

ESCUELA DE POSGRADO NEWMAN

MAESTRÍA EN GESTIÓN DE TECNOLOGÍAS DE INFORMACIÓN



"Propuesta de mejora del proceso de aprendizaje de la programación, basada en la herramienta Github Copilot en los estudiantes del primer año de ingeniería de sistemas de la Universidad Católica de Santa María de Arequipa"

**Trabajo de investigación
para optar el Grado a Nombre de la Nación de:**

Maestro en
Gestión de Tecnologías de Información

Autores:

Bachiller Lazo Barreda, Dely Marysheck
Bachiller Flores Gutiérrez, Oscar Javier

Director:

Maestro Valderrama Herrera, Roberto Marcel

TACNA-PERÚ

2024

15%

SIMILARITY INDEX

14%

INTERNET SOURCES

3%

PUBLICATIONS

10%

STUDENT PAPERS

“El texto final, datos, expresiones, opiniones y apreciaciones contenidas en este trabajo son de exclusiva responsabilidad de los autores”.

ÍNDICE GENERAL

RESUMEN.....	11
ABSTRACT.....	12
INTRODUCCIÓN.....	13
I. ANTECEDENTES DEL ESTUDIO.....	15
1.1. Título del Tema.....	15
1.2. Planteamiento del Problema.....	15
1.3. Objetivos de la Investigación.....	16
1.3.1. Objetivo Principal.....	16
1.3.2. Objetivos Secundarios.....	16
1.4. Metodología.....	17
1.4.1. Estrategias de recolección de datos.....	18
1.5. Justificación.....	19
1.6. Definiciones.....	21
1.7. Alcances y limitaciones.....	25
1.7.1. Alcances:.....	25
1.7.2. Limitaciones:.....	26
1.8. Cronograma.....	27
2. MARCO TEÓRICO.....	28
2.1. Conceptualización de tópicos clave.....	28
2.1.1. Inteligencia artificial.....	28
2.1.2. Contexto Histórico de la Inteligencia artificial.....	28
2.1.3. Lenguaje de programación (L.P.).....	30

2.1.4.	Python.....	30
2.1.5.	IDE Visual Studio Code	30
2.1.6.	Programación	31
2.1.7.	Algoritmos.....	31
2.1.8.	Complejidad Algorítmica.....	31
2.1.9.	Ética en la I.A. para educación.....	33
2.1.10.	Rendimiento Académico.....	34
2.1.11.	Métodos de evaluación del rendimiento	35
2.1.12.	Factores de evaluación del rendimiento.....	35
2.1.13.	Recursos educativos que aplican I.A.	36
2.1.14.	Knewton.....	38
2.2.	Importancia de la Inteligencia Artificial en la educación	38
2.2.1.	Análisis comparativo.....	39
2.3.	Análisis crítico	40
2.3.1.	Desafíos de la I.A. en la educación.....	40
3.	MARCO REFERENCIAL. TRABAJO DE INVESTIGACIÓN	41
3.1.	Reseña Histórica	41
3.2.	Filosofía organizacional.....	43
3.2.1.	Misión de la UCSM.....	43
3.2.2.	Visión de la UCSM.....	43
3.2.3.	Valores de la UCSM	43
3.2.4.	Propósito de la UCSM	44
3.2.5.	Misión de la Facultad de Ciencias e Ingenierías Físicas y Formales	44
3.2.6.	Visión de la Facultad de Ciencias e Ingenierías Físicas y Formales	44

3.2.7.	Misión de la Escuela Profesional de Ingeniería de Sistemas	45
3.2.8.	Visión de la Escuela Profesional de Ingeniería de Sistemas	45
3.3.	Diseño organizacional	45
3.3.1.	Datos generales.....	45
3.3.2.	Organigrama estructural.....	46
3.4.	Productos y/o servicios	47
3.4.1.	Escuelas Profesionales de pregrado en modalidad presencial por áreas	47
3.4.2.	Escuelas Profesionales de pregrado en modalidad a distancia.....	48
3.4.3.	Postgrado	49
3.4.4.	Servicios	50
3.4.4.1.	Servicios artísticos culturales	50
3.4.4.2.	Servicios deportivos	53
3.5.	Diagnóstico organizacional	54
3.5.1.	Contexto Institucional	55
3.5.2.	Análisis del proceso de Enseñanza actual	55
3.5.3.	Revisión de la Herramienta GitHub Copilot:.....	56
3.5.4.	Competencias y capacidades para alcanzar por los estudiantes del curso de Metodología de la Programación.....	57
4.	PROPUESTA DE MEJORA	58
4.1.	Identificación del área a mejorar	58
4.2.	Diagnóstico	58
4.2.1.	Tipo de investigación	58
4.2.2.	Características de la población	59
4.2.3.	Delimitación de la población	59

4.2.4.	Tipo de muestra.....	59
4.2.5.	Tamaño de la muestra.....	60
4.2.6.	Proceso de selección.....	60
4.2.7.	Los métodos y las técnicas	60
4.2.7.1.	Métodos teóricos	60
4.2.7.2.	Técnicas e instrumentos.....	61
4.2.8.	Análisis de la práctica de evaluación.....	62
4.2.9.	Análisis y tabulación de la encuesta.....	63
4.2.10.	Análisis de entrevistas a docentes del curso de metodología de la programación	74
4.3.	Diseño de mejora	80
4.4.	Definición de los mecanismos de control	87
5.	SUGERENCI.A. S.....	88
5.1.	Conclusiones.....	88
5.2.	Recomendaciones.....	89
6.	BIBLIOGRAFÍA	91
7.	ANEXOS	94

ÍNDICE TABLAS

Tabla 1: FODA Escuela de Ingeniería de Sistemas.....	54
Tabla 2: Tipos de preguntas práctica calificada	63

ÍNDICE IMÁGENES

Imagen 1: Cronograma	27
Imagen 2: Ubicación Geográfica de la Universidad Católica de Santa María	46
Imagen 3: Organigrama estructural de la UCSM	47
Imagen 4: <i>Admisión para pregrado modalidad presencial UCSM</i>	48
Imagen 5: <i>Admisión para pregrado modalidad virtual UCSM</i>	49
Imagen 6: <i>Admisión para postgrado EPG UCSM</i>	49
Imagen 7: <i>Tuna Universitaria de la UCSM</i>	50
Imagen 8: <i>Elenco de teatro de la UCSM</i>	51
Imagen 9: <i>Orquesta Sinfónica de la UCSM</i>	51
Imagen 10: <i>Rondalla santamariana de la UCSM</i>	52
Imagen 11: <i>Ballet Folclórico de la UCSM</i>	53
Imagen 12: <i>Equipo de Voley de la UCSM</i>	53
Imagen 13: ¿Utilizó antes GitHub Copilot?	64
Imagen 14: ¿Con qué frecuencia utiliza GitHub Copilot en el desarrollo de sus trabajos?	64
Imagen 15: ¿Cuántos ejercicios resolvió completamente con GitHub Copilot?	65
Imagen 16: ¿Cuántos ejercicios resolvió completamente? – Método tradicional	66
Imagen 17: ¿Cuál de los ejercicios le pareció más dificultoso con GitHub Copilot?	67
Imagen 18: ¿Cuál de los ejercicios le pareció más dificultoso? - Método tradicional	67
Imagen 19: ¿Cuál de los ejercicios le pareció más sencillo?	68
Imagen 20: ¿Cuál de los ejercicios le pareció más sencillo? - Método tradicional	69
Imagen 21: ¿Cuánto tiempo aproximadamente en minutos le tomó resolver cada uno de los ejercicios?	69
Imagen 22: ¿Cuánto tiempo aproximadamente en minutos le tomó resolver cada uno de los ejercicios? - Método tradicional	70
Imagen 23: ¿Cuáles fueron las principales dificultades que tuvo?.....	71
Imagen 24: ¿Cuáles fueron las principales dificultades que tuvo? - Método tradicional	71

Imagen 25: ¿Cuántos intentos erróneos tuvo antes de generar la solución final con GitHub Copilot?.....	72
Imagen 26: ¿Cuántos intentos erróneos tuvo antes de generar la solución final con GitHub Copilot? - Método tradicional.....	73
Imagen 27: ¿Considera que el uso de herramientas de Inteligencia Artificial es beneficioso en el aprendizaje de sus estudiantes?	75
Imagen 28: Usted como docente, ¿Ha utilizado la herramienta GitHub Copilot para elaborar su material de clase?	76
Imagen 29: ¿Cuál cree que sería la principal dificultad al utilizar las herramientas de Inteligencia Artificial en la educación?.....	77
Imagen 30: En una escala del 1 al 5, donde 1 es nada útil y 5 es totalmente útil, cómo respondería usted la siguiente pregunta, ¿Qué tan útil es la herramienta GitHub Copilot para enseñar su materia?	78
Imagen 31: En una escala del 1 al 5, donde 1 es no contribuye y 5 contribuye positivamente. ¿En qué medida considera que la herramienta GitHub Copilot contribuye al desarrollo de habilidades de programación de los estudiantes?	79
Imagen 32: Considera usted que ¿La herramienta es fácil de usar para los estudiantes con diferentes niveles de habilidad en programación?	80
Imagen 33: Ejercicio propuesto 1	81
Imagen 34: Ejercicio propuesto 2	82
Imagen 35: Ejercicio propuesto 3	82
Imagen 36: Ejercicio propuesto 4	83
Imagen 37: Ejercicio propuesto 5	83
Imagen 38: <i>Ejercicio propuesto 6</i>	84
Imagen 39: <i>Ejercicio propuesto 7</i>	84
Imagen 40: <i>Ejercicio propuesto 8</i>	85
Imagen 41: <i>Ejercicio propuesto 9</i>	86
Imagen 42: <i>Ejercicio propuesto 10</i>	86

ÍNDICE ANEXOS

Anexo 1: Cuestionario complejidad 1	94
Anexo 2: Cuestionario complejidad 2	96
Anexo 3: Cuestionario complejidad 3	98
Anexo 4: Cuestionario aplicado a los estudiantes (una pregunta aleatoria por cada nivel de complejidad).....	100
Anexo 5: Opinión de los catedráticos.....	102

RESUMEN

El permanente desarrollo de la programación requiere una adaptación continua, principalmente de los métodos de enseñanza, para asegurar una adecuada preparación de los estudiantes de Ingeniería de Sistemas.

En este contexto, se estudió a GitHub Copilot como una herramienta basada en inteligencia artificial, que proporciona sugerencias de código en tiempo real a los usuarios, en este caso estudiantes, para crear código fuente de forma rápida y eficiente, lo cual conllevó a la mejora de la productividad y posiblemente el rendimiento de los estudiantes en tareas de programación y desarrollo de software.

La UCSM tiene la Escuela Profesional de Ingeniería de Sistemas, en cuya malla curricular se contiene el curso de Metodología de la Programación en los estudiantes de Primer Año, con la finalidad de brindar a los estudiantes conocimientos básicos, habilidades y destrezas en la lógica de programación, para gestionar y producir información que permita a las personas e instituciones tomar decisiones, principalmente en los procesos diarios que tienen que enfrentar, permitiendo ahorrar tiempo, esfuerzo y dinero.

En resumen, el propósito de esta investigación se centró en una propuesta de mejora del proceso de enseñanza de la programación basada en la herramienta GitHub Copilot en los estudiantes del Primer Año de Ingeniería de Sistemas de la Universidad Católica de Santa María de Arequipa, UCSM.

Palabras Claves: GitHub, Copilot, enseñanza, programación.

ABSTRACT

Permanent change in development of software requires continuous adaptation of teaching methods to ensure preparation of Systems Engineering students.

GitHub Copilot is a tool based on artificial intelligence, provides real-time code suggestions to users, students of software development, to help them create source code, quickly and efficiently, which leads to improving productivity and possibly the performance of the devices to students of software development tasks.

Furthermore, UCSM states that it hosts the Professional School of Systems Engineering, within which the curriculum includes the course on Programming Methodology for first-year students. The aim is to provide students with fundamental knowledge, skills, and abilities in programming logic to manage and generate information, enabling individuals and institutions to make informed decisions in their daily processes. This facilitates saving time, effort, and money.

In summary, the purpose of this research focuses on a proposal to improve the programming teaching process based on the GitHub Copilot tool for students of the First Year of Systems Engineering at the Universidad Católica de Santa María de Arequipa.

Keywords: GitHub, Copilot, teaching, programming.

