌های مختلف برنامه‌نویسی بر پایه‌ی دغدغه‌هایی در ابعاد مختلف پرداخته‌اند.

نخستین کار در دانشگاه Twente هلند انجام یافت که در مورد فیلترهای ادغام‌سازی کار می‌کردند. به طوری که در پیاده‌سازی فیلترهایی که رفتار شی را در اجرا پیشرفت می‌دادند دخیل بودند. در دانشگاه Northeastern نیز انتزاعی از ساختار کلاس‌ها انجام گرفت که پشتیبانی بهتری از مفهوم دانش و رفتار عملیاتی ارائه می‌داد. در سال 1997، کریستیانا لوپز از هر دو مقاله استفاده کرد و زبان D-Java را به عنوان اولین مجموعه‌ی رسمی از زبان جنبه‌گرا ارائه نمود.

شیوه‌ی موضوعی اولین روشی بود که مفاهیم جنبه‌گرایی را با زبان مدل‌سازی یکپارچه ادغام کرد. این کار مشترکی از چندین گروه با گروه برنامه‌نویسی موضوع‌گرا است. برنامه‌نویسی موضوع‌گرا به طراحی موضوع‌گرا تبدیل شده و در سال 2001 به Theme/UML تبدیل گردید. تعریف و نمایش دغدغه‌ها برای نخستین بار در مستندات الیسا و گیل مورفی از دانشگاه British Columbia ارائه شد و در سال 2003 به عنوان بخشی از شیوه‌ی موضوعی طراحی جنبه‌گرا به نام Theme/Doc مطرح گردید.

حدود یک دهه‌ی قبل، به دنبال موفقیت درخور توجه ابزار CASE ، چیکوفسکی و کراس مبحث مهندسی معکوس و بازیابی طراحی را مطرح نمودند. تعریفی که آنها از مهندسی معکوس داشتند در زیر ارائه شده است:

"مهندسی معکوس، تحلیل یک سیستم به منظور تشخیص اجزا، ترکیبات فعلی، روابط بینابین آنها، استخراج و تولید تجریدهای موجود در سیستم و داده‌های مربوط به طراحی است."

در دو دهه‌ی قبل، محققان امکاناتی را به منظور کشف، اعمال تغییر، تحلیل، جمع‌بندی، تولید، تجزیه و به تصویر کشیدن محصولات نرم‌افزاری ابداع کردند.

این امکانات شامل تهیه‌ی اسناد نرم‌افزاری در شکل‌های گوناگون، نمایش کد میانی، داده و معماری بود.

اغلب ابزارهای مهندسی معکوس بر استخراج ساختار درونی سیستم موجود با هدف انتقال آن به ذهن مهندس نرم افزار تمرکز دارد. در هر صورت، این ابزارها راه زیادی در پیش دارند تا به مرحله‌ای برسند که مورد استفاده‌ی روزانه‌ی مهندسان نرم‌افزار قرار گیرند. مطالعه و درک برنامه در صنعت نرم‌افزار به منظور کنترل هزینه و ریسک چرخه‌ی تحولات سیستم‌های نرم‌افزاری از اهمیت بالایی برخوردار می‌باشد.

تکامل زبانهای برنامه نویسی

سال: ۱۹۵۷

زبان برنامه نویسیFortran :

FORTRANمخفف کلمه (FORmula TRANslation) به معنی ترجمه گر فرمول ها بود و قدیمی ترین زبانی هست که هنوز مورد استفاده قرار می گیرد.

فرترن که توسط آقای جان باکوس (John Backus) به وجود آمد در ادامه به منظور انجام محاسبات سطح بالا در مسائل مهندسی، ریاضی و آمار توسعه یافت.

این زبان هنوز در سازمان های فضایی، صنایع خودروسازی و سازمان های دولتی و موسسات تحقیقاتی مورد استفاده قرا می گیرد.

نمونه کاربرد زبان برنامه نویسی: سرویس ملی هواشناسی

نکات تکمیلی:

۱-صفحه کلیدهای به شکل امروزی موسوم به QWERTY نخستین بار در سال ۱۸۷۴ به منظور بالابردن سرعت تایپ حروف در ماشین تحریر ها به وجود آمدند، اما همین صفحه کلید ها علت اصلی ساختار حرفی اکثر زبان های برنامه نویسی کامپیوتری هستند!

۲-چینش حروف در این نوع صفحه کلید ها بر اساس آنالیز موارد احتمالی قفل کردن دکمه های حروف در اثر فشردن سریع بود و طراحی QWERTY بر اساس جدا قرار گرفتن دکمه های نزدیک به هم در دو ردیف مانند T و H صورت گرفت.

سال: ۱۹۵۹

زبان برنامه نویسیCOBOL :

مخفف عبارت Common Business Oriented Language یا زبان عمومی مخصوص تجارت، همانطور که از اسمش معلوم بود پشت پرده اکثر سیستم های مالی و بانکی قرار داشت از جمله سیستم دستگاه های خودپرداز یا ATM ، کارت های اعتباری، همچنین در شبکه های مخابراتی و تلفن های ثابت و سلولی، سازمان ها و نهاد های دولتی و بیمارستان ها، صنایع خودروسازی و حتی در سیستم های ترافیک شهری.

