Aplicaciones del lenguaje Python

|  |  |  |
| --- | --- | --- |
| Monserrat Castillo Quesada Universidad Nacional de Costa Rica, Heredia, Costa Rica  montserrat.castillo.quesada@est.una.ac.cr | Adrián Morales Pincay Universidad Nacional de Costa Rica,  Heredia, Costa Rica adrian.morales.pincay@est.una.ac.cr | Luis Antonio Aguilar Monge Universidad Nacional de Costa Rica Heredia, Costa Rica  [luis.aguilar.monge@est.una.ac.cr](mailto:luis.aguilar.monge@est.una.ac.cr) |

|  |  |
| --- | --- |
| Ángel Quesada Jiménez Universidad Nacional de Costa Rica,  Heredia, Costa Rica angel.quesada.jimenez@est.una.ac.cr |  |

1. Marco Teórico

[1]. La aplicación del formato de datos base estándar de Weather Radar resuelve el fenómeno de que una variedad de formatos de archivos de datos básicos coexista en CINRAD Weather Radar durante mucho tiempo. En terminales de aplicaciones empresariales, solo se necesita el método de formato de datos base estándar para analizar todos los datos base de los datos base del radar meteorológico CINRAD. Este artículo presenta un análisis de los datos base estándar del radar meteorológico e introduce cómo utilizar el lenguaje de programación Python para realizar lecturas de datos y otras operaciones gráficas en el formato de datos base estándar del radar meteorológico.

[2]. El lenguaje de programación Python proporciona un entorno de desarrollo adecuado tanto para tareas computacionales como de visualización. Una de las ventajas clave de Python es que permite a los desarrolladores utilizar paquetes que amplían el lenguaje para proporcionar capacidades avanzadas, como manipulación de matrices y matrices, procesamiento de imágenes, procesamiento de señales digitales y visualización. Se han creado varias herramientas populares de exploración y visualización de datos en Python.

[3]. En el análisis de circuitos a menudo aparecen cálculos y dibujos manuales complicados, lo que dificulta el análisis de circuitos. Los métodos manuales de análisis del circuito requieren mucho tiempo, son laboriosos y propensos a errores. Python puede resolver mejor el problema en el análisis de circuitos que de forma manual. Python es un lenguaje de programación interpretado de alto nivel y de propósito general que se usa ampliamente en informática e ingeniería científicas.

[4]. Traducir pseudocódigo a un lenguaje de programación puede resultar difícil y llevar mucho tiempo, especialmente para personas que no están familiarizadas con los lenguajes involucrados. Sin embargo, los avances recientes en el procesamiento del lenguaje natural (NLP) han hecho posible crear un modelo que puede convertir automáticamente pseudocódigo escrito en lenguaje natural en código en un lenguaje de programación específico. Este estudio de investigación propone una solución a este problema mediante el uso de un modelo transformador de PNL para traducir pseudocódigo a código Python automáticamente.

[5]. En los últimos años, la amplia aplicación del lenguaje Python ha hecho que su trabajo de análisis sea cada vez más valioso. Muchos algoritmos de análisis estático deben depender de la construcción de gráficos de llamadas. En este artículo, realizamos un análisis empírico comparativo de varias herramientas de gráficos de llamadas estáticas de Python ampliamente utilizadas, tanto cuantitativa como cualitativamente.

[6]. Como lenguaje de programación dinámico, Python se utiliza ampliamente en muchos campos. Para los desarrolladores, varias características del lenguaje afectan la experiencia de programación. Para los investigadores, afectan la dificultad de desarrollar tareas como la búsqueda de errores y la optimización de la compilación. Investigaciones anteriores han demostrado que los programas con características dinámicas de Python son más propensos a cambios. Sin embargo, sabemos poco sobre el uso y el impacto de las características del lenguaje Python en proyectos Python del mundo real.

[7]. En este artículo se intenta esbozar la contribución del aprendizaje experiencial, la gamificación y las tecnologías emergentes a la enseñanza del lenguaje de programación Python. Se diseñaron cuatro escenarios con el fin de constituir una nueva propuesta didáctica para la consolidación de módulos específicos, incorporando elementos de juegos dinámicos, tecnología emergente y abordando el tema que se trata, de manera experiencial en un marco de aprendizaje auténtico mediante la resolución de problemas de la vida cotidiana.