INTRODUCCIÓN

Durante el desarrollo de la carrera de Ingeniería de Sistemas en la Universidad Católica de Santa María, UCSM, de Arequipa, se inicia con cursos pilares, esenciales en la formación en programación, uno de ellos es el curso Metodología de la Programación, donde se exploran técnicas y herramientas, que facilitan el aprendizaje de temas enfocados al desarrollo del software. En esta ocasión, el trabajo de investigación desarrollado consiste en una propuesta de mejora para el dictado del curso, haciendo uso de una herramienta que se lanzó recientemente, la cual se denomina GitHub Copilot.

GitHub Copilot sirve para generar código fuente y está basado en técnicas de Machine Learning, también conocido como *pair programming* en I.A. (Inteligencia Artificial). Para entenderlo mejor, este software se comporta como un clásico *Intellisense* (auto completa el código) pero es más eficiente, e irá mejorando a medida que lancen sus actualizaciones. Además, la herramienta hace pequeñas sugerencias de código, es decir, predice lo que se quiere completar y ofrece funciones complementarias con múltiples variantes.

En este contexto se destaca la enorme importancia de adaptarse a las innovaciones tecnológicas en el sector educación, específicamente en la Ingeniería de Sistemas. La investigación buscó mejorar el proceso de enseñanza de la programación basado en la herramienta GitHub Copilot mostrando el impacto en la eficiencia de la escritura de código, en la percepción estudiantil, en la comprensión abstracta de conceptos y la resolución de problemas algorítmicos de los estudiantes de Primer Año de Ingeniería de Sistemas de la Universidad Católica de Santa María.

Para el desarrollo del presente trabajo se ha considerado en el primer capítulo los antecedentes de estudio más importantes, en el segundo capítulo se explicó la fundamentación teórica con los diversos conceptos que se requieren para la presente investigación, finalmente en el capítulo tres se describe la organización de la investigación aplicada.

Este estudio pretende proporcionar información valiosa, para adaptar las estrategias pedagógicas a las herramientas tecnológicas emergentes y así contribuir al rendimiento académico de los estudiantes. La metodología que se utilizó consta de métodos cuantitativos, incluye encuestas, entrevistas y análisis de desempeño, diseñados para ayudar a comprender cómo esta innovadora herramienta impacta en la formación académica y profesional de los ingenieros de sistemas en su primer año de estudio.

I. ANTECEDENTES DEL ESTUDIO

1.1. Título del Tema

Propuesta de mejora del proceso de enseñanza de la programación, basada en la herramienta *Github Copilot* en los estudiantes del Primer Año de Ingeniería de Sistemas de la Universidad Católica de Santa María de Arequipa.

1.2. Planteamiento del Problema

Actualmente, la tecnología está experimentando cambios constantes y vertiginosos debido a que se encuentra en un permanente proceso de transformación, generando un cambio en la modalidad de desarrollar el software; lo cual, evidentemente, también tiene un impacto en la educación superior, al momento de desarrollar cursos de programación. La formación en programación es un componente principal en la malla curricular de Ingeniería de Sistemas, siendo esencial para preparar a los estudiantes para los desafíos tecnológicos emergentes. Sin embargo, el desarrollo continuo de las tecnologías de información demanda una adaptación cuasi permanente de los métodos de enseñanza para garantizar la relevancia y eficacia del proceso educativo.

En este contexto, surge la necesidad de analizar exhaustivamente el método educativo de la enseñanza de la programación a nivel de Primer Año de Ingeniería de Sistemas de la UCSM, Arequipa. Entre estos desafíos, se destaca la dificultad que los docentes encuentran en el dictado de la lógica de la programación a los estudiantes que inician la carrera y se debe abordar la escritura de código de manera efectiva y eficiente. Por lo cual los docentes están en constante búsqueda de herramientas que

se puedan aplicar para optimizar el aprendizaje, y así, romper las barreras de la comprensión y aplicación práctica de los conceptos de programación, disminuyendo los obstáculos en el desarrollo de habilidades esenciales en esta disciplina.

Adicionalmente, la brecha entre los métodos tradicionales de enseñanza y las demandas del entorno laboral actual plantea interrogantes, sobre la capacidad de los graduados, para integrarse exitosamente en el ámbito de la Ingeniería de Sistemas. En este caso, la herramienta GitHub Copilot representa una innovación, potencialmente transformadora, en el desarrollo de la enseñanza de la programación en los estudiantes de Primer Año de la UCSM.

Por lo tanto, el aporte significativo de esta investigación radica en realizar un planteamiento para mejorar el desarrollo de la enseñanza de la programación, basada en la herramienta GitHub Copilot en los estudiantes de Primer Año de Ingeniería de Sistemas de la UCSM.

1.3. Objetivos de la Investigación

1.3.1. Objetivo Principal

Realizar una propuesta de mejora del proceso de aprendizaje de la programación basada en la herramienta *GitHub Copilot* en los estudiantes de Primer Año de Ingeniería de Sistemas de la UCSM de Arequipa en el año 2023.

1.3.2. Objetivos Secundarios

Precisar el grado de familiaridad en el uso de *GitHub Copilot* entre los educandos de Primer Año de Ingeniería de Sistemas de la UCSM en el año 2023.

Determinar la existencia de una relación directa entre el uso frecuente de *GitHub Copilot* y el rendimiento académico de los alumnos de programación.

Identificar los beneficios e inconvenientes para un buen desempeño académico con el uso de *GitHub Copilot* en esta muestra de estudiantes universitarios.

Cuantificar el porcentaje de aprobados y desaprobados en el rendimiento académico por el uso de *GitHub Copilot* entre los estudiantes de Primer Año de Ingeniería de Sistemas de la Universidad Católica de Santa María en el año 2023.

1.4. Metodología

Se ha implementado una metodología que consta de las siguientes fases:

Fase 1: Análisis de las referencias bibliográficas

Se realizó una revisión minuciosa de la literatura relacionada con el uso de *GitHub Copilot* en la educación y del rendimiento académico, así como un análisis del impacto entre estas.

Fase 2: Formulación y validación de documentos a aplicar en investigación

Se diseñó un conjunto de encuestas, las cuales contemplaban preguntas pertinentes para los estudiantes de programación en la UCSM.

Las encuestas permitieron analizar y evaluar la propuesta de mejora del aprendizaje con el uso de la herramienta *GitHub Copilot* en los estudiantes de Primer Año de Ingeniería de Sistemas.

Fase 3: Aplicación del cuestionario de preguntas de la entrevista a los docentes del curso Metodología de la Programación

Como instrumento de estudio se ha considerado la entrevista, debido a que la información proviene de los catedráticos, quienes dictan el curso de Metodología de la Programación y ésta se realiza para conocer el impacto en el rendimiento académico de los estudiantes.

Fase 4: Análisis de resultados y desarrollo de conclusiones

Los resultados se presentaron a través de la tabulación, explicación e interpretación de las respuestas de las encuestas usando Microsoft Office 365 y el software para estadística. Finalmente, se formularon conclusiones y recomendaciones tras interpretar los resultados alcanzados de los procesos previos.

1.4.1. Estrategias de recolección de datos

a. Población y muestra

Se ha planteado que la población objetivo sean todos los estudiantes del curso de Metodología de la Programación del Primer Año de Ingeniería de Sistemas de la UCSM de Arequipa, en el año 2023.

Datos:

- Población: 220 estudiantes.

- La cifra 4 es el coeficiente de confiabilidad para el nivel de confianza con el 94%.
- Las incógnitas p y q representan las probabilidades de éxito y fracaso.
- El valor 5 representa a E, que es el error seleccionado.

Fórmula:

$$n = \frac{4Np \cdot q}{E^2(N - 1) + 4p \cdot q}$$

$$n = \frac{4(220)(50)(50)}{5^2(220 - 1) + 4(50) \cdot (50)}$$

$$n = 142 \text{ (valor redondeado)}$$

1.5. Justificación

Es innegable que, hoy en día, los estudiantes utilizan herramientas de ayuda en el aprendizaje, sobre todo en asignaturas de programación, más aún con la irrupción de nuevas herramientas basadas en inteligencia artificial, surgieron múltiples instrumentos, una de estas presenta la empresa GitHub. Esta herramienta es llamada GitHub Copilot, y es un programador automático, basado en I.A. que ofrece sugerencias de estilo y autocompletado de código fuente, en este contexto si es bien orientada puede ser muy productiva. Este estudio se pretende medir el nivel de impacto en el rendimiento estudiantil.

Actualmente la evolución de las habilidades de programación, como la tecnología avanzada GitHub Copilot, que utiliza modelos de lenguaje natural para generar código automáticamente, está teniendo un impacto crucial para comprender cómo esta herramienta puede influir en la evolución de las habilidades de programación de los estudiantes. Por otra parte, la teoría del aprendizaje sugiere que

los estudiantes se adaptan a nuevas herramientas y tecnologías en su proceso educativo, de ahí la importancia de evaluar cómo los estudiantes asimilan y utilizan GitHub Copilot en su capacidad de adaptación de los programas académicos y su alineación con las demandas del mercado laboral. Asimismo, es importante analizar el impacto de GitHub Copilot en la creatividad y la destreza para encontrar soluciones a problemas computacionales cotidianos de los estudiantes.

En esta investigación es imprescindible realizar la comparación de los estudios de caso en entornos académicos controlados, comparando el rendimiento de los estudiantes que utilizan GitHub Copilot con aquellos que no lo hacen, logrando proporcionar datos valiosos sobre el impacto en términos de velocidad de desarrollo, calidad del código y comprensión de conceptos. Asimismo, es importante plantear preguntas éticas en términos de originalidad del código y plagio inadvertido; por consiguiente, evaluar su impacto práctico en abordar cuestiones éticas para el uso responsable de la tecnología en un entorno académico, alineándose con las expectativas del mundo real y las demandas de la industria.

En este contexto la aplicación de GitHub Copilot podría mejorar la productividad en el desarrollo del software, por otro lado, si es mal orientada podría hacer caer, a los estudiantes, en el facilismo, cayendo en la regla del menor esfuerzo en los estudiantes, en tal sentido es fundamental orientarla para que sea una fuente de ayuda, mejora del pensamiento computacional y de las habilidades de abstracción e implementación de software, he ahí la importancia de su estudio.

1.6. Definiciones

a. Aprendizaje:

Desde el punto de vista psicológico según (Pérez S. Antonio y Poveda S. Patricia, 2008) el aprendizaje es “cualquier cambio del organismo que origine una nueva forma de pensar y/o comportarse”.

Por otro lado, acorde a lo que indica (Gross, Z., Rutland, S., 2017) se puede definir que el aprendizaje es “el procedimiento mediante el cual se obtienen y perfeccionan las habilidades, conocimientos, comportamientos y valores”. Por ende, la enorme importancia de su análisis para la mejora del rendimiento académico de estudiantes universitarios.

b. GitHub Copilot:

Es un software generador de código, se podría definir según (GitHub, 2023) como: “Es un programador de pares de inteligencia artificial que ayuda a los usuarios a escribir código más rápido y con menos trabajo”. La comprensión de esta herramienta es fundamental en el presente trabajo y es de gran ayuda para la comunidad académica UCSM.

c. Rendimiento académico:

De acuerdo con los siguientes autores (Tejedor Tejedor, 2003) se puede mencionar que el rendimiento académico “Forma parte de una variable compuesta, en la que concurren diversos factores como: aptitud del estudiante, voluntad, esfuerzo, metodología de la enseñanza que ha recibido y que es una expresión, en cierto sentido, de todos los estudiantes, y que no solo refleja el resultado del

aprendizaje”. En este estudio se pretende evaluar cuando varía esta variable con la aplicación de GitHub Copilot. El rendimiento académico también se puede definir como “La manifestación de habilidades y rasgos psicológicos del alumno, desarrollados y mejorados a través del proceso de enseñanza, lo cual conduce a la obtención de logros académicos durante un período determinado, los cuales se resumen en una evaluación cuantitativa”(Chadwick, 1979).

d. Métricas del rendimiento académico:

Es fundamental comprender como medir el rendimiento académico, al cual podemos definir como “Una evaluación de las habilidades o señales adquiridas que indican de manera favorable lo que una persona aprendió como resultado de un proceso educativo” (Pizarro, 1985).

