

Traductor de Seudocódigo a Python: Facilitando la Inmersión en la Programación

Jhonnier Daniel Gutierrez Palacios

Johan Stiwar Bolívar Mosquera

Compiladores

Luis Delascar Valencia Mena

Fundación Universitaria Claretiana “Uniclaretiana”

Facultad de Ingeniería

Quibdó Chocó, Colombia

2023

**Introducción**

En la actualidad, la programación se ha convertido en una habilidad fundamental en numerosas disciplinas. Sin embargo, para muchas personas, aprender un lenguaje de programación puede ser intimidante o complicado. El seudocódigo se utiliza a menudo como un punto de partida más accesible para comprender los conceptos básicos de la programación. En este trabajo, se propone el desarrollo de un compilador básico que traduce un lenguaje de seudocódigo a código Python, con el objetivo de facilitar la transición de los principiantes hacia un lenguaje de programación más completo como Python.

**Objetivo General**

Desarrollar un compilador de seudocódigo a Python que facilite la comprensión de la programación y las estructuras de control en Python, especialmente para principiantes, como bien lo podrían ser estudiantes interesados en la materia.

**Objetivos específicos**

1. Desarrollar un compilador que traduzca un conjunto limitado de instrucciones de seudocódigo a código Python.

2. Facilitar la comprensión de la sintaxis.

3. Proporcionar una herramienta educativa para principiantes en programación.

**Justificación**

La programación es una habilidad esencial en la sociedad actual, y Python es un lenguaje de programación ampliamente utilizado y apreciado en la industria actual. Sin embargo, la curva de aprendizaje puede ser abrupta para los principiantes. El Colegio Carrasquilla imparte una formación especializada en la cual los estudiantes que hacen parte de la modalidad de Ingeniería de Sistemas deben adquirir habilidades fundamentales en programación y lógica. Sin embargo, a menudo, los conceptos de programación pueden resultar abrumadores para los estudiantes novatos. El desarrollo de un compilador que traduzca seudocódigo a Python puede ayudar a abordar esta brecha y permitir a los estudiantes y nuevos programadores comprender los conceptos básicos de la programación de manera más accesible.

**Fases del Proyecto**

1. Diseño del lenguaje de seudocódigo compatible con Python.

2. Desarrollo del compilador.

3. Pruebas y depuración.

4. Documentación del proyecto.

5. Implementación de mejoras y optimizaciones.

**Procesos**

- Diseñar la gramática y sintaxis del lenguaje de seudocódigo.

- Desarrollar el compilador en Python.

- Realizar pruebas exhaustivas del compilador utilizando ejemplos de seudocódigo.

- Corregir errores y optimizar el código.

- Documentar el proyecto, incluyendo un manual de usuario.

**Métodos**

Para la traducción de seudocódigo a Python, se utilizará un enfoque similar al código proporcionado en la pregunta, que maneja las instrucciones comunes de seudocódigo como "INICIO", "FIN", "SI", "ENTONCES", "SINO", "MIENTRAS", "HACER", "LEER" y "ESCRIBIR". Se analizarán las líneas de seudocódigo y se generarán las correspondientes líneas de código Python utilizando una serie de reglas y patrones.

**Despliegue del Compilador**

Para el desarrollo de este compilador se hace uso de Visual Studio CODE. El compilador estará disponible como una aplicación de línea de comandos. El usuario podrá proporcionar un archivo de seudocódigo como entrada y obtendrá un archivo de código Python como salida. Además, se proporcionará documentación detallada sobre cómo usar la herramienta.

**Metodología**

El proyecto de desarrollo de un compilador de seudocódigo a Python centrado en estudiantes de Ingeniería de Sistemas del Colegio Carrasquilla funcionar mejor bajo la metodología ágil por varias razones. Como la flexibilidad y adaptabilidad, lo cual permite que el proyecto se adapte a cambios y necesidades emergentes. En el contexto educativo, las necesidades de los estudiantes y educadores pueden evolucionar a medida que el proyecto avanza. La metodología ágil permite ajustar el enfoque y las prioridades de manera continua en función de la retroalimentación de los usuarios. Además de esto, en un entorno educativo, es esencial recopilar la retroalimentación de los usuarios, en este caso, estudiantes y educadores. La metodología ágil fomenta la retroalimentación continua a través de las reuniones de revisión y retrospectiva al final de cada sprint, lo que facilita la adaptación del proyecto según las necesidades reales.

**Fase 1: Diseño del Lenguaje de Seudocódigo**

* Identificar y definir las instrucciones y estructuras de control clave utilizadas en el seudocódigo.
* Diseñar la gramática y la sintaxis del lenguaje de seudocódigo, asegurándose de que sea intuitivo y compatible con Python.
* Crear una lista de palabras clave y reglas gramaticales del lenguaje de seudocódigo.

**Fase 2: Desarrollo del Compilador en Python**

* Implementar un analizador léxico y sintáctico en Python que pueda procesar el lenguaje de seudocódigo y generar código Python equivalente.
* Establecer reglas de traducción para cada estructura de control y palabra clave del seudocódigo.