[8]. Los métodos de enseñanza y los enfoques de aprendizaje tradicionales ya no son apropiados para muchos niños. Según los estudios, estos enfoques perjudican la motivación y el interés de los estudiantes por estudiar programación. Según la investigación, los juegos pueden ayudar a aprender a programar en Python. El objetivo de este artículo es presentar nuestro juego educativo que desarrollamos en el motor de juego Unity 3D. El juego se centra en los conceptos básicos puros de programación en lenguaje Python.

[9]. Este artículo propone un método que se basa en la semántica compartida para leer, escribir o manipular diseños a nivel de transistor. El objetivo final del marco es: leer un diseño de entrada escrito en una sintaxis específica y luego permitir escribir el mismo diseño en otra sintaxis. Primero, la descripción de entrada es analizada por una interfaz específica del lenguaje que la convierte en un árbol de sintaxis abstracta en memoria que sigue la semántica común. Esta actividad requiere agregar, quitar o reemplazar nodos, componentes o incluso subcircuitos completos. Por lo tanto, el marco está completamente escrito en C++ y sus API también están conectadas con Python. Todo el marco es de código abierto y está disponible en GitHub.

[10]. Escribir código utilizando lenguaje natural es una aplicación muy interesante de la traducción automática neuronal. Para lograr una pequeña parte de dicha aplicación, en este artículo intentamos generar fragmentos de código fuente de Python a partir de descripciones en inglés natural utilizando el conjunto de datos de Django. Entrenamos la arquitectura del transformador basada en la autoatención en los fragmentos del conjunto de datos

II. REFRENCIAS

[1] Y. Wen, Z. Jing, and W. Huan, “Application of Python in Weather Radar Standard Base Data,” in 2019 International Conference on Meteorology Observations (ICMO), Chengdu, China: IEEE, Dec. 2019, pp. 1–3. doi: 10.1109/ICMO49322.2019.9076594.

[2] E. W. Anderson, G. A. Preston, and C. T. Silva, “Using Python for Signal Processing and Visualization,” Comput. Sci. Eng., vol. 12, no. 4, pp. 90–95, 2010, doi: 10.1109/MCSE.2010.91.

[3] H. Zhang, P. Yang, and Y. Niu, “Application of Python Scientific computing library and Simulation in Circuit Analysis,” in 2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT), Bhopal, India: IEEE, Apr. 2023, pp. 892–898. doi: 10.1109/CSNT57126.2023.10134600.

[4] S. P. Tiwari, S. Prasad, and M. G. Thushara, “Machine Learning for Translating Pseudocode to Python: A Comprehensive Review,” in 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India: IEEE, May 2023, pp. 274–280. doi: 10.1109/ICICCS56967.2023.10142254.

[5] L. Yu, “Empirical Study of Python Call Graph,” in 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA: IEEE, 2019, pp. 1274–1276. doi: 10.1109/ASE.2019.00160.

[6] Y. Peng, Y. Zhang, and M. Hu, “An Empirical Study for Common Language Features Used in Python Projects,” in 2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA: IEEE, 2021, pp. 24–35. doi: 10.1109/SANER50967.2021.00012.

[7] C. Velaora and A. Kakarountas, “Logic Design as an Enabler to Python Programming Language Teaching,” in 2019 Panhellenic Conference on Electronics & Telecommunications (PACET), Volos, Greece: IEEE, 2019, pp. 1–6. doi: 10.1109/PACET48583.2019.8956286.

[8] M. Hlavaty, A. Kozakova, and O. Haffner, “Application for Python Programming Language Education Developed by Unity Engine,” in 2022 Cybernetics & Informatics (K&I), Visegrád, Hungary: IEEE, Sep. 2022, pp. 1–6. doi: 10.1109/KI55792.2022.9925955.

[9] N. Dall’Ora, S. Azam, E. Fraccaroli, A. Alberts, and F. Fummi, “A Common Manipulation Framework for Transistor-Level Languages,” in 2021 Forum on specification & Design Languages (FDL), Antibes, France: IEEE, Sep. 2021, pp. 01–07. doi: 10.1109/FDL53530.2021.9568379.

[10] M. Shah, R. Shenoy, and R. Shankarmani, “Natural Language to Python Source Code using Transformers,” in 2021 International Conference on Intelligent Technologies (CONIT), Hubli, India: IEEE, Jun. 2021, pp. 1–4. doi: 10.1109/CONIT51480.2021.9498268.