También se puede indicar que rendimiento “Es el propósito de todos los esfuerzos e iniciativas educativas emprendidas tanto por el docente como por el estudiante en la adquisición de conocimientos. El aporte del maestro se evalúa según el nivel de aprendizaje alcanzado por sus estudiantes” (Kaczynska, 1986).

e. Programación:

En el presente trabajo es fundamental definir que la programación de ordenadores es “El procedimiento de concebir y aplicar un conjunto de instrucciones que una computadora va a ejecutar. Ello implica la creación de algoritmos y la expresión de esos algoritmos en un lenguaje de programación, como Python, Java o C++, con el objetivo de lograr una tarea o resolver un problema específico” (Sebesta, 2015).

f. Lenguaje de programación

Según (Aguilar, 2008): "Los Lenguaje de programación son un sistema de normas y símbolos que habilita a los desarrolladores para crear programas de computadora. Estos lenguajes son esenciales para la redacción de programas, los cuales consisten en series de instrucciones codificadas como secuencias de dígitos numéricos entendibles por las computadoras".

g. Metodología de la programación

Se puede decir que "Es el enfoque sistemático y estructurado que los programadores adoptan para diseñar, implementar y mantener software. Involucra la planificación de algoritmos, la escritura de código fuente, la depuración y la documentación. La metodología de la programación busca aumentar la eficacia, la transparencia y la excelencia del código, facilitando el desarrollo de software más efectivo" (Sommerville, 2011).

h. Inteligencia artificial

Es un algoritmo concebido para llevar a cabo ciertas operaciones que se consideran características de la inteligencia humana, tales como el autoaprendizaje y el razonamiento. De manera más formal, se puede definir como: "La capacidad de una máquina para ejecutar tareas que normalmente requieren inteligencia humana, abarcando el aprendizaje, el razonamiento, la percepción y la interacción con el entorno. Los sistemas de Inteligencia Artificial emplean algoritmos y modelos matemáticos para analizar datos y tomar decisiones, emulando la inteligencia humana en una variedad de aplicaciones" (S. Russell & P. Norving, 2018).

i. Open I.A.

Es una empresa de investigación en inteligencia artificial (I.A.) cuya misión es avanzar en la exploración de I.A. de manera segura y beneficiosa para la humanidad. Fundada en 2015, OpenAI ha estado a la vanguardia de la investigación en aprendizaje profundo (Deep Learning) y ha desarrollado modelos de lenguaje avanzados, incluyendo aquellos utilizados en *GitHub Copilot*" (OpenAI, 2015).

j. Github

Es una plataforma en línea para el desarrollo colaborativo que se apoya en el sistema de control de versiones Git. Esta herramienta posibilita a los desarrolladores trabajar conjuntamente en proyectos de software, monitorizar cambios en el código fuente, administrar problemas y facilitar la integración continua. GitHub es usado de manera amplia en la comunidad de desarrollo de software para alojar y revisar código, así como para administra proyectos de manera eficiente" (GitHub, 2023).

1.7. Alcances y limitaciones

1.7.1. Alcances:

Evaluación de la implementación de *GitHub Copilot*: Este estudio se centró en la valoración específica de la implementación de la herramienta *GitHub Copilot* en el proceso de la adquisición de conocimientos y habilidades en la programación por parte de los alumnos de Primer Año de Ingeniería de Sistemas.

Impacto en el desarrollo de habilidades prácticas: Se investigó el impacto de *GitHub Copilot* en el desarrollo de habilidades prácticas de programación, analizando cómo la herramienta facilita el proceso de aprendizaje y aplicación de conceptos de programación.

Medidas de rendimiento académico: el desempeño educativo de los alumnos que participan en el programa de aprendizaje mejorado *GitHub Copilot* se analizó y comparó con un grupo de control que utiliza métodos de aprendizaje tradicionales.

Comentarios de los estudiantes: se recopilaron comentarios directos de los estudiantes, acerca de su experiencia con *GitHub Copilot*, explorando percepciones, desafíos y beneficios percibidos.

1.7.2. Limitaciones:

Duración del Estudio: La limitación temporal del estudio puede afectar la capacidad de capturar a largo plazo el impacto de *GitHub Copilot* en el rendimiento académico, ya que los efectos pueden desarrollarse a lo largo del tiempo y no específicamente durante el período 2023.

El espacio específico: Los resultados pueden ser específicos para la muestra seleccionada de la UCSM de Arequipa y no generalizarse completamente a otras universidades.

Factores Externos: Cambios en el plan curricular y otros eventos inesperados, pueden influir en los resultados que están fuera del control del estudio.

1.8. Cronograma

Imagen 1: Cronograma

EDT	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Trabajo de Investigación	60 días	lun 16/10/23	jue 7/03/24	
1.1	Capítulo I: Antecedentes del estudio	13 días	lun 16/10/23	sáb 11/11/23	
1.1.1	Título del tema	1 día	lun 16/10/23	mar 17/10/23	
1.1.2	Planteamiento del problema	2 días	mar 17/10/23	vie 20/10/23	3
1.1.3	Objetivos de la investigación	1 día	vie 20/10/23	mar 24/10/23	4
1.1.4	Metodología	2 días	mar 24/10/23	vie 27/10/23	5
1.1.5	Justificación	2 días	vie 27/10/23	mié 1/11/23	6
1.1.6	Definiciones	2 días	mié 1/11/23	lun 6/11/23	7
1.1.7	Alcances y limitaciones	2 días	lun 6/11/23	jue 9/11/23	8
1.1.8	Cronograma	1 día	jue 9/11/23	sáb 11/11/23	9
1.2	Capítulo II: Marco Teórico	7 días	lun 13/11/23	lun 27/11/23	2
1.2.1	Conceptualización de las variables	1 día	lun 13/11/23	mar 14/11/23	
1.2.2	Importancia de las variables	1 día	mar 14/11/23	jue 16/11/23	12
1.2.3	Análisis comparativo	1 sem	jue 16/11/23	lun 27/11/23	13
1.2.4	Análisis crítico	1 sem	jue 16/11/23	lun 27/11/23	14CC
1.3	Capítulo III: Marco referencial	11 días	lun 27/11/23	mié 20/12/23	11
1.3.1	Reseña Histórica	3 días	lun 27/11/23	lun 4/12/23	
1.3.2	Filosofía organizacional	3 días	lun 27/11/23	lun 4/12/23	17CC
1.3.3	Diseño organizacional	1 sem	lun 4/12/23	jue 14/12/23	18
1.3.4	Servicios	1 día	jue 14/12/23	vie 15/12/23	19
1.3.5	Diagnóstico organizacional	2 días	vie 15/12/23	mié 20/12/23	20
1.4	Capítulo IV: Resultados	10 días	mié 20/12/23	vie 26/01/24	16
1.4.1	Marco metodológico	1 sem	mié 20/12/23	mar 16/01/24	
1.4.2	Resultados	1 sem	mar 16/01/24	vie 26/01/24	23
1.5	Capítulo V: Sugerencias	5 días	vie 26/01/24	mar 6/02/24	22
1.5.1	Conclusiones	1 sem	vie 26/01/24	mar 6/02/24	
1.5.2	Recomendaciones	1 sem	vie 26/01/24	mar 6/02/24	26CC
1.6	Recomendaciones	14 días	mar 6/02/24	jue 7/03/24	25
1.6.2	Revisión Integral	2 sem.	mar 6/02/24	mié 28/02/24	27
1.6.1	Explicaciones de defensas y tribunales	4 días	mié 28/02/24	jue 7/03/24	29

Fuente: Elaboración propia

2. MARCO TEÓRICO

2.1. Conceptualización de tópicos clave

Es fundamental comprender ciertos conceptos clave para visualizar la profundidad del presente trabajo de investigación.

2.1.1. Inteligencia artificial

Formalmente se puede decir que: “Se puede describir como la habilidad de un sistema para ejecutar tareas que normalmente implican inteligencia humana como: el aprendizaje, el razonamiento, la percepción y la interacción con el entorno. Los sistemas de Inteligencia Artificial emplean algoritmos y modelos matemáticos para analizar datos y tomar decisiones, emulando la inteligencia humana en una variedad de aplicaciones”. (S. Russell & P. Norving, 2018).

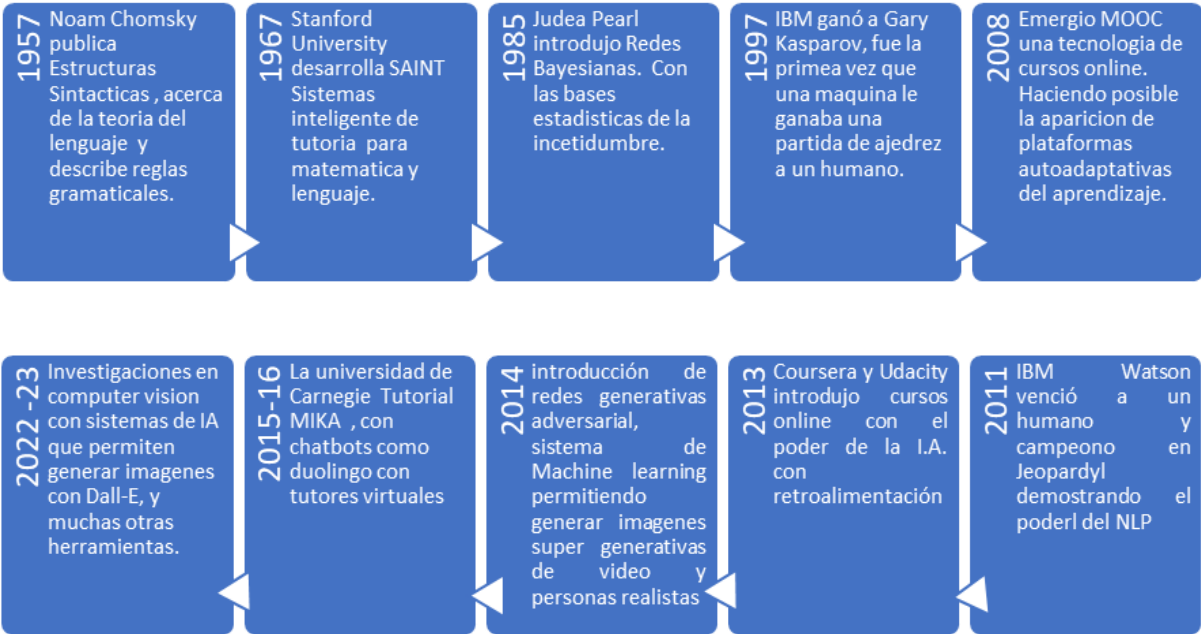
2.1.2. Contexto Histórico de la Inteligencia artificial

La inteligencia artificial (I.A. , *Artificial Intelligence*, o A.I.) consiste en replicar los procesos mentales humanos mediante máquinas, especialmente en sistemas informáticos. Estos procesos incluyen el aprendizaje, que implica adquirir información y reglas para su aplicación; el razonamiento, donde se utilizan estas reglas para llegar a conclusiones aproximadas o definitivas; y la capacidad de autocorrección. Algunas áreas de aplicación específicas de la inteligencia artificial son los sistemas expertos, el reconocimiento de voz y la visión artificial.

El concepto de inteligencia artificial, acuñado por el informático estadounidense John McCarthy en 1956 durante la Conferencia de Dartmouth, marca el nacimiento de esta

disciplina. Actualmente, se ha convertido en un término amplio que engloba desde la automatización de procesos robóticos hasta los avances actuales en robótica.

En la educación se aplicado la I.A. en diversos casos, a continuación, se muestra una figura donde se describe como a lo largo del tiempo la I.A. fue aplicándose en la educación.



Fuente: Elaboración propia

La inteligencia artificial (I.A.) ha ganado espacio recientemente debido, en parte, a los grandes volúmenes de datos, al aumento de velocidad, tamaño y variedad de datos que las empresas están recopilando (Big Data). I.A. puede realizar tareas tales como identificar patrones en los datos de manera más eficiente que los seres humanos, lo que permite a las empresas obtener más información sobre sus datos.

2.1.3. Lenguaje de programación (L.P.)

Como se dijo en puntos anteriores se planteó que L.P. es el grupo de sentencias mediante las cuales los seres humanos se comunican con las computadoras. De manera más formal, se define como “El conjunto de normas y símbolos formales, utilizados para escribir códigos fuente, que luego son traducidos a instrucciones ejecutables por una computadora.

Estos lenguajes permiten a los programadores comunicarse directamente con la máquina de manera estructurada y comprensible, facilitando la creación de software y el desarrollo de aplicaciones” (Sebesta, 2015). Por otro lado, existen cientos de lenguajes de programación, pero esta investigación se centra en Python por su simplicidad.