**Fase 3: Pruebas y Depuración**

* Crear un conjunto de pruebas exhaustivas que incluyan ejemplos de seudocódigo representativos de los temas cubiertos en la modalidad de Ingeniería de Sistemas del Colegio Carrasquilla.
* Ejecutar las pruebas y corregir cualquier error o comportamiento inesperado del compilador.
* Asegurarse de que el código Python generado sea funcional y cumpla con los requerimientos de cada estructura de control.

**Fase 4: Documentación del Proyecto**

* Elaborar una documentación completa que incluya una descripción detallada de la gramática del seudocódigo, la sintaxis admitida y las reglas de traducción.
* Crear un manual de usuario que explique cómo usar el compilador, con ejemplos y guías paso a paso.
* Preparar documentación técnica para futuras referencias y posibles mejoras en el proyecto.

**Fase 5: Implementación de Mejoras y Optimizaciones**

* Recopilar comentarios de usuarios, especialmente de los estudiantes de Ingeniería de Sistemas del Colegio Carrasquilla, y utilizarlos para identificar áreas de mejora en el compilador.
* Realizar actualizaciones y optimizaciones según sea necesario para mejorar la usabilidad y la eficiencia del compilador.
* Mantener una comunicación abierta con los usuarios para garantizar que sus necesidades y expectativas se cumplan.

**Antecedentes**

Aunque el desarrollo de un compilador de seudocódigo a Python puede ser un proyecto específico, hay antecedentes relacionados con la creación de compiladores, lenguajes intermedios y herramientas educativas que sirven como inspiración.

**Jupyter Notebooks:** Jupyter Notebook es una plataforma ampliamente utilizada para la enseñanza de programación. Si bien no es un compilador en sí, permite la creación de documentos interactivos que combinan texto explicativo con fragmentos de código Python ejecutable. Sitio web oficial: [Jupyter](https://jupyter.org/).

**Scratch:** Scratch es un lenguaje de programación visual diseñado para enseñar a los niños y principiantes a programar. Aunque no se traduce directamente a Python, su enfoque en la lógica y las estructuras de control puede inspirar la creación de un lenguaje de pseudocódigo. Sitio web oficial: [Scratch](https://scratch.mit.edu/).

**Codeconverter:** Es una página web que alberga una gran cantidad de lenguajes de programación en su haber, de forma que permite pegar un código que ya tengamos realizado y por medio de la IA lo convierta a otro lenguaje seleccionado, aunque no tiene ninguna clase de seudocódigo, puede servir para hacerle saber a las personas primerizas las diferencias entre los lenguajes. Sitio web oficial: [Codeconverter](https://codeconverter.com/).

**Entorno de desarrollo para la ejecución y traducción de pseudocódigo:** Es un proyecto de fin de carrera realizado por Jara Loayza y Juan Carlos que correspondo a la construcción de un entorno de desarrollo que permita la ejecución de pseudocódigo como herramienta de apoyo a las etapas de diseño, ejecución y validación de un algoritmo. Link de referencia: Jara, L. Juan, C. (2014, 23 junio). *Entorno de desarrollo para la ejecución y traducción de pseudocódigo*. <https://tesis.pucp.edu.pe/repositorio/handle/20.500.12404/5401>

**MultiPseudo, lenguaje de programación en pseudocódigo multi-idioma:** Este es un informe realizado por Antonio López García y Jaime Urquiza Fuentes. MultiPseudo es un intérprete de pseudocódigo presentado como software educacional libre, pensado para su uso en la enseñanza de la programación. Permite la traducción de algoritmos entre los diferentes lenguajes de pseudocódigo, así como la traducción del código a Python 2. MultiPseudo también permite la programación y desarrollo de algoritmos en Python 2, lo que contribuye a mejorar su idoneidad como herramienta de aprendizaje. Además, la herramienta contempla la posibilidad de evaluar y añadir nuevos lenguajes de pseudocódigo definidos por el usuario. Link de referencia: Antonio, L. G. (2023). Jaime, U. F. *MultiPseudo, lenguaje de programación en pseudocódigo multi-idioma*. <https://rua.ua.es/dspace/handle/10045/137226>

**Herramientas empleadas**

Para la realización de este proyecto se utilizó Visual Studios Code ya que admite una amplia variedad de extensiones que facilitan el desarrollo en Python. Se le puede instalar la extensión "Python" para obtener funciones avanzadas de edición, depuración, administración de entornos virtuales y resaltado de sintaxis específicas de Python. Además de esto, incluye una excelente integración con Git, lo que facilita el seguimiento de cambios y el trabajo colaborativo en tu proyecto. Finalmente, gracias a su estandarización es conocida u usada por la gran mayoría de programadores, lo cual permite que mediante tutoriales, documentación y extensiones creadas por la comunidad esta plataforma de programación tenga una alta gama de posibilidades.

