تمرین 1.4

تفاوت مفسر و کامپالر

**کامپایلر چیست؟**

در کامپایلر باید قبل از اجرای برنامه، کدها یکبار به زبان ماشین کامپایل شوند تا برنامه به درستی اجرا شود. هر تغییر کوچکی در کدها اعمال شود، باید یکبار کدها از اول کامپایل شوند. بعد از کامپایل شدن کدها، یک فایل خروجی از برنامه (برای مثال exe) برای اجرا ساخته می‌شود.

اگر خطایی در کدها وجود داشته باشد، کدها کامپایل نمی‌شوند و خروجی فایل تحویل داده نمی‌شود و با خطای Compile error مواجه می‌شویم. وقتی یکبار کد کامپایل شود، خروجی آن فایل، «زبان ماشین» است. برای اجرای این فایل در دفعات بعدی نیاز به کامپایل دوباره نیست چون یکبار این کدها به زبان ماشین ترجمه شده‌اند اما اگر یک تغییر کوچک در کد ایجاد شود، باید دوباره از اول همه‌ی کدها کامپایل شوند تا فایل جدید ساخته شود.

## مفسر چیست؟

**مفسر در برنامه نویسی چیست؟** مفسر به انگلیسی Interpreter نیز مانند کامپایلر، نرم‌افزاری است که کدهای برنامه نویسی را از زبان سطح بالای قابل فهم برای انسان به زبان قابل فهم برای کامپیوتر یا «زبان ماشین» تبدیل می‌کند. **پس تفاوت کامپایلر با مفسر چیست؟**

در کامپایلر، پیش از اجرای برنامه باید کدها به «زبان ماشین» ترجمه شوند اما در مفسر موقع اجرای برنامه به‌صورت خط به خط به «زبان ماشین» ترجمه می‌شوند. در کامپایلرها قبل از اجرای برنامه وقتی کدها کامپایل می‌شوند یک فایل خروجی (مانند exe) به ما تحویل داده می‌شود اما در مفسر، فایل خروجی به ما تحویل داده نمی‌شود بنابراین برای هر بار اجرا، باید خط به خط کدها به زبان ماشین ترجمه شوند. در صورتی که در کامپایلر یک فایل نهایی ایجاد می‌شود که در اجرای بعدی نیازی به ترجمه کد نیست.

منابع

<https://developersho.com/blog/compiler-vs-interpreter>

## شرح تفاوت های مفسر و کامپایلر در برنامه نویسی

نرم افزار مفسر و کامپایلر هر کدام با روش خاصی کدهای نوشته شده را به کدهای قابل درک برای ماشین تبدیل می‌کنند. در این بخش قرار است از همه زوایا تفاوت کامپایلر و مفسر مورد بررسی قرار بگیرد. تفاوت اصلی و واضح مفسر و کامپایلر در این است که کامپایلرها کدهای سطح بالا را قبل از پیاده‌سازی به کدهای ماشین تبدیل می‌کنند و یک فایل اجرایی ( .exe  ) برای آن‌ها می‌سازند. اما مفسر کدهای سطح بالا را هنگام پیاده‌سازی خط به خط برنامه به کدهای قابل فهم برای ماشین تبدیل می‌کند. در‌ بخش‌های بعدی به صورت جزئی‌تری هر یک از تفاوت‌های مفسر و کامپایلر مورد بررسی قرار می‌گیرند.

منابع

<https://blog.faradars.org/%D8%AA%D9%81%D8%A7%D9%88%D8%AA-%DA%A9%D8%A7%D9%85%D9%BE%D8%A7%DB%8C%D9%84%D8%B1-%D9%88-%D9%85%D9%81%D8%B3%D8%B1-%DA%86%DB%8C%D8%B3%D8%AA/>

**Compiler چیست؟**  
[Compiler](https://en.wikipedia.org/wiki/Compiler) یک برنامه کامپیوتری است که کدهای یک زبان‌ برنامه‌نویسی سطح بالا را به کدی خوانا برای ماشین تبدیل می‌کند. به عبارتی، برنامه‌ای است که کدهای قابل خواندن توسط انسان را به زبانی که پردازنده‌های کامپیوتر قادر به درک آن باشند (یعنی کدهای باینری یا همان صفر و یک) تبدیل می‌کند.

یک کامپایلر باید با سینتکس زبان‌ برنامه‌نویسی که کدهای برنامهٔ مذکور با آن نوشته می‌شوند آشنایی داشته باشد اما در عین حال باید در نظر داشت که کامپایلر نمی‌تواند ارورها و خطاهای موجود در برنامه را تصحیح کند و از همین روی اگر خطایی در کدتان وجود داشته باشد، باید تغییراتی را در سینتکس برنامه ایجاد کنید که در غیر این‌ صورت کدتان کامپایل نخواهد شد.

فرآیند کامپایل فرآیندی نسبتاً پیچیده است که طی آن زمان بسیار زیادی صرف تجزیه و تحلیل و ترجمهٔ سورس‌کد به کدی قابل درک برای کامپیوتر می‌شود. به طور کلی، کامپایلرها سورس‌کد را می‌خوانند و یک کد قابل‌اجرا در خروجی تحویل می‌دهند. به عبارت دیگر، سورس‌کد نرم‌افزارهایی را که با یک زبان سطح بالا نوشته شده‌اند به صفر و یک‌هایی تبدیل می‌کند که کامپیوتر قادر به درک آن‌ها باشد. در واقع، کدی را که یک برنامه‌نویس می‌نویسد را به فرمتی خوانا برای CPU تبدیل می‌کند.

**Interpreter چیست؟**  
[Interpreter](https://en.wikipedia.org/wiki/Interpreter_(computing)) (مُفسِر) یک برنامهٔ کامپیوتری است که هر خط از دستورات یک زبان سطح بالا را به کد ماشین تبدیل می‌کند. Compiler و Interpreter هر دو کاری یکسان، یعنی تبدیل کدهای زبان سطح بالا به کد ماشین، انجام می‌دهند اما کامپایلر کد را **پیش** از اجرای برنامه به کد ماشین تبدیل می‌کند (یعنی یک فایل اجرایی همچون exe می‌سازد) در حالی‌ که مفسر کد را **حین** اجرا به کد ماشین تبدیل می‌کند.

همان‌طور که پیش از این گفته شد، اینترپریتر سورس‌کد را خط به خط در حین اجرا ترجمه می‌کند به طوری که سورس یک برنامهٔ نوشته شده با زبانی سطح بالا را به طور کامل به زبان ماشین ترجمه می‌کند و این در حالی است که مفسر اجازه می‌دهد تا ارزیابی و اصلاح برنامه در حین اجرا (Run-time) صورت پذیرد.

نیاز به توضیح نیست که پروسهٔ اینترپریت زمان نسبتاً کمتری در مقایسه با کامپایل صرف تجزیه و تحلیل و پردازش برنامه می‌کند اما در عین حال سرعت اجرای برنامه در مقایسه با زبان‌هایی که کدشان کامپایل می‌شود، نسبتاً کندتر است.

منایع

<https://sokanacademy.com/blog/compiler-%D8%A8%D8%A7-interpreter-%DA%86%D9%87-%D8%AA%D9%81%D8%A7%D9%88%D8%AA%D9%87%D8%A7%DB%8C%DB%8C-%D8%AF%D8%A7%D8%B1%D8%A7-%D8%A7%D8%B3%D8%AA>