2.1.4. Python

Según su creador Python es “Un lenguaje de programación de cuarta generación, interpretado y típicamente accesible con una lectura y aprendizaje sencillo. Se usa en desarrollo web, inteligencia artificial, análisis de datos y otras áreas por su sintaxis clara y enfoque en la legibilidad del código. Python fomenta la eficiencia y la productividad del programador” (G. Van Rossum & F.L. Drake, 2011).

2.1.5. IDE Visual Studio Code

Los lenguajes de programación deben ser desarrollados en un entorno, una interfaz adecuada conocida como IDE. Por lo tanto, este IDE es un entorno integrado de programación, es gratuito y de código *open source* o abierto, que fue desarrollada por Microsoft. Este IDE es conocido por su versatilidad y soporte para múltiples lenguajes de programación. Proporciona herramientas bastante avanzadas de edición

de código, depuración, control de versiones y una amplia gama de extensiones, lo que lo convierte en una opción popular entre los desarrolladores (Microsoft, 2023).

2.1.6. Programación

Según (Mathieu, 2014), en su libro *Introducción a la Programación*, es el procedimiento de elaborar un conjunto de sentencias que indican a una computadora cómo llevar a cabo una determinada tarea. Engloba también todas las acciones requeridas para garantizar el correcto funcionamiento del código y la consecución del objetivo para el que fue diseñado.

2.1.7. Algoritmos

Según (T. H. Corme, C. E. Rivest & C. Stein, 2019), en el libro *Introduction to algorithms* del MIT, definen como un conjunto de instrucciones o reglas definidas, precisas y finitas que usualmente posibilitan resolver un problema, realizar un cálculo, manipular datos y ejecutar otras acciones o actividades. Por otro lado, La definición de algoritmos proporcionada por Knuth (1997) es similar a la de Cormen et al. (2019). Sin embargo, Knuth enfatiza el hecho de que los algoritmos se utilizan para resolver problemas.

2.1.8. Complejidad Algorítmica

Según (Thomas H. Cormen, C. Leiserson, R. Rivest & C. Stein), en el libro *"Introduction to Algorithms"*, 4ta edición: dice: "El análisis de la complejidad algorítmica implica examinar la eficacia de los algoritmos en relación con el tiempo que requieren para ejecutarse y con el espacio que ocupan".

En otro sentido, (Knuth, 1997), en el libro *The Art of Computer Programming*, define la complejidad algorítmica como: “La ciencia que se ocupa de evaluar la eficiencia de los algoritmos computacionales. La complejidad algorítmica se centra en el tiempo y el espacio necesario para resolver un problema basado en algoritmos de forma eficiente”.

La complejidad algorítmica puede tener implicaciones en el desempeño estudiantil. Al estudiar algoritmos y comprender su complejidad, los estudiantes pueden desarrollar habilidades de análisis y resolución de problemas más avanzados, asimismo, el estudio de la complejidad algorítmica puede ayudar a los estudiantes a comprender cómo evaluar la eficiencia de los algoritmos y tomar decisiones informadas sobre qué algoritmo utilizar en diferentes situaciones.

En el año 2018, (Sebastián A. Furtado, Rafael Furtado & Gustavo Machado, 2018), realizaron un estudio comparativo donde se analizó dos enfoques para enseñar algoritmos:

- Enfoque tradicional: Este enfoque se centra en la memorización de conceptos y definiciones de algoritmos.
- Enfoque basado en la resolución de problemas: En este enfoque, la complejidad algorítmica se enfoca en la aplicación de algoritmos para resolver problemas concretos.

La investigación involucró a 140 estudiantes de ciencias de la computación de una universidad en Brasil. Estos estudiantes fueron divididos al azar en dos grupos de estudio. Los hallazgos revelaron que aquellos que fueron expuestos al enfoque

centrado en la resolución de problemas superaron en rendimiento a los que siguieron el enfoque tradicional durante los exámenes.

2.1.9. Ética en la I.A. para la educación

La ética tiene un rol prioritario en el desarrollo e implementación de la inteligencia artificial para la educación. Es importante asegurar que la I.A. se emplee con ética y de forma responsable, y que no se utilice para discriminar o dañar a los estudiantes. Algunos de los principios éticos clave que deben aplicarse a la I.A. en la educación incluyen:

- La equidad: La I.A. debe utilizarse de manera equitativa para todos los estudiantes, independientemente de sus antecedentes socioeconómicos, etnia, género u otros factores.
- La privacidad: Los datos de los estudiantes deben protegerse de la recopilación y el uso indebido.
- La seguridad: Los sistemas que involucren I.A. deben dar seguridad y protección a los alumnos.
- La transparencia: Es esencial que los sistemas de I.A. sean comprensibles y que los alumnos puedan comprender su funcionamiento.
- La responsabilidad: Los desarrolladores y usuarios de sistemas de I.A. deben ser responsables de sus acciones.

Para garantizar que la I.A. se utilice de manera ética en la educación, es importante que los gobiernos, las instituciones educativas y los desarrolladores de tecnología trabajen juntos para desarrollar y aplicar políticas y procedimientos adecuados. Estas políticas y procedimientos deben abordar los principios éticos

mencionados anteriormente y deben ser revisados y actualizados periódicamente a medida que la tecnología evoluciona.

Además de las políticas y los procedimientos, también es importante que los educadores estén capacitados en ética de la I.A. Los educadores deben comprender los principios éticos que se aplican a la I.A. y deben estar en condiciones de ayudar a los estudiantes a utilizar la I.A. de manera responsable.

Al enfrentar los dilemas éticos relacionados con la I.A. en la educación, podemos contribuir a asegurar que sea empleada de forma responsable y ética en beneficio de todos los estudiantes.

2.1.10. Rendimiento Académico

Existen varias definiciones, pero de acuerdo con (Bisquerra, 2009), en el libro Evaluación de la calidad educativa, “el desempeño académico es la evaluación de la efectividad con la que un estudiante alcanza los objetivos de aprendizaje establecidos por un programa educativo”. Por otro lado, para (J.A. Rodríguez, M.P. Sánchez & M.J. García, 2018) el desempeño académico es un concepto complejo que abarca dimensiones cognitivas, emocionales y conductuales”.

El rendimiento académico, es un factor fundamental y prioritario en la evaluación de la calidad educativa. Se dice que los estudiantes con un buen rendimiento académico son más propensos a tener éxito en su vida profesional y personal. Naturalmente, existen otros factores que influyen en el éxito profesional,

pero el rendimiento es fundamental, y es importante analizar los diversos métodos de evaluación de este.

2.1.11. Métodos de evaluación del rendimiento

Existen diferentes métodos para evaluar el rendimiento académico, los más utilizados son los siguientes:

Exámenes. Los exámenes son una forma tradicional de evaluar el rendimiento académico. Los exámenes pueden ser escritos, orales o prácticos.

Tareas. Las tareas son diversas actividades que los alumnos deben llevar a cabo como parte de su trabajo académico. Las tareas pueden incluir proyectos, informes, ensayos, etc.

Participación en clase. La participación en clase es una forma de evaluar el compromiso de los estudiantes con el aprendizaje. La participación en clase puede incluir preguntas, respuestas, debates, etc.

Autoevaluación. La autoevaluación es una forma de que los estudiantes reflexionen sobre su propio aprendizaje. La autoevaluación puede realizarse a través de cuestionarios, escalas de valoración o diarios de aprendizaje.

2.1.12. Factores de evaluación del rendimiento

Factores para considerar en la valoración del desempeño o rendimiento académico.

En la evaluación del rendimiento académico, es importante considerar los siguientes factores:

Los objetivos de aprendizaje: son lo que los estudiantes deben aprender, los profesores deben asegurarse de que las evaluaciones estén alineadas con los fines de aprendizaje.

La diversidad de los alumnos: cada uno de ellos tienen diferentes fortalezas y debilidades. Los profesores deben utilizar una variedad de métodos de evaluación para lograr conseguir un resultado completo del rendimiento de los estudiantes.

Las condiciones de evaluación: deben ser justas y equitativas para todos los estudiantes.

Es importante indicar que la evaluación del rendimiento académico es un proceso importante que debe realizarse de forma cuidadosa y reflexiva. Los Catedráticos deben considerar los diferentes métodos de evaluación disponibles y deben utilizar una variedad de métodos para conseguir un resultado completo del rendimiento de los estudiantes.

2.1.13. Recursos educativos que aplican I.A.

La inteligencia artificial (I.A.) se está utilizando cada vez más en los recursos para la educación destinados a potenciar el aprendizaje de los alumnos. Existen diversos recursos educativos que están aplicando Inteligencia Artificial, uno de los

más conocidos es el portal de capacitación inteligente llamado *Knewton*, (Knewton, 2023) que permite capacitación personalizada y automática con I.A. en el contenido de diversos temas.

- **Tutores virtuales.** Son programas informáticos que pueden proporcionar a los estudiantes apoyo y retroalimentación personalizada. Los tutores virtuales pueden utilizar I.A. para analizar el progreso de los estudiantes y adaptar su instrucción en consecuencia.
- **Aplicaciones de aprendizaje adaptativo.** Las aplicaciones de aprendizaje adaptativo utilizan I.A. para adaptar el contenido y la instrucción al nivel de habilidad de cada estudiante. Estas herramientas pueden asistir a los alumnos en su proceso de aprendizaje individualizado y concentrarse en las áreas donde requieren mayor refuerzo.
- **Herramientas de evaluación automatizadas.** Las herramientas de evaluación automatizadas utilizan I.A. para calificar tareas y exámenes. Estas herramientas pueden ayudar a los profesores a ahorrar tiempo y a proporcionar retroalimentación más rápida y precisa a los educandos.
- **Plataformas de aprendizaje en línea.** Los sistemas de aprendizaje en línea emplean Inteligencia Artificial para adaptar la experiencia educativa de los estudiantes de manera personalizada. Estas plataformas pueden utilizar I.A. para recomendar contenido relevante, proporcionar retroalimentación y seguimiento del progreso.

2.1.14. Knewton

Es una plataforma que utiliza I.A. para personalizar los cursos digitales, adaptando el contenido en tiempo real según el progreso del estudiante. La plataforma de *Knewton* recopila datos sobre el progreso de los estudiantes, analiza su desempeño y preferencias de aprendizaje; luego recomienda recursos y actividades específicas para mejorar su comprensión en áreas problemáticas.

Utiliza algoritmos de Aprendizaje Automático (*Machine Learning*), para ajustar continuamente el contenido del curso, la dificultad de las preguntas y la secuencia de aprendizaje, con el fin de optimizar la experiencia de aprendizaje de cada estudiante. La plataforma de aprendizaje analiza extensas cantidades de datos anónimos mientras los educandos emplean productos con tecnología de *Knewton* durante períodos prolongados de tiempo, dicha plataforma es adaptativa. *Knewton* usa *Amazon Elastic Map Reduce* (Amazon EMR) para analizar de manera simultánea grandes cantidades de datos en la plataforma y ajustarlos a las recomendaciones de los modelos analíticos.

2.2. Importancia de la Inteligencia Artificial en la educación

La I.A. tiene gran cantidad de aplicaciones en la educación universitaria dando en primer lugar el soporte a la investigación a través de diversas tareas como, por ejemplo: el análisis de datos, generación de modelos, análisis de publicaciones y también podría generar automáticamente manuscritos. En segundo lugar, el desarrollo de planes de estudio y contenido a través de tareas como, por ejemplo: generación de mallas curriculares, creación de contenidos (separatas, presentaciones,

evaluaciones, entre otros), generación de recursos interactivos (realidad virtual, realidad aumentada, laboratorios remotos).

2.2.1. Análisis comparativo

La I.A. tiene la capacidad de transformar completamente el ámbito educativo, en ese sentido la I.A. ofrece una serie de ventajas potenciales sobre los métodos de enseñanza tradicionales, en el presente estudio de investigación se realizó el comparativo entre la herramienta *GitHub Copilot* y los métodos tradicionales aplicados al proceso de enseñanza de la programación en los educandos de Primer Año de Ingeniería de Sistemas de la UCSM, algunas de las ventajas de la herramienta *GitHub Copilot* métodos tradicionales son:

- a. **Personalización:** La I.A. puede utilizarse para adaptar el contenido y la instrucción al nivel de habilidad de cada estudiante. Esto puede aportar en los educandos a progresar a su propia velocidad de aprendizaje y a centrarse en las áreas que necesitan más apoyo.
- b. **Efectividad:** La I.A. puede utilizarse para proporcionar a los estudiantes retroalimentación más rápida y precisa. Esto puede asistir a los estudiantes a reconocer sus áreas de oportunidad de mejora y a emprender acciones para optimizar su proceso de aprendizaje.
- c. **Eficiencia:** La I.A. puede utilizarse para automatizar tareas que actualmente realizan los profesores, como la puntuación de las evaluaciones y tareas. Esto puede otorgar a los profesores más tiempo para

dedicarse a labores de mayor relevancia, como la preparación de clases y brindar atención personalizada a los educandos.