**Resultados**

A lo largo de este proyecto, se han logrado los siguientes resultados:

**Diseño del Lenguaje de Seudocódigo:** Se ha diseñado un lenguaje de seudocódigo que es compatible con Python y permite la traducción directa y sin ambigüedades. Este lenguaje incluye las principales estructuras de control, como condicionales, bucles y operaciones de entrada/salida.

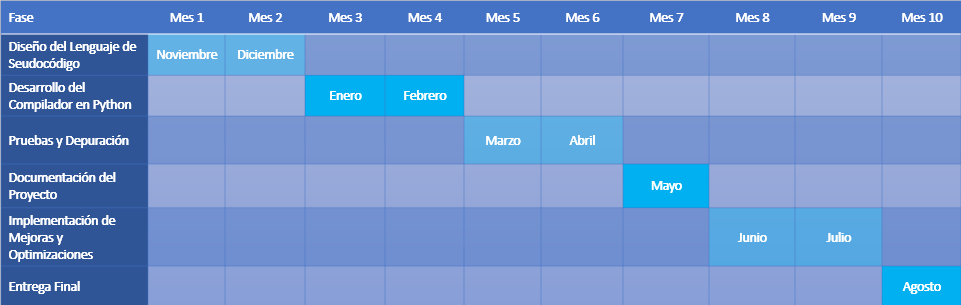
**Desarrollo del Compilador en Python:** Se ha implementado un compilador en Python capaz de analizar el lenguaje de seudocódigo y generar código Python equivalente. El compilador cumple con los estándares de calidad y se ha diseñado para ser fácil de mantener y extender en el futuro.

**Pruebas y Depuración:** Se han llevado a cabo pruebas exhaustivas del compilador utilizando una variedad de ejemplos de seudocódigo, incluidos casos que abarcan todas las instrucciones admitidas. Las pruebas han permitido identificar y corregir errores, asegurando que el compilador funcione de manera fiable y produzca código Python válido.

**Documentación del Proyecto:** Se ha creado documentación completa que incluye una descripción detallada de la gramática del seudocódigo, las reglas de traducción, y las instrucciones admitidas. Además, se ha preparado un manual de usuario que explica cómo usar el compilador, con ejemplos y guías paso a paso.

**Implementación de Mejoras y Optimizaciones:** Basándonos en los comentarios de los usuarios, se han realizado mejoras y optimizaciones en el compilador. Estas mejoras han permitido aumentar la usabilidad y eficiencia de la herramienta, garantizando una experiencia de aprendizaje más efectiva.

**Cronograma de actividades**



**Conclusión**

El desarrollo de un compilador de seudocódigo a Python centrado en estudiantes de Ingeniería de Sistemas del Colegio Carrasquilla ha resultado en una herramienta educativa efectiva y valiosa. A través de este proyecto, se ha logrado diseñar un lenguaje de seudocódigo claro y compatible con Python, desarrollar un compilador funcional en Python, y llevar a cabo pruebas exhaustivas para garantizar su precisión y confiabilidad. La documentación detallada y el manual de usuario proporcionan recursos esenciales para los estudiantes y educadores. Además, la retroalimentación de los usuarios y las iteraciones han permitido mejorar y optimizar el compilador para una experiencia de aprendizaje más enriquecedora.

Este proyecto tiene el potencial de facilitar la enseñanza y el aprendizaje de la programación en Python, especialmente para los estudiantes de Ingeniería de Sistemas del Colegio Carrasquilla. Proporciona una transición gradual y accesible desde el seudocódigo hacia Python, lo que fomenta la comprensión de conceptos clave de programación. La flexibilidad y versatilidad de Visual Studio Code (VSCode) como entorno de desarrollo han facilitado el desarrollo y la implementación exitosa de este compilador.

Bibliografía

"Python Documentation" - Documentación oficial de Python. [Python.org](https://docs.python.org/3/)

"Jupyter Documentation" - Documentación oficial de Jupyter para la ejecución de Jupyter Notebooks. [Jupyter.org](https://jupyter.readthedocs.io/en/latest/index.html)

"Visual Studio Code Documentation" - Documentación oficial de Visual Studio Code. [Visual Studio Code Docs](https://code.visualstudio.com/docs)

"Scratch" - Sitio web oficial de Scratch para explorar el concepto de lenguajes de programación visuales. [Scratch](https://scratch.mit.edu/)

“Codeconverter” – Página oficial de Codeconverter. [Codeconverter](https://codeconverter.com/)

“Entorno de desarrollo para la ejecución y traducción de pseudocódigo” – Documentación oficial. Jara, L. Juan, C. (2014, 23 junio). *Entorno de desarrollo para la ejecución y traducción de pseudocódigo*. <https://tesis.pucp.edu.pe/repositorio/handle/20.500.12404/5401>

“MultiPseudo, lenguaje de programación en pseudocódigo multi-idioma” - Documentación oficial. Antonio, L. G. (2023). Jaime, U. F. *MultiPseudo, lenguaje de programación en pseudocódigo multi-idioma*. <https://rua.ua.es/dspace/handle/10045/137226>