تیم توسعه زبان کوبول به رهبری دکتر گریس موری هاپر (Grace Murray Hopper) در سال ۱۹۵۹ برای ایجاد یک زبان یک دست و کاربر پسند برای معاملات مالی و تجاری تشکیل شد.

نمونه کاربرد زبان برنامه نویسی: سرویس پست ملی ایالات متحده امریکا

نکات تکمیلی:

۱ -در سال ۱۹۳۷ ، کد های باینری موضوع رساله آقای کلاود شانون در زمینه ترجمه متن به کد های ریاضی بود که پایه و اساس اولین رایانه الکترومکانیکی کاملا عملی در سال ۱۹۴۱ به نام زئوس Z3 گردید.

۲ -کامپوترها هنوز از زبان باینری ۰ و ۱ استفاده می کنند، اما دیگر کمتر برنامه نویسی پیدا می شود که هنگام برنامه نویسی مجبور باشد مدام از ۰ و ۱ برای تایپ دستورات استفاده نماید!

سال: ۱۹۶۴

زبان برنامه نویسیBASIC :

زبانی که توسط گروهی از دانشجویان در کالج دارتموث به وجود آمد، مخفف عبارت-Bigginers All purpose Symbolic Instruction Codeیا زبان همه منظوره سمبلیک سطح مبتدی بود که به منظور ارائه یک زبان ساده شده برای استفاده افرادی که پیش زمینه قوی در زمینه ریاضیات یا اطلاعات فنی نداشتند طراحی شد.

نسخه بهینه سازی شده ای از بیسیک که توسط بیل گیتس و پل آلن نوشته شده بود به عنوان اولین محصول شرکت نوپای مایکروسافت حالت تجاری به خود گرفت.

این محصول به M.I.T.S برای توسعه محصول این شرکت با نام Altair فروخته شد.

نمونه کاربرد زبان برنامه نویسی: نمونه اصلاح شده بیسیک در سال ۱۹۷۷ به عنوان موتور راه انداز سیستم عامل رایانه اپل ۲ مورد استفاده قرار گرفت.

نکات تکمیلی:

۱ -هم اکنون بیسیک بیش از ۲ میلیون خط کد مورد استفاده دارد، در حالی که این رقم در سال ۱۹۷۵ تنها ۴۰۰۰ خط بود.

سال: ۱۹۶۹

زبان برنامه نویسیC :

زبان سی بین سال های ۱۹۶۹ و ۱۹۷۳ توسط دنیس ریتچی (Ritchie) در لابراتور تلفن بل برای استفاده در سیستم های یونیکس توسعه یافت. این زبان به این علت C نام گرفت که ویژگی های اصلی خود را از زبان دیگری که B نام داشت گرفته بود و در واقع نسخه اصلاح شده آن زبان بود.

این زبان به قدری قدرتمند بود که در سال های بعد اکثر سیستم ها با هسته یونیکس به زبان سی بازنویسی شدند-یونیکسی که از اولین هسته های سیستم عامل بود که در زبانی غیر از اسمبلی به کار گرفته شد.

نمونه کاربرد زبان برنامه نویسی: سیستم عامل لینوکس امروزی بر مبنای C نوشته شده.

سال: ۱۹۷۰

زبان برنامه نویسیPASCAL :

زبانی که به افتخار بلیز پاسکال دانشمند معروف فرانسوی به این نام نهاده شد، دانشمندی که اولین ماشین حساب را در سال ۱۶۴۱ اختراع کرد. نیکلاس ورث (Wirth) ابتدا زبان را به جهت ارائه یک ابزار آموزشی ایجاد کرد اما در ادامه رشد یافت و کاربرد تجاری به خود گرفت.

نمونه کاربرد زبان برنامه نویسی: شبکه) Skype زبانOBJECT PASCAL)

نکات تکمیلی:

اولین نگارش برنامه word تقریبا ۲۷۰۰۰ خط کد برنامه نویسی داشت. امروزه آخرین نگارش آفیس بیش از ۳۰ میلیون خط کد برنامه نویسی دارد!

سال: ۱۹۸۳

زبان برنامه نویسیC++ :

در آزمایشگاه های بل، آقای بجارن استروستروپ (Bjarne Stroustrup) زبان C را به نسخه C++ ارتقا داد و یکی از محبوب ترین زبان های برنامه نویسی تا زمان حاضر را خلق کرد.

این زبان از سال ۱۹۸۶در لیست ده زبان برنامه نویسی برتر تاریخ رایانه قرار گرفت و در سال ۲۰۰۳ موفق به کسب Hall of Fame گردید.