2.3. Análisis crítico

2.3.1. Desafíos de la I.A. en la educación

A pesar de las ventajas potenciales de la I.A., también existen algunos desafíos que deben abordarse antes de que la I.A. pueda ser plenamente adoptada en las aulas. Estos desafíos incluyen:

- **Costo:** La implementación de la I.A. en las aulas puede ser costosa. Los gobiernos y las instituciones educativas deben invertir en el desarrollo e implementación de recursos educativos basados en I.A.
- **Accesibilidad:** La I.A. puede ser una herramienta poderosa para el aprendizaje, pero solo si es accesible para todos los estudiantes. Las instituciones educativas deben asegurar de que los estudiantes tengan debido acceso a los recursos educativos basados en I.A.
- **Equidad:** La I.A. debe utilizarse de manera equitativa para todos los estudiantes, sin importar sus antecedentes socioeconómicos u otros factores. Las instituciones educativas deben desarrollar políticas y procedimientos para asegurar que la I.A. se emplee de manera justa.

3. MARCO REFERENCIAL. TRABAJO DE INVESTIGACIÓN

3.1. Reseña Histórica

La Universidad Católica de Santa María (UCSM, 2023) fue fundada el 6 de diciembre en el año 1961 por el padre William Morris, quien llegó Arequipa con el fin de fundar un Claustro de Estudios superiores, siendo, por primera vez, ubicada en la 4ta. cuadra de la calle Santa Catalina. La primera locación fue una vieja casona, con el nombre de “Universidad de Santa María”, se sabe que luego de dos años la locación fue cambiada a la sede de Umacollo, donde actualmente se mantiene.

La apertura de la Universidad a los estudiantes se llevó a cabo el 9 de abril de 1962, este comienzo se realizó con 150 alumnos y 8 maestros fundadores; en dicha fecha fue la 3ra. universidad privada en el Perú y fue la primera en provincia. En la década de 1970, se emite la Resolución de la definitiva autorización a la UCSM con el nombre “Universidad Católica de Santa María” y se crean los Departamentos Académico Humanidades y Ciencias Sociales; así como los departamentos Ciencias Físicas, Naturales y Matemáticas, también se crean los departamentos de Servicio Social y Ciencias de la Comunicación Social; Derecho y Ciencias Clínicas. Se logró consolidar la infraestructura, el desarrollo académico y administrativo, logrando ajustar al Sistema Nacional de la Universidad Peruana llevando a cabo por primera vez la evaluación a todos los docentes.

Posteriormente, en la década de 1980, se fundan el Programa profesional de Ciencias de la educación Inicial y así como, las Facultades de Histórico Sociales, Trabajo Social, Enfermería e Ingeniería, con los Programas: Sistemas, Agronomía, Alimentaria, Veterinaria y el programa de Medicina; funcionan 14 facultades, 38

Programas Profesionales y 27 Especialidades. Durante estas décadas, la UCSM continuó su desarrollo, adaptándose a los cambios en la educación y la sociedad; se modernizaron las instalaciones y se implementaron tecnologías emergentes para mejorar la experiencia educativa.

En la década de 1990, la universidad estableció su primera escuela de post grado, renovó los sistemas informáticos y amplió la infraestructura. Con el tiempo, la institución inició una organización académica - administrativa de todas sus escuelas profesionales, reconstruyendo los programas que habían caracterizado a la mayoría de las especializaciones desde sus inicios.

En el siglo XXI, todas las carreras importantes tuvieron reformas curriculares para adaptarse a las leyes universitarias peruanas vigentes. En dicha reforma curricular se incluyó a las áreas de “Estudios Generales”, “Investigación Formativa” y “Responsabilidad Social” para complementar la formación de profesionales egresados de la UCSM. Posteriormente en el año 2017 se consiguió la acreditación obtenida por la SUNEDU, dando legitimidad y credibilidad a la institución y sus alumnos. Adicionalmente se otorga el Licenciamiento UCSM, Patrimonio Cultural de Arequipa, así como la Certificación ISO 9001.

En los últimos años la UCSM ha crecido en cantidad de alumnado, así como en infraestructura, el proceso de pandemia contribuyó al aumento de la cantidad de estudiantes, el año 2022 se tuvo 19,300 estudiantes en las 29 escuelas.

3.2. Filosofía organizacional

3.2.1. Misión de la UCSM

Según (UCSM, 2023), la misión de la Universidad Católica de Santa María es declarada como: “Formar profesionales e investigadores líderes con principios éticos y humanísticos, en su compromiso con el desarrollo de la ciencia, innovación, tecnología y la mejora de la calidad de vida en la sociedad”.

3.2.2. Visión de la UCSM

De acuerdo (UCSM, 2023), la visión de la Universidad Católica de Santa María es declarada como: “Ser reconocida por su excelencia académica, por el desarrollo científico pertinente, por la formación de personas competentes en su área de actuación y socialmente responsables”.

3.2.3. Valores de la UCSM

- Perseverancia.
- Sostenibilidad.
- Transparencia.
- Ética.
- Resiliencia.
- Liderazgo.
- Competencia.
- Autocrítica.
- Tolerancia.
- Solidaridad.

- Originalidad.

3.2.4. Propósito de la UCSM

Según (UCSM, 2023), el propósito es “Fomentar el desarrollo de aprendizaje colaborativo entre estudiantes y profesores como una actividad que promueve la creación de conocimiento y experiencias sociales, tanto en el contexto de las disciplinas especializadas como en el ámbito más amplio de la cultura y las ciencias humanas”.

3.2.5. Misión de la Facultad de Ciencias e Ingenierías Físicas y Formales

La misión de la facultad según (UCSM, 2023) es: “Educar de manera integral a ingenieros altamente competitivos, críticos y creativos, que se inspiren en principios cristianos, valoren su identidad cultural y posean sólidos principios éticos. Buscando constantemente la excelencia en su formación, fomentando la investigación y el progreso tecnológico, con el fin de anticipar y satisfacer las necesidades de la sociedad”.

3.2.6. Visión de la Facultad de Ciencias e Ingenierías Físicas y Formales

La visión de la FCIFF según (UCSM, 2023) es declarada como: “Formar integralmente profesionales de ingeniería altamente competitivos, críticos, creativos, inspirados en principios cristianos, con sentido de eco identidad, y valores éticos, buscando siempre la excelencia en su formación, fomentando la investigación y el desarrollo tecnológico; orientado a anticipar y satisfacer los requerimientos de la sociedad”.

3.2.7. Misión de la Escuela Profesional de Ingeniería de Sistemas

Según (UCSM, 2023), la misión de la Escuela Profesional de Ingeniería de Sistemas (EPIS) es “Formar integralmente Profesionales de Ingeniería altamente competitivos, críticos, creativos, inspirados en principios cristianos, con sentido de eco identidad, y valores éticos, buscando siempre la excelencia en su formación, fomentando la investigación y el desarrollo tecnológico; orientado a anticipar y satisfacer los requerimientos de la sociedad”.

3.2.8. Visión de la Escuela Profesional de Ingeniería de Sistemas

La visión de la EPIS acorde a (UCSM, 2023) es “Ser la Facultad líder en la enseñanza de ingeniería, con nivel de competencia nacional e internacional, consolidada y reconocida por su calidad en la formación de profesionales íntegros con excelencia académica y responsabilidad social contando con infraestructura de vanguardia, que responda a las necesidades globales y contribuyendo al desarrollo sostenible a través de sus egresados, sus investigaciones y programas de extensión”.

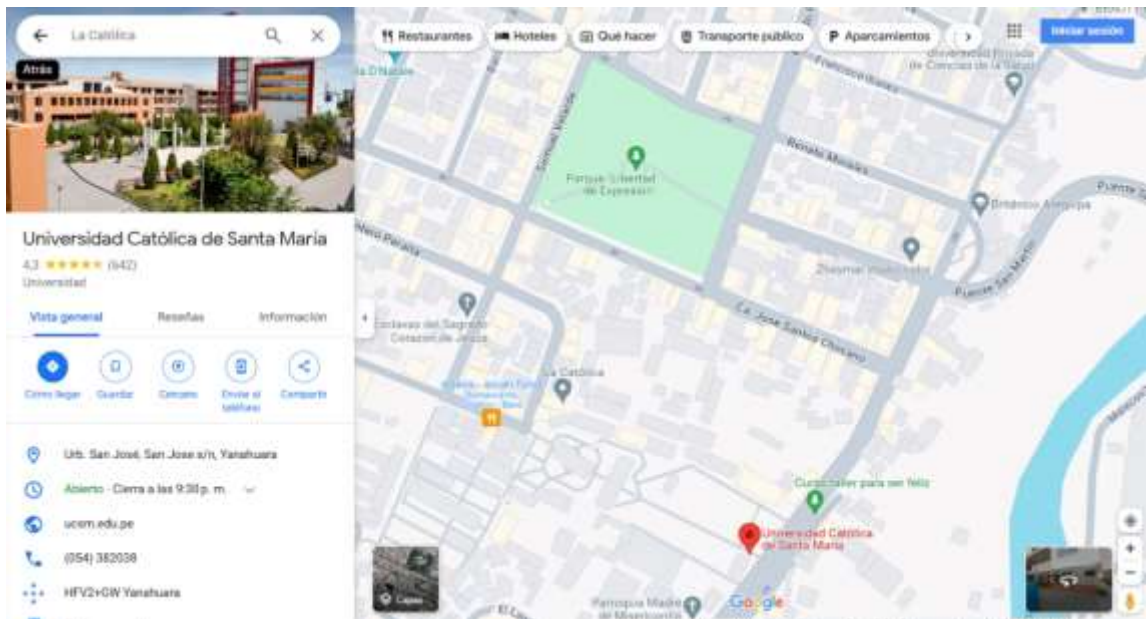
3.3. Diseño organizacional

3.3.1. Datos generales

Según (UCSM, 2023) los datos son los siguientes:

- Nombre y razón social: UNIVERSIDAD CATOLICA DE SANTA MARIA.A.
- RUC: 20141637941
- Contactos: 054-382038
- Dirección: Urbanización San José (Umacollo), Cercado, Arequipa, Perú
- Ubicación:

Imagen 2: *Ubicación Geográfica de la Universidad Católica de Santa María*

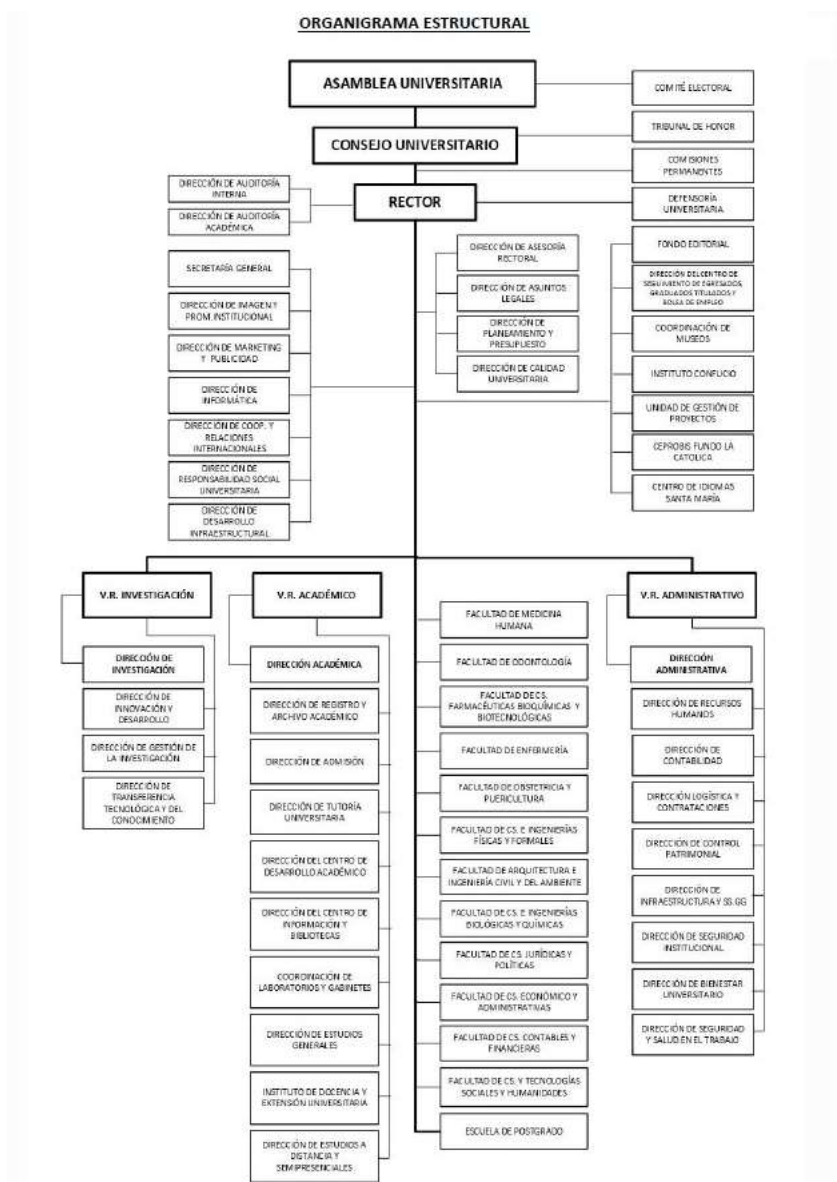


Fuente: Imagen obtenida de Google Maps

3.3.2. Organigrama estructural

La entidad máxima de nuestra universidad está dada por la Asamblea Universitaria, seguido por el Consejo Universitario, en tercera línea tenemos al Rector y luego el Vicerrectorado de Investigación, Académico y Administrativo. El organigrama se muestra a continuación:

