به به! میبینم که به برنامه نویسی هم علاقمند شدی و میخوای قدم در این وادی بذاری. منم خوشحال میشم که تا جایی که میتونم راهنماییت کنم. ☺

درباره تفاوت زبان های برنامه نویسی باید بگم که بر اساس معیار های گوناگون، زبان های برنامه نویسی رو در طبقه بندی های گوناگونی قرار میدن؛ یکی از این طبقه بندی ها، بر اساس مفسری (Interpreter) یا کامپایلری (compiler) بودن زبان هاست.

اگه بخوام درباره تفاوت این زبان ها توضیح بدم، اول باید یه مروری از مسیر برنامه هایی که با زبان های برنامه نویسی مینویسیم داشته باشیم. همونطور که میدونی، دستوراتی که ما با استفاده از زبان های برنامه نویسی به کامپیوترمون میدیم، مجموعه از حروف و اعداد و کاراکتر های گوناگونه، اما *کامپیوتر که این چیزا سرش نمیشه! کامپیوتر فقط زبون صفر و یک میفهمه و بس!* پس باید یه سری برنامه ها وجود داشته باشن که دستورات ما رو به زبان باینری ترجمه کنن. این برنامه ها، بر اساس نحوه انجام کار، به دو دسته مفسری و کامپایلری تقسیم بندی میشن، و هر زبان برنامه نویسی، در یکی از این دسته ها قرار میگیره.

دسته اول، زبان های کامپایلری هستن که کامپایلر های این زبان ها (همون برنامه های مترجمی که دربارش صحبت کردیم)، مجموعه همه دستورات رو میگیرن و ابتدا اونها رو به یه زبان سطح پایین تر (احتمالن اسمبلی) تبدیل میکنن و در مرحله دوم هم نتیجه مرحله قبل رو به زبان کامپیوتر ترجمه میکنن. (درباره سطح بندی زبان های برنامه نویسی هم صحبت میکنیم به زودی)

دسته دوم، زبان های مفسری هستن که برنامه مفسر، دستورات برنامه رو به شکل خط به خط (و نه یه جا) میگیرن و اجرا میکنن؛ که البته این ویژگیشون باعث میشه تا خیلی وابسته به سخت افزار نباشن و برنامه های این نوع از زبان ها، قابلیت اجرا بر روی سیستم عامل های گوناگون رو دارن!

به غیر از نحوه اجرا و مستقل بودن یا نبودن از سیستم عامل، زبان های مفسری و کامپایلری تفاوت های دیگه ای هم دارن، مثلن یکی از عیب های زبان های مفسری اینه که مقدار زیادی از CPU و RAM رو اشغال میکنن و سرعت اجراشون هم پایین تره (این موضوع به خصوص توی برنامه هایی که با داده های زیادی سر و کار دارن خودشو نشون میده)؛ دلیل این موضوع هم همون خط به خط اجرا شدن برنامه ست.

در عوض، رفع کردن خطا های برنامه در زبان های مفسری خیلی راحت تره، چون برنامه خط به خط اجرا میشه و به سادگی میشه تشخیص داد مشکل برنامه کجاشه. البته این روزا، محیط های توسعه برنامه نویسی (IDE ها)، به ما این قابلیت رو میدن که برنامه های کامپایلری رو هم به شکل خط به خط اجرا کنیم که این موضوع خیلی به ما در رفع خطا های برنامممون کمک میکنه.

زبان های مفسری، مث BASIC، MATLAB، Perl، PHP، Python، Ruby و...

اما زبان های کامپایلری، مث C++، C#،Pascal، Java و...

و اما سطح بندی زبان های برنامه نویسی...

زبان های برنامه نویسی رو بر اساس نزدیکیشون به زبان انسان یا زبان کامپیوتر، سطح بندی میکنیم؛ یعنی هر چی یه زبانی به زبان انسان (البته پدا پهتر بگم، زبان انسان های انگلیسی زبان!) نزدیک باشه، سطح بالا تره و هر چی به زبان صفر و یکی کامپیوتر نزدیک باشه، سطح پایین تره.

در ابتدا که هنوز هیچ زبان برنامه نویسی وجود نداشت، برنامه نویسا دستوراتشون رو با زبان باینری و با استفاده از کارت های سوراخدار به کامپیوتر هاشون میدن؛ تا اینکه اولین نسل زبان های برنامه نویسی، که این روزا به عنوان سطح پایین ترین زبان های برنامه نویسی ازشون یاد میشه، روی کار اومدن که زبان اسمبلی، معروف ترینشونه! این زبان ها، اونقدری به زبان کامپیوتر نزدیک هستن که یک تناظر یک به یک بین هر یک از اجزای اونها با اجزای زبان باینری وجود داره.

هر چند برنامه نویسی با زبان هایی مث اسمبلی، خیلی ساده تر از صفر و یک هاست، اما هنوزم فاصله زیادی با زبان انسان داره و برنامه نوشتن با این زبان ها، کار هر کسی نیس! حوصله زیاد میخواهد و مرد کهن!!!

چندی بعد از ارائه زبان های سطح پایین مث اسمبلی، زبان هایی ارائه و تولید شدن که زبان های سطح بالا لقب گرفتن، زبان هایی مث C، C++ و Java؛ این زبان ها خیلی به زبان انسان نزدیک تر بودن و کار کردن باهاشون خیلی ساده تر بود، اما این هم آخر راه نبود، و با گذشت زمان و تکمیل تر شدن نسل پشر، دسته دیگه ای از زبان ها وارد گود شدن که از زبان هایی که گفتیم هم سطح بالا تر بودن و با گرفتن لقب سطح بالا از دسته قبلی، اونها رو وارد دسته بندی جدیدی به نام زبان های سطح میانی کردن! زبانهایی مث Python، نسخه های مختلف زبان Basic و...

همونطور که اشاره کردیم، هر چی یه زبان سطح بالاتر باشه، هم یاد گرفتنش و هم استفاده ازش راحت تر و سریع تره، مثلن یه برنامه ای که نوشتنش با پایتون فقط یه نصفه خط کد زدن لازم داره، برای اجرا شدن توی زبانی مث جاوا، تبدیل به چار، پنج خط کد میشه و توی زبانی مث اسمبلی میشه پونزده، بیست خط کد!!!

البته اینم هستش که هر چی یه زبان سطح پایین تر باشه، هم سریع تره و هم قوی تر! چون به طور خیلی مستقیم با پردازنده ارتباط بر قرار میکنه و کنترل پردازنده رو کاملا در دست میگیره! واسه همین واسه بعضی کارای خیلی خفن، مث کرک کردن بازیای خیلی شاخ، فقط میشه از زبان هایی در سطح اسمبلی استفاده کرد (البته اگه میخواین یاد بگیرین، واسه کارای حلال یاد بگیرین، نه واسه کرک! 😐)

با آرزوی موفقیت

اشکان شکیبا