نمونه کاربرد زبان برنامه نویسی: مایکروسافت آفیس، ادوبی PDF Reader و مرورگر موزیلا فایرفاکس

سال: ۱۹۸۷

زبان برنامه نویسیPERL :

لری وال(Wall) ، یک برنامه نویس یونیکس زبان پرل را پس از آنکه مشغول استخراج داده برای تهیه یک گزارش بود و متوجه شد یونیکس قادر نیست این عملیات را به شیوه مطلوب به انجام رساند به وجود آورد PERL .مخفف عبارت Practical Extraction Report Language یا زبان عملی استخراج گزارش است. آنطور که این زبان توسط مخترعش توصیف شده: زبانی است برای «انجام سریع کار شما!»

نمونه کاربرد زبان برنامه نویسی: استفاده شده توسطCRAIGSLIST

سال: ۱۹۹۱

زبان برنامه نویسیPYTHON :

Monty Pythonکه نام یک برنامه کمدی تلویزیونی بود به عنوان الهام بخش نام این زبان استفاده شد. آقای Guido Van Rossum این زبان را به منظور اصلاح مشکلات موجود در زبان ABC توسعه داد و همچنان به عنوان رهبر تیم طراحی این زبان انجام وظیفه می کند.

نمونه کاربرد زبان برنامه نویسی: مورد استفاده در موتور جستجوی گوگل، یوتیوب و سازمان فضایی ناسا

نکات تکمیلی:

در سیستم عامل مک OS/X 90 میلیون خط کد استفاده شده.

سال: ۱۹۹۳

زبان برنامه نویسیRUBY :

یوکی هیرو ماتز ماتسوموتو این زبان را برای ماه تولد خود که جولای بود رابی نامید. او این زبان را با ترکیبی از قسمت های مورد علاقه خود از زبان های پرل، اسمالتاک(Smalltalk) ،Eiffel، Adaو Lispبه وجود آورد.

نمونه کاربرد زبان برنامه نویسی: استفاده شده توسطBASECAMP

سال: ۱۹۹۵

زبان برنامه نویسیPHP :

راسموس لردورف زبان پی اچ پی را نخستین بار برای جایگزین کردن اسکریپت هایی از زبان پرل برای صفحه وب شخصی خود توسعه داد. امروزه این زبان به قدری توسعه یافته که بخش عظیمی از معماری جهان وب را بر دوش می کشد . از جمله ۲۰ میلیون وب سایت اینترنت.

نمونه کاربرد زبان برنامه نویسی: فیس بوک

نکات تکمیلی:

برای ویندوز ۹۵ حدود ۱۵ میلیون خط کد نوشته شد. این عدد برای ویندوز ۷ بیش از ۵۰ میلیون خط کد است!

سال: ۱۹۹۵

زبان برنامه نویسیJAVA :

تیمی از توسعه دهندگان شرکت سان مایکرو سیستمز به رهبری جیمز گاسلینگ زبان جاوا را برای اجرا در ست-آپ باکس های تلویزیون های تعاملی توسعه دادند. جاوا اکنون روی بیش از ۱,۱ میلیارد رایانه شخصی در سرتاسر جهان کار می کند و بسیاری وب سایت ها بدون ان اصلا کار نمی کنند.

نمونه کاربرد زبان برنامه نویسی: استفاده شده در خودروی مریخ نورد در سال ۲۰۰۴

سال: ۱۹۹۵

زبان برنامه نویسیJAVASCRIPT :

زبان های JAVA و JAVASCRIPT به هم بی ارتباطند و تفاوت های بسیاری با هم دارند.

زبان JAVASCRIPT در اصل توسط برندن ایچ (Eich) در شرکت Netscape و تحت نام Mocha شکل گرفت. جاوا اسکریپت از دستوراتی استفاده می کند که الهام گرفته از زبان C است.

اگر چه می توان از جنبه کلاینت یا مرورگر آن را مورد استفاده قرار داد، اما امروزه بیشتر در سرور ها به عنوان نود js استفاده می گردد. همچنین زبان برنامه نویسی آژاکس (AJAX) نیز وابسته به جاوا اسکریپت است.

نمونه کاربرد زبان برنامه نویسی: مورد استفاده در RACKSPACE سمت کلاینت

سال: ۲۰۰۵

زبان برنامه نویسیRUBY ON RAILS :

زبان RUBY ON RAILS توسط دیوید هینمیر هنسون زمانی که در BACECAMP کار می کرد به وجود آمد، یک ابزار مدیریت پروژه با ۳۷ ابزار.

آقای هنسون ابتدا زبان RUBY ON RAILS را به صورت منبع باز در جولای ۲۰۰۴ منتشر کرد.

اما حقوق استفاده از این پروژه را تا فوریه ۲۰۰۵ در اختیار دیگر برنامه نویسان قرار نداد.

هم اکنون این زبان به نگارش ۳,۰,۷ رسیده و بیش از ۱۸۰۰ شرکت کننده دارد.

منابع

progforum.ir

iranelmi.blogfa.com

mozhganit.blogfa.com

www.articles.ir

gooyait.com