Imagen 3: Organigrama estructural de la UCSM



Fuente: Portal oficial de la UCSM

3.4. Productos y/o servicios

3.4.1. Escuelas Profesionales de pregrado en modalidad presencial por áreas

Según (UCSM, 2023) actualmente la UCSM tiene 29 Escuelas Profesionales, las que se encuentran distribuidas en 4 áreas: Ciencias Sociales, Ciencias Jurídicas

y Empresariales, Ciencias de la Salud y Ciencias e Ingenierías. La EPIS pertenece al área de Ciencias e Ingenierías.

Imagen 4: *Admisión para pregrado modalidad presencial UCSM*



Fuente: Portal Institucional de la UCSM

3.4.2. Escuelas Profesionales de pregrado en modalidad a distancia

A partir del presente año según (UCSM, 2023), se consolidó como la primera universidad del sur del país en lograr el Licenciamiento de 12 carreras profesionales en la modalidad a distancia, semipresencial, dentro de las cuales se pueden mencionar: Ciencia Política y Gobierno, Diseño Gráfico Publicitario, Ingeniería Agronómica y Agrícola, Contabilidad, Economía y Finanzas, Educación Secundaria, Marketing y Gestión Comercial, Marketing Digital, Teología, Trabajo Social, Turismo y Gastronomía.

Imagen 5: Admisión para pregrado modalidad virtual UCSM



Fuente: Portal Institucional de la UCSM

3.4.3. Postgrado

La UCSM ofrece Maestrías en las modalidades presenciales y semipresenciales; adicionalmente Doctorados. Estas se dividen en cinco áreas: Ciencias Jurídicas, Ciencias de la Salud, Ciencias Económicas y financieras, Ciencias Sociales, Ciencias e Ingenierías haciendo en total 22 Maestrías.

Imagen 6: Admisión para postgrado EPG UCSM

A screenshot of the EPG UCSM admission page. The header includes the EPG logo (Universidad Católica de Santa María, Escuela de Postgrado) and navigation links: La Escuela, Maestrías, Doctorados, 100% Virtuales, and Admisión. The main content area features a black and white photo of three students in profile, with the text 'Admisión EPG UCSM'. Below the photo, there are two sections: 'Inicio de clases' with a green arrow icon and the text 'Maestrías y Doctorados Virtuales: 23 de octubre del 2023', and 'Duración' with a green arrow icon and two bullet points: 'Doctorados Virtuales: Seis ciclos académicos (tres años)' and 'Maestrías Virtuales: Tres ciclos académicos (un año y medio)'. The 'Admisión' link in the header is underlined.

Fuente: Portal Institucional de la UCSM

3.4.4. Servicios

3.4.4.1. Servicios artísticos culturales

La UCSM ofrece a todos los estudiantes pertenecer a los siguientes grupos:

a) Tuna Universitaria

Agrupación musical que ejecuta diversos ritmos, está compuesta por estudiantes universitarios que se reúnen para tocar música, cantar y actuar en eventos culturales, festivales, festividades universitarias.

Imagen 7: *Tuna Universitaria*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

b) Elenco de Teatro:

Un conjunto artístico cuya meta es fomentar, revitalizar y difundir el arte teatral, además de impulsar el entretenimiento y el bienestar de los miembros del elenco. Esto se logra mediante el desarrollo de habilidades artísticas y escénicas, así como la expresión corporal y el dominio del texto o guion teatral, junto con la participación en talleres artísticos de clown y títeres.

Imagen 8: *Elenco de teatro*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

c) Orquesta sinfónica:

Se trata de un conjunto musical que busca fomentar en la comunidad universitaria una apreciación positiva por las artes musicales en sus formas sinfónicas o de cámara.

Imagen 9: *Orquesta Sinfónica*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

d) Rondalla Santa Mariana:

Su objetivo es expresar alegría a través de la música y el principio de vivir y compartir entre las integrantes cada reto, triunfo y adversidad como una familia. El estilo musical de la rondalla inició con instrumentos acústicos, agregándose progresivamente instrumentos como la batería, el bajo electrónico, la flauta travesa, el charango, el cajón.

Imagen 10: *Rondalla santa mariana*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

e) Ballet Folclórico:

El propósito del grupo de danza es promover y difundir las expresiones artísticas de la danza a nivel regional, nacional e internacional.

Imagen 11: *Ballet Folclórico*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

3.4.4.2. Servicios deportivos

La UCSM considera la importancia de demostrar por parte de los educandos las habilidades que tienen en las diversas disciplinas de deportes: Vóley, Fútbol/Futsal, Básquet, Técnica de tiro, Karate, Taekwondo, Kung fu, Judo, Ajedrez y Atletismo.

Imagen 12: *Equipo de Vóley*



Fuente: Portal Institucional de la UCSM (UCSM, 2023)

3.5. Diagnóstico organizacional

La Escuela Profesional de Ingeniería de Sistemas en el año 2022 tuvo 229 ingresantes, en el presente año 2023 la cantidad de alumnos matriculados en segundo año es de 174 estudiantes, lo cual denota que hay una deserción o repetición en un 24% aproximadamente y es importante desarrollar el análisis exhaustivo para decidir adecuadamente. Para realizar un análisis más completo del diagnóstico organizacional se presenta un FODA de la EPIS en la UCSM:

Tabla 1: FODA Escuela de Ingeniería de Sistemas

FORTALEZAS <ul style="list-style-type: none">- Catedráticos Calificado: La escuela cuenta con un cuerpo docente altamente capacitado, el 40% con grado de doctor y el 100% con grado de magister.- Infraestructura Tecnológica: Dispone de laboratorios equipados con tecnología de punta, proporcionando a los estudiantes un entorno propicio para el aprendizaje.- Colaboraciones con empresas e instituciones, permitiendo oportunidades de pasantías y proyectos conjuntos.- Prestigio, más de 30 años de formación de prestigiosos profesionales insertados en el mundo laboral avalan una buena reputación.	OPORTUNIDADES <ul style="list-style-type: none">- Demanda Creciente: El crecimiento continuo en el sector de tecnología y sistemas ofrece oportunidades para aumentar la demanda de programas académicos.- Colaboración Industrial: Mayor integración con empresas para desarrollar programas académicos que satisfagan las necesidades del mercado laboral.- Desarrollo Tecnológico: Participación en proyectos que impliquen investigación y desarrollo tecnológico, los cuales pueden contribuir al prestigio de la escuela y generar nuevas oportunidades de financiamiento.
DEBILIDADES <ul style="list-style-type: none">-Lentitud en la modificación del Plan Curricular de estudio, puede resultar en la falta de alineación con las demandas del mercado laboral, debido a la gran cantidad de cambios tecnológicos.- Recursos Financieros Limitados: La escuela puede enfrentar restricciones presupuestarias que limiten la adquisición de nuevas tecnologías y la mejora de la infraestructura.	AMENAZAS <ul style="list-style-type: none">- Competencia de Otras instituciones universidades: Entidades que ofrecen programas similares podría afectar la captación de estudiantes.- Impacto de Crisis Económicas: Las crisis dificultan la disponibilidad de recursos y financiamiento, lo que podría impactar negativamente en la calidad de la educación ofrecida.

Fuente: Elaboración propia

En este contexto y considerando que uno de los cursos base es Metodología de la Programación.

Analizaremos los siguientes puntos:

3.5.1. Contexto Institucional

La malla curricular vigente del año 2021 muestra en el primer semestre de la EPIS el curso Metodología de la Programación, el cual tiene un peso 5 créditos y se dicta 2 horas de teoría y 4 horas de práctica de forma semanal durante 14 semanas. Este curso tiene por objetivo enseñar de forma abstracta la lógica que se aplica a la hora de programar líneas de código.

3.5.2. Análisis del proceso de Enseñanza actual

La metodología aplicada actualmente en el dictado del curso en mención es de forma tradicional haciendo uso del software que nos permite manejar diagramas de flujo (DFD), algoritmos (*PSeint*) hasta lenguajes de programación como Python. Un gran desafío que se tiene es aplicar una metodología en la que se utilice herramientas de inteligencia artificial con el objetivo de incrementar la comprensión de los estudiantes y disminuir el nivel de deserción de la carrera.

De acuerdo con una encuesta efectuada a los catedráticos de la UCSM en la capacitación docente llamada “La Inteligencia Artificial en el Proceso Educativo Universitario” efectuada el 05 de marzo de 2024, ante la pregunta ¿Emplean o estarían dispuestos a usar la I.A. en la UCSM? con las siguientes alternativas de respuesta:

- a. Sí.
- b. No
- c. Prefiero conocer más de la I.A.

El 87% respondió de forma afirmativa, 13% indicó que prefiere conocer más de la I.A. y no hubo ninguna respuesta negativa, es decir 0% en la respuesta No.

3.5.3. Revisión de la Herramienta GitHub Copilot:

Según (A. S. Al-Sadoon y S. M. Al-Aqeeli, 2022) en su trabajo de investigación evaluó el uso de *GitHub Copilot* en un curso de programación de nivel introductorio para estudiantes universitarios, los resultados mostraron que los estudiantes que utilizaron *GitHub Copilot* obtuvieron mejores resultados en las tareas de programación que los estudiantes que no lo utilizaron.

De acuerdo con el estudio de caso efectuado por (M. A. Al-Khafaji y A. S. Al-Sadoon, 2022) los resultados mostraron que los estudiantes que utilizaron *GitHub Copilot* se sintieron más motivados para aprender programación y que mejoraron su comprensión de los conceptos de programación.

Se obtuvo el siguiente resultado *GitHub Copilot* puede tener un impacto positivo en el aprendizaje de la programación, ya que se requiere continuar la investigación para determinar su eficacia a largo plazo de acuerdo con la investigación (J. A. Pérez-González y M. A. Jiménez-Sánchez, 2022).

Acorde al estudio realizado por (S. S. Al-Sharif, A. S. Al-Sadoon y M. A. Al-Khafaji, 2023) donde se investigó el uso de *GitHub Copilot* para apoyar diferentes estilos de aprendizaje en la educación de la programación. Los resultados mostraron que *GitHub Copilot* puede ser una herramienta útil para apoyar a estudiantes con diferentes estilos de aprendizaje.

