



تصویر: aft.com

## سوال اساسی در مورد زبان C

تجربه کار با C++ را نداشتم و آن موقع، تنها منبعی که می شد برای C++ به او مراجعه کرد، خودم بودم که تازه آن هم تمام وقت نبود. اما در بهترین حالت، دوست داشتم که توابع مجازی، Template ها و استثناها را اگر در سال ۸۵ پیاده می کردم، C++ زبان بسیار بهتری از آب در می آمد.

● وقتی در سال ۱۹۹۸، C++ به یک استاندارد بدل شد، چه احساسی داشتید و چه نقشی را در روند استانداردسازی بازی کردید؟  
سال های زیاد بود که سخت کار می کردم (۱۹۸۹ - ۱۹۹۷)، هر چند که الان دارم روی استاندارد زیرمجموعه اش کار می کنم. اینکه زبان استاندارد را بخواهی محبوب حفظ کنی، کاری دشوار و وظیفه ای سخت است. C++ هیچ پدری ندارد و هیچ کس نیست که برای توسعه اش از جیب بزند، کتابخانه هایش رایگان هستند و رایگان هم بازاریابی می شوند. باز هم دست موسسه ایزو درد نکند که به رشد جامعه C++ کمک کرد و همین رشد باعث شد داوطلبان زیادی از سراسر دنیا داشته باشد.

● جالب ترین برنامه ای که تا به حال دیده ای که با C++ نوشته اند چیست؟

نمی توانم یکی را انتخاب کنم. بیشتر به سیستم های کامل نگاه می کنم که بخشی از آنها را با C++ نوشته اند. مثلاً زیرسیستم خودروهای مریخ نشین ناسا، موتور جستجوی گوگل و سیستم رزرو هوایی آمازون به ذهنم می رسد. اگر فقط بخواهم به کد نگاه کنم، به نظرم STL الکساندر استپانوویچ یکی از مفیدترین، موثرترین و جالب ترین کدهای C++ بود که تا به حال دیده ام.

● آیا فکر نمی کنی که GNU C++ بهتر است خطاهای طولانی کمتری داشته باشد که دانشجویان ترسند؟

البته، اما واقعا تقصیر کامپایلر گنو نیست. مشکل اصلی اینجاست که C++ استاندارد ۹۸ هیچ راه ساده ای برای تعریف یک تمپلیت پیش پای برنامه نویس نمی گذارد. این یکی از ضعف های این زبان است، نه کامپایلر آن! و فقط با تغییر زبان می شود آن را درست کرد که قرار است بخشی از C++0x (استاندارد جدید C++ باشد).

● در آخر چیز دیگری هست که بخواهید اضافه کنید؟  
بله من فکر می کنم که ما باید هنر انتقال مفاهیم پایه ای C++ را هم داشته باشیم. بسیار سخت است که بخواهیم از نظر فنی برنامه را توسعه بدهیم، اما به برنامه نویسان نگویم که قابلیت های فنی آن چیست. از طرف دیگر، خود ویژگی های زبانی کسل کننده اند. برنامه نویسان باید خودشان با ترکیب ویژگی های پایه، به ویژگی جامع تری برسند و این نیازمند زمان است.

اخیرا من از صنعت خارج شدم و به سمت دانشگاهی شدن پیش رفته ام و حالا مشکلات تحصیلی را از زوایای مختلف درک می کنم. ما باید سطح سواد توسعه دهندگان نرم افزارمان را بالا ببریم. طی سه سال گذشته، شیوه جدید تدریس برای سال اولی ها طراحی کردم و نتیجه آن کتابی شده است با عنوان برنامه نویسی، مفاهیم و تمرین با استفاده از C++.

منبع

<http://www.computerworld.com>

● تا به حال به موانع یا مشکلاتی برخوردیده اید که توسعه زبان را به خطر بیندازد؟

تا دل تان بخواهد! قوانین اصلی طراحی برنامه چیست؟ چه چیزی باید در زبان باشد و چه چیز نباید باشد؟ اغلب مردم به دنبال زبانی می گردند که بسیار کوچک باشد و در عین حال تمام امکاناتی که آنها به دنبالش هستند را هم در خود داشته باشد، خوب این غیرممکن است. بعد از مدت کوتاهی که به شانس و اقبال اکتفا کرده بودم، بالاخره به این نتیجه رسیدم که باید به قانونی مشخص برسم. زبانی که در آن واحد هم پایدار باشد و هم برای برنامه نویسی سیستم (همچون زبان C) مناسب باشد. واضح است که هیچ برنامه ای نمی تواند هر دو را به طور کامل داشته باشد. اما هدف C++ این بود، حالا چقدر به آن نزدیک شدیم، بعدا مشخص شد.

● فرق بین زبان C همراه با کلاس که گفتید و C++ چیست؟  
اصل فرق در تکنیک پیاده سازی آن نهفته است. C همراه با کلاس را یک پیش پردازنده پیاده سازی می کرد، اما C++ به یک کامپایلر درست و حسابی نیاز داشت (که من نوشتم). انتقال برنامه ها از C همراه با کلاس به C++ بسیار ساده بود اما این دو زبان ۱۰۰ درصد سازگار نبودند. از دیدگاه زبانی اگر به این سوال نگاه کنیم، شاید به این پاسخ برسیم که C++ از نظر توابع مجازی بهتر عمل می کرد. ارزش گفتن دارد که مدیریت منابع، سازنده ها و ویرانگرهای C++ درست همان هایی بودند که در C همراه با کلاس از آن استفاده شد. از طرف دیگر، Template ها و استثناها در نسخه های بعدی C++ (۱۹۸۹) معرفی شد.

