دانشکده علوم ریاضی و آمار

مدرس: دکتر مجتبی رفیعی

نیمسال اول ۱۴۰۰-۱۴۰۱

مبانی کامپیوتر و برنامه‌سازی

جلسه هفتم

نگارندگان: فاطمه رضائی و شینا آتش فراز

۹ آبان ۱۴۰۰

فهرست مطالب

۱ مقایسه کامپایلر و مفسر

در جلسات قبل، نرم‌افزار را تعریف کردیم و با نوعی از آن که نرم‌افزارهای برنامه‌نویسی هستند آشنا شدیم. زبان‌های برنامه‌نویسی یکی از مثال‌های مربوط به نرم‌افزارهای برنامه‌نویسی هستند که با دقیق‌تر شدن بر روی این موضوع برنامه‌های ترجمه را نیز شناختیم:

- اسمبلر (Assembler)،
- کامپایلر (Compiler)،
- مفسر (Interpreter).

جدول زیر تفاوت‌های اصلی کامپایلر و مفسر را به طور مختصر نشان می‌دهد.

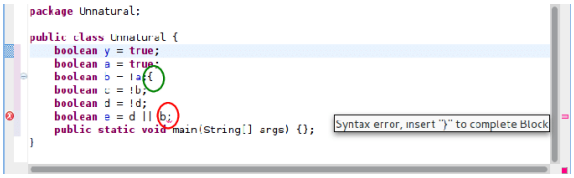
مفسر (Interpreter)	کامپایلر (Compiler)
اجرای خط به خط برنامه	اجرای کل برنامه
زمان تحلیل کم است ولی زمان اجرا در حالت کلی زیاد است.	زمان ترجمه و تحلیل برنامه زیاد است ولی زمان اجرا در حالت کلی کم است.
اشکال‌زدایی راحت‌تر است چرا که ترجمه تا زمانی که خطا وجود ندارد اتفاق افتاده است.	پیام خطا بعد از ترجمه کل متن اتفاق می‌افتد بنابراین خطایابی و پیدا کردن آن ممکن است نیاز به تلاش بیشتری داشته باشد.
حافظه به مراتب کمتری در مقایسه با کامپایلر نیاز دارد چرا که آبجکت کد تولید نمی‌شود.	کامپایلر نیازمند حافظه برای تولید آبجکت کد است.
مفسر برای اهداف امنیتی آسیب‌پذیرتر است.	برای اهداف امنیتی کامپایلر سودمندتر است و علت آن آبجکت کدها و سختی تغییر آن‌ها در مقایسه با سورس کد است.

جدول ۱: مقایسه کامپایلر و مفسر

۲ اشکال (Bug)

انواع خطاهایی که در یک برنامه ممکن است بروز کند به قرار زیر است:

- **خطاهای نحوی:** به خطاهایی اطلاق می‌شود که مربوط به ساختار برنامه و قوانین مربوط به آن است. تصویر زیر یک نمونه از این خطا را نشان می‌دهد.



زبان‌های برنامه‌نویسی گوناگون رویکردهای مختلفی در برابر چنین خطاهایی دارند:

۱. صرفا بیان می‌کند که خطا وجود دارد.

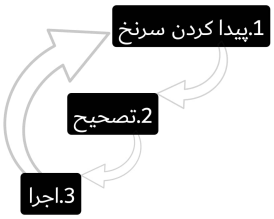
۲. علاوه بر خطا، خط مربوط بر آن را نیز نشان می‌دهد.

- **خطاهای زمان اجرا:** خطاهایی هستند که تا زمان اجرا مشاهده نمی‌شوند. به این نوع خطاها اصطلاحا استثنا (Exception) نیز گفته می‌شود، چرا که بیانگر یک اتفاق بد یا استثنا هستند. خطای تقسیم بر صفر، یک نمونه از این خطا می‌باشد.

- **خطاهای معنایی:** برنامه با موفقیت اجرا می‌شود و کامپیوتر هم خطایی تولید نمی‌کند اما خروجی موردنظر حاصل نمی‌شود. به عبارت دیگر برنامه‌ای که نوشته‌اید آن برنامه‌ای که انتظارش را دارید، نیست. شکل زیر بیانگر یک خطا از این نوع است. مثال: محاسبه یک عبارت محاسباتی با توجه به الویت عملگرها:
  - عبارت محاسباتی مد نظر: $(A + B) * C$،
  - عبارت نوشته شده در برنامه: $A + B * C$،
  - تعبیر برنامه از عبارت با توجه به الویت‌های تعریف شده در زبان برنامه‌نویسی برای عبارت‌های فاقد پرانتز: $A + (B * C)$.

۳ رفع اشکال (Debug)

اهمیت اشکال زدایی تا آن حد است که برخی از افراد برنامه‌نویسی و اشکال زدایی را یکسان می‌دانند. بنابراین یکی از مهارت‌های مهم در حوزه برنامه‌نویسی اشکال زدایی است که جذابیت خاص خودش را دارد: مثل یک فرایند کاراگاهی! شکل زیر گام‌های این فرایند را به تصویر کشیده است.



۴ پیونددهنده (Linker)

سورس کد ما ممکن است شامل ارجاع به کتابخانه‌هایی باشد یا نیاز به دسترسی به ماژول‌های مستقلی داشته‌باشد که به منظور خاصی طراحی و پیاده‌سازی شده‌اند. برای این منظور وظیفه پیونددهنده، پیوند زدن کتابخانه و ماژول‌های استفاده شده در برنامه با برنامه اصلی است.

برنامه کامپایل شده + کتابخانه ها و سایر ماژول های استفاده شده در برنامه = کد قابل اجرا

شکل زیر جایگاه پیونددهنده را در میان سایر مولفه‌های موثر در اجرای یک برنامه نشان می‌دهد.