3.5.4. Competencias y capacidades para alcanzar por los estudiantes del curso de Metodología de la Programación

Acorde a la matriz de competencias tomado de (UCSM, 2023), los estudiantes del curso en mención deben alcanzar las siguientes capacidades con sus respectivas competencias:

- **Diseño en Ingeniería:** Diseña un sistema, producto o proceso en el campo de la ingeniería de sistemas que satisface necesidades y requerimientos, considerando salud pública, seguridad y bienestar, así como factores globales, culturales, sociales, ambientales y económicos.
 - Propone y compara diferentes alternativas de solución, y selecciona la más adecuada satisfaciendo las especificaciones.
 - Desarrolla la solución haciendo uso de los métodos, técnicas, normas y estándares de ingeniería apropiados.
 - Presenta y describe la solución en forma gráfica a través de planos, simulaciones virtuales y diagramas.
- **Ética y Responsabilidad Profesional:** Desarrolla un comportamiento ético y asume responsabilidad por los proyectos y trabajos realizados, tomando decisiones de manera informada y justa.
 - Valora el cumplimiento puntual y responsable de sus actividades.
 - Toma en consideración el interés común y el beneficio social.
 - Respeto la propiedad intelectual y reconoce la autoría de trabajos y proyectos de otras personas.

4. PROPUESTA DE MEJORA

4.1. Identificación del área a mejorar

En el presente trabajo se desarrolló una propuesta de mejora, aplicada en el proceso de adquisición del conocimiento para el dictado del curso Metodología de la Programación en los alumnos de Primer Año de Ingeniería de Sistemas de la UCSM, incluyendo estrategias para los docentes de dicho curso y la integración de *GitHub Copilot* en el plan de estudios y seguimiento continuo.

4.2. Diagnóstico

4.2.1. Tipo de investigación

La investigación desarrollada tiene como base el método analítico – descriptivo, puesto que esta investigación tiene como objetivo principal mejorar el proceso de enseñanza de la programación, basada en la herramienta *GitHub Copilot*, en los estudiantes de Primer Año de Ingeniería de Sistemas de la Universidad Católica de Santa María. A continuación, se detalla los tipos de investigación usados en el desarrollo del presente trabajo.

Según (Científica, s.f.) “La investigación de campo se caracteriza como un enfoque cualitativo que implica la recopilación directa de datos o la observación en el entorno natural de las personas u objetos de estudio. En otras palabras, esta metodología se basa en la recolección de datos originales o primarios”.

Investigación descriptiva: según Tamayo y Tamayo M., en su libro Proceso de Investigación Científica, “La investigación descriptiva implica la detallada descripción, registro, análisis e interpretación de la naturaleza actual, composición o proceso de

los fenómenos. El énfasis se centra en las conclusiones predominantes o en grupos de personas, objetos o eventos”.

4.2.2. Características de la población

En la presente investigación la población objetivo la conformaron 220 estudiantes del curso Metodología de la Programación del Primer Año de Ingeniería de Sistemas de la UCSM de Arequipa, en 2023; su edad osciló entre los 16 y 18 años. Un dato relevante es que el porcentaje de desaprobados en el periodo lectivo 2022 fue 15.58% y el porcentaje de inasistentes 7.79%, lo cual hace un total de 23.38% que no aprobaron la asignatura, y son quienes deben mejorar su aprendizaje.

4.2.3. Delimitación de la población

La Universidad Católica de Santa María se encuentra ubicada en la ciudad de Arequipa, Provincia y departamento del mismo nombre. La población objeto de estudio en esta investigación es limitada, dado que se cuenta con el número total de estudiantes inscritos en la asignatura que cursan el primer año de la carrera con estudios profesionales en Ingeniería de Sistemas en la UCSM.

4.2.4. Tipo de muestra

La muestra utilizada para la presente investigación fue probabilística porque se seleccionaron los estudiantes de manera aleatoria cumpliendo la cantidad que sugiere el muestreo, por tratarse de una población finita. En primer lugar, se evaluó una misma guía de práctica a dos grupos de estudiantes: Algunos de ellos podrán utilizar la herramienta *GitHub Copilot* y el resto lo desarrollaron de manera tradicional.

Adicionalmente, se utilizaron como instrumento de medición las encuestas a los alumnos y la entrevista a los catedráticos.

4.2.5. Tamaño de la muestra

Se consideró la cifra 4 como coeficiente de confiabilidad y el porcentaje 94 para el nivel de confianza, la cantidad de muestra redondeada considerada es de 142 estudiantes por el lado de las encuestas y 4 responsables del dictado del curso para las entrevistas de los docentes.

4.2.6. Proceso de selección

El proceso de selección de la muestra estuvo bajo el parámetro del semestre que cursan los estudiantes que fueron el ente de estudio, es decir, alumnos de primer año de Ingeniería de Sistemas en la UCSM.

4.2.7. Los métodos y las técnicas

4.2.7.1. Métodos teóricos

Los métodos teóricos de pesquisas o investigación desempeñan siempre un papel crucial en la generación y validación del conocimiento científico, proporcionando herramientas y enfoques para formular, evaluar e interpretar teorías en diversas disciplinas. (Guba & Lincoln, 1994).

Uno de los enfoques más comunes en la investigación teórica es el método deductivo, que parte de la formulación de hipótesis basadas en teorías existentes para luego contrastarlas con evidencia empírica (Popper, 1959). Este método busca

confirmar o refutar teorías mediante la observación, así como con el análisis de datos, contribuyendo así al avance del conocimiento científico.

Otro enfoque relevante es el método hermenéutico, que se basa en la interpretación y comprensión de textos, discursos para desentrañar su significado (Gadamer, 1975), siendo uno de los métodos más aplicado dentro de disciplinas como la sociología, la antropología y los estudios culturales.

Además, los métodos teóricos también pueden incluir el análisis de contenido, que consiste en examinar y categorizar el contenido de documentos, entrevistas u otros materiales para identificar patrones, temas o tendencias relevantes (Krippendorff, 2004). Este enfoque se usa en la investigación cualitativa y puede proporcionar información valiosa para desarrollar y evaluar teorías en diversas áreas del conocimiento.

Por el método inductivo-deductivo: Hace referencia al procedimiento donde persona elabora una conclusión y justifica su certeza. El método deductivo implica iniciar con premisas generales y luego aplicarlas a una situación concreta para llegar a una conclusión.

4.2.7.2. Técnicas e instrumentos

Las técnicas e instrumentos de investigación son elementos fundamentales para la recopilación y análisis de datos, permiten obtener información válida y confiable para responder a las preguntas de investigación planteadas.

Una de las estrategias más empleadas en la investigación tanto social como científica" es la encuesta, que consiste en la recolección de datos a través de preguntas organizadas y enfocadas en una muestra que refleje adecuadamente a la población" (Babbie, 2016). Las encuestas pueden ser realizadas de manera presencial, telefónica, por correo electrónico o en línea, y son útiles para obtener información sobre actitudes, opiniones y comportamientos de los participantes.

Otro método común es la observación, que incluye la recolección sistemática e imparcial de información mediante la observación directa de sucesos, conductas o fenómenos (Bryman, 2016). Esta técnica es especialmente útil en estudios cualitativos, donde se busca comprender el contexto y la dinámica social de manera detallada.

En cuanto a los instrumentos de investigación, estos pueden incluir cuestionarios, entrevistas estructuradas o semiestructuradas, escalas de medición, pruebas psicométricas, entre otros (Fink, 2019). Cada instrumento se selecciona en función de las variables que se pretenden medir y los objetivos del estudio.

4.2.8. Análisis de la práctica de evaluación

En la guía práctica que utilizamos para el desarrollo de la investigación se ha dividido en tres secciones, cada estudiante fue sometido a la prueba de investigación desarrollaron tres ejercicios, uno de cada tipo de complejidad en un tiempo máximo de 15 minutos (Anexo 1).

Tabla 2: *Tipos de preguntas práctica calificada*

Tipo de Pregunta	Puntuación
Complejidad básica o tradicional	5 puntos
Complejidad intermedia	7 puntos
Complejidad avanzada	8 puntos
Total	20 puntos

Fuente: Elaboración propia

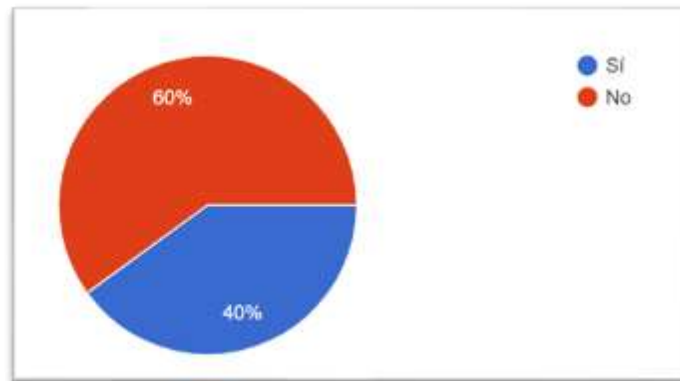
La evaluación de la guía práctica fue presentada a los estudiantes del primer semestre de la Escuela Profesional de Ingeniería de Sistemas, en formato cuestionario, a través de la plataforma Moodle, 45% de la muestra resolvieron los ejercicios sin utilizar el *GitHub Copilot* y el 55% resolvió el cuestionario siguiendo la guía con ayuda de del *GitHub Copilot* , para tener un muestreo representativo y por evaluar el nivel de aprendizaje de la programación.

4.2.9. Análisis y tabulación de la encuesta

Una vez obtenidos los resultados, se presenta los siguientes análisis:

De acuerdo con la primera pregunta formulada: ¿Utilizó antes *GitHub Copilot* como herramienta de apoyo en la programación?, se puede verificar que, en su mayoría, el 60% de los encuestados, no habían utilizado esa herramienta como recurso de apoyo para realizar programas e implementar de manera óptima.

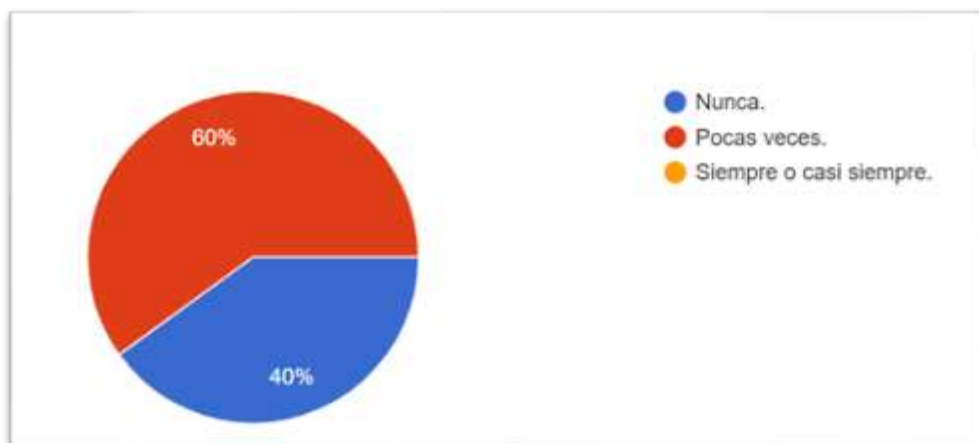
Imagen 13: ¿Utilizó antes GitHub Copilot?



Fuente: Elaboración propia.

En la segunda interrogante, es donde se verifica la frecuencia de uso de la herramienta *GitHub Copilot*, en el desarrollo de sus trabajos, lamentablemente los resultados obtenidos muestran que sólo en un 60% es aplicada esta herramienta para desarrollar trabajos de programación; esto significa que se tiene un trabajo por realizar para la difusión de esta para poder mejorar el proceso de aprendizaje.

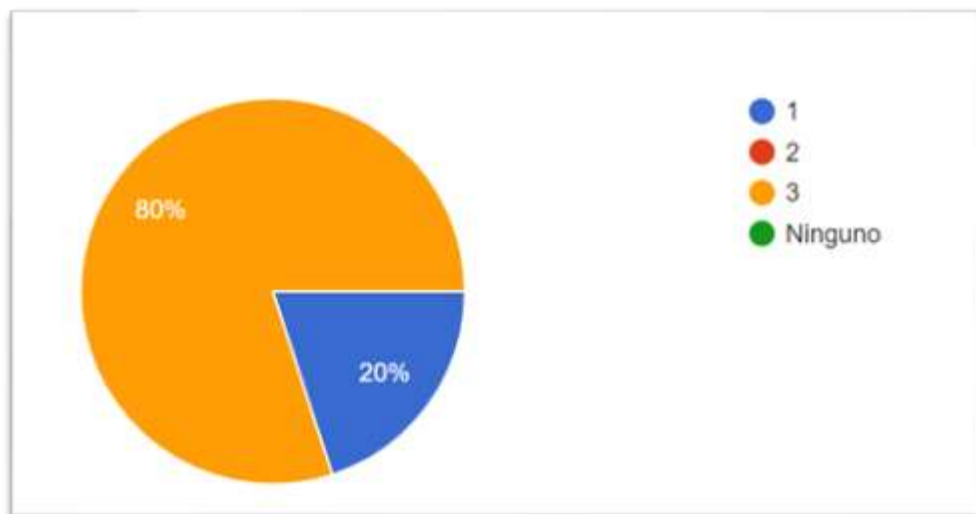
Imagen 14: ¿Con qué frecuencia utiliza GitHub Copilot en el desarrollo de sus trabajos?