● اگر الان شانس این را داشتید که C++ را دوباره اختراع کنید، کدام بخش آن را تغییر می دادید؟

عجب سوال کلیشه ای پرسیدید، آن موقع خب معلوم است که ۳۰ سال

امیر بهالالدین سبط الشیخ  
مجله الکترونیکی کامپیو ترورد با مخترع C++، بیارن استراستروپ (Bjarne Stroustrup) مصاحبه ای انجام داده که تقریبا تمام جوانب این زبان برنامه نویسی را از دید سازنده آن بررسی کرده است، از میان این مصاحبه طولانی، چند سوال را انتخاب کردیم که دانستن آنها در مورد این زبان برنامه نویسی محبوب خالی از لطف نیست:

● چه کسی C++ را شروع کرد؟  
من به دنبال ابزاری بودم که بتوانم نگارش توزیع شده از هسته یونیکس را طراحی و پیاده سازی کنم. در آن سال (۱۹۷۹) چنین چیزی وجود نداشت. به ابزاری نیاز داشتم که می توانست ساختار یک برنامه را بیان کند و مستقیما با سخت افزار سرو کار داشته باشد و به درد برنامه نویسی های اساسی و جدی سیستم بخورد.

● نام C++ از کجا آمده است؟  
ابتدای کار، اسمش C with Classes بود. بسیاری نمی توانستند آن را تلفظ کنند و کم کم به اسم C معروف شد. اما گویا متوجه نبودند که C صدازدن این برنامه یعنی اشاره کردن به برنامه دنیس ریچی (Dennis Ritchie) نه برنامه من، برای همین اسم های مختلفی برای زبان ها گذاشتند. سی قدیم، سی بهتر، و چنین چیزهایی... تا این که یک روز از طرف آزمایشگاه های بل (Bell Labs) برای من نامه آمد که یک اسم درست و حسابی برای این زبان انتخاب کنیم که ما آن موقع به C++ بسنده کردیم. انتخاب زیادی نداشتیم، یک لیست بود که به ما دادند و من از میان آنها همین را انتخاب کردم. البته بیشتر موافق بودند که C++ را انتخاب کنیم، اما به نظر من تلفظش دیگر بسیار سخت می شد.

## روش تبدیل تاریخ میلادی به شمسی

از ۷۹ کمتر است. مساله اصلی اینجاست که اگر سال قبل میلادی کیبسه باشد، فاصله سال جاری تا اولین روز دی ماه، ۱۱ روز است. یعنی یازدهم سال بعد، اول ژانویه است. اگر باقیمانده تقسیم صفر بشود، روز را ۳۰ محاسبه می کنیم و ۹ ماه به خارج قسمت اضافه می کنیم. اگر باقیمانده صفر نشود، ماه شمسی را با ۱۰ جمع می کنیم و باقیمانده، روز ماه است. در زیر فهرستی از چندین سورس تبدیل تاریخ میلادی به شمسی وجود دارد:

- <http://www.farsiweb.info/jalali/jalali.php>
- <http://ehbrahmkhodaei.blogfa.com/post-18.aspx>
- <http://radcom.ir/weblog/fatemeh/archive/2005/04/17/244.aspx>

دقیق آن ماه کافی است در مرحله اول، ۷۹ روزمان را کم کنیم که درست در اول فروردین بایستیم. حالا باید حساب شش ماه اول را از شش ماه دوم جدا کنیم. چون در شش ماه اول، شش ماه ۳۱ روزه بدون کاستی وجود دارد. برای همین اگر روز ما کمتر از ۳۱×۶ بود، در شش ماه نخست سال است، و اگر نه، در شش ماه دوم سال. محاسبه روز و ماه آن ساده است و به عهده خودتان می گذاریم.

نکته: فاصله سال میلادی و سال شمسی ۶۲۱ سال است.

● فاصله روز تا روز اول سال که در آن قرار داریم،

روز با آغاز سال میلادی فاصله داریم.  
برای این که بفهمیم سال کیبسه است یا خیر، از این شرط استفاده می کنیم:  
سال میلادی باید هم بر ۱۰۰ و هم بر ۴۰۰ بخش پذیر باشد؛ یا این که بر ۱۰۰ بخش پذیر نباشد، ولی بر ۴ بخش پذیر باشد. یعنی سال کیبسه داریم.  
برای محاسبه تاریخ شمسی، کلا دو حالت بیشتر موجود نیست:

● فاصله روز تا روز اول سال که در آن قرار داریم، از ۷۹ بیشتر است. در این حالت، قطعا در ماه های بعدی فروردین قرار داریم. برای محاسبه تاریخ

تبدیل تاریخ میلادی به شمسی، از آنجایی که اغلب سیستم های نرم افزاری و زبان های برنامه نویسی پشتیبانی از نوع تاریخ ما را ندارند، همواره یکی از دغدغه های اصلی برنامه نویسی به شمار می آید. شیوه های مختلفی برای تبدیل یک تاریخ میلادی به شمسی وجود دارد و همچنین به لطف برنامه نویسان سخاوتمند ایرانی، نمونه کدهای زیادی به بیشتر زبان ها موجود است. در ادامه شیوه ساده ای از چگونگی تبدیل تاریخ میلادی به شمسی را شرح خواهیم داد. برای تبدیل تاریخ میلادی به شمسی، ابتدا بهتر است روزهای نخست هر دو سال را بدانیم. سال میلادی اول ژانویه، که تقریبا ۱۰ دی ماه است، شروع می شود. بنابراین اولین روز فروردین ماه، ۷۹