Fuente: Elaboración propia

De acuerdo con la tercera pregunta formulada, ¿Cuántos ejercicios resolvió completamente? En este caso se resolvieron los 3 ejercicios en un 80 % de los estudiantes y solo un 20 % presentó dificultades al resolver dos ejercicios, teniendo como herramienta *GitHub Copilot*.

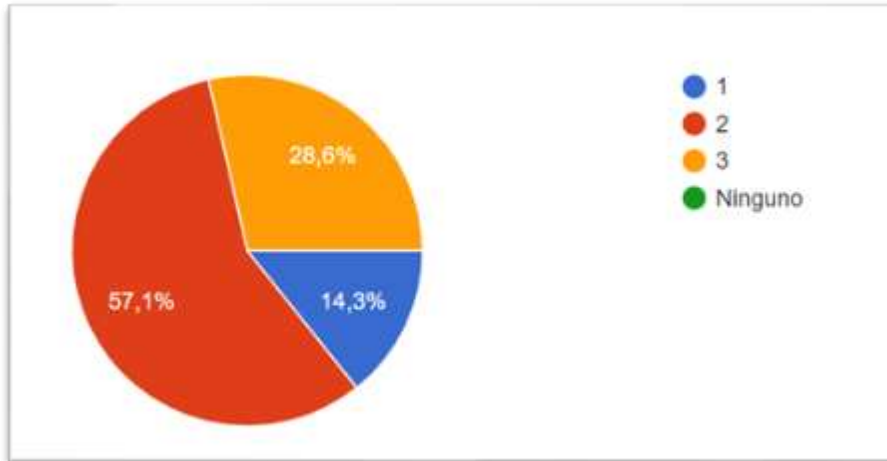
Imagen 15: ¿Cuántos ejercicios resolvió completamente con *GitHub Copilot*?



Fuente: Elaboración propia

En esta misma pregunta efectuada a los estudiantes que no hicieron uso de la herramienta *GitHub Copilot*, se obtuvo una marcada diferencia, ya que solo el 57.1% logró resolver completamente los 3 ejercicios, notando una clara diferencia de 22.9% entre usar y no hacer uso de la herramienta I.A., lo que evidencia el gran apoyo que esta herramienta proporciona, pero siempre debe ser usada con cautela.

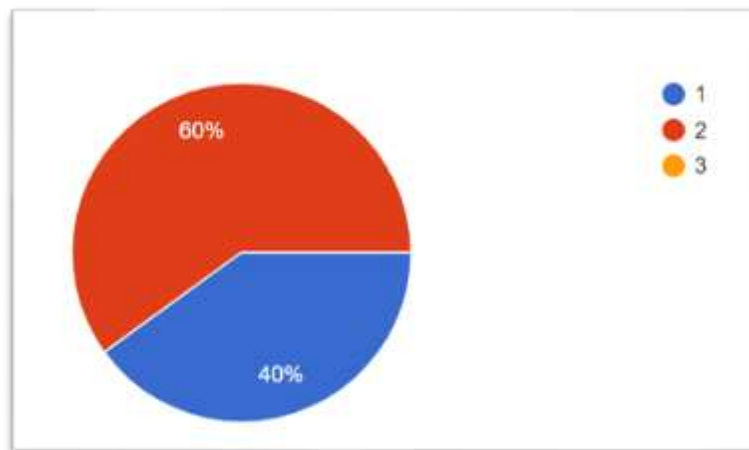
Imagen 16: ¿Cuántos ejercicios resolvió completamente? – Método tradicional



Fuente: Elaboración propia

La cuarta pregunta, se intenta recoger cuál de los ejercicios les pareció más difícil, y podemos observar que es el segundo ejercicio con un 60% el que les presentó mayor exigencia de raciocinio para resolverlo, a pesar de ser de complejidad intermedia. Esta pregunta intenta plantear la aplicación de los bucles anidados, pero con la ayuda de la herramienta de *GitHub Copilot* pudieron comprenderlo rápidamente, ya que se obtuvieron respuestas correctas en la solución de estas por parte de la mayoría de los estudiantes, lo que, efectivamente, demuestra que es una herramienta de ayuda.

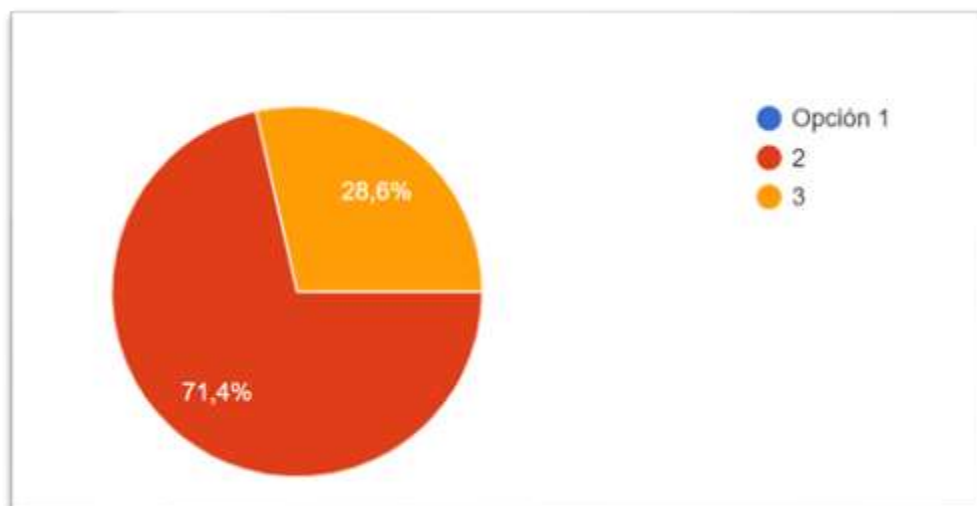
Imagen 17: *¿Cuál de los ejercicios le pareció más dificultoso con GitHub Copilot?*



Fuente: Elaboración propia

Por otra parte, los estudiantes que desarrollaron la práctica de manera tradicional indicaron en un 71.4% la mayor dificultad la encontraron en el ejercicio 2; sin embargo, la solución a dicho ejercicio solo fue hallada por una minoría de la muestra, siendo este un nicho para posteriormente aplicar herramientas que ayuden su aprendizaje.

Imagen 18: *¿Cuál de los ejercicios le pareció más dificultoso? - Método tradicional*

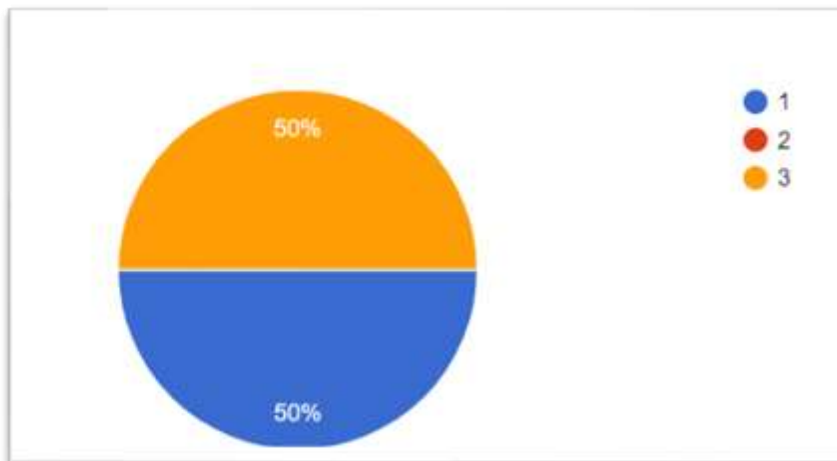


Fuente: Elaboración propia

El objetivo de la quinta pregunta es conocer el ejercicio de menor dificultad, en este caso buscar soluciones simples, clasificadas como complejidad básica y el

completar código son los que tienen menor dificultad, ya que es texto la manera de formularlo y su grado de abstracción es menor que el de la categoría intermedia, esta necesita una expresión gráfica, que debe trasladarse al detalle escrita para conocer el requerimiento.

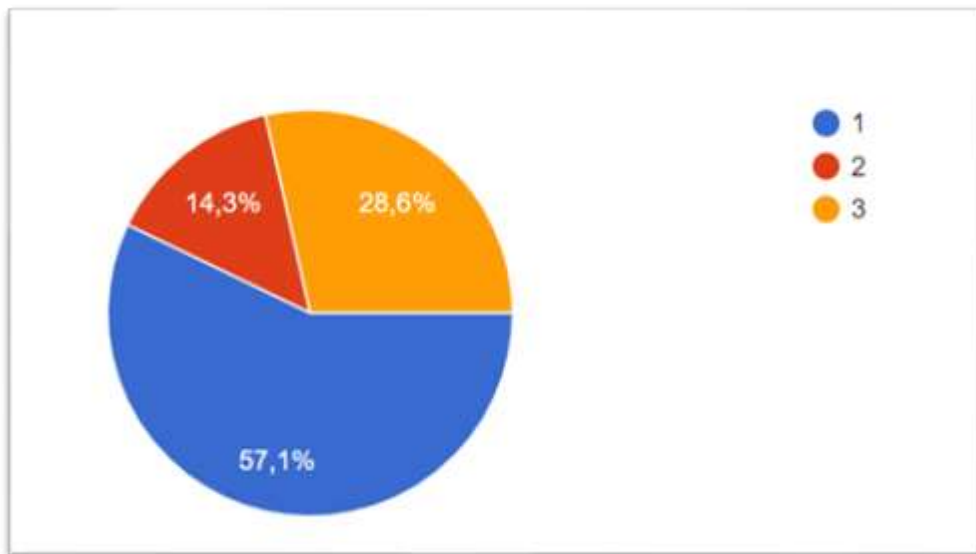
Imagen 19: *¿Cuál de los ejercicios le pareció más sencillo?*



Fuente: Elaboración propia

En la muestra de estudiantes que desarrollaron la práctica de manera tradicional, indicaron en su mayoría, con un 57.1% que el ejercicio 1 es el que se presenta con menor dificultad, se debe mencionar que dicho ejercicio ha sido planteado con complejidad básica y orientado a la forma tradicional de evaluación.

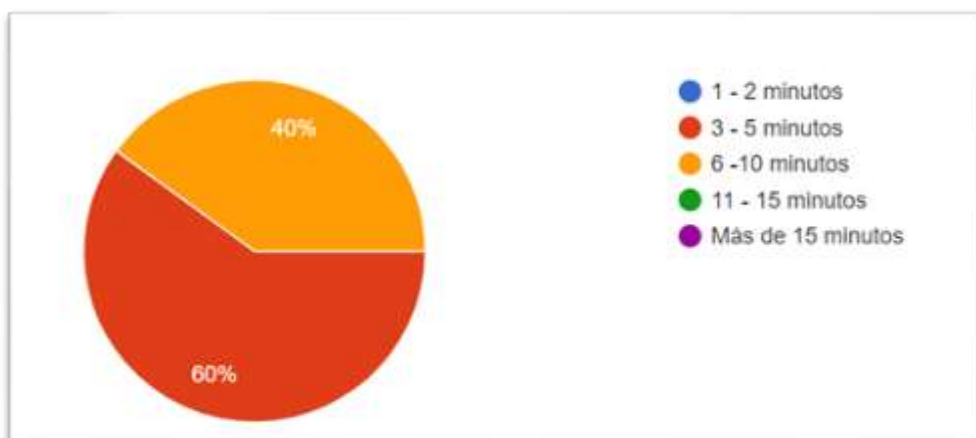
Imagen 20: ¿Cuál de los ejercicios le pareció más sencillo? - Método tradicional



Fuente: Elaboración propia

En la sexta pregunta que indica: ¿Cuánto tiempo aproximadamente en minutos le tomó resolver la práctica? Para la muestra que hace uso de la herramienta *GitHub Copilot* se puede observar claramente que al 60% le tomó entre 3 y 5 minutos y al otro 40% le tomó entre 6 y 10 minutos. Evidenciándose claramente la mejora del uso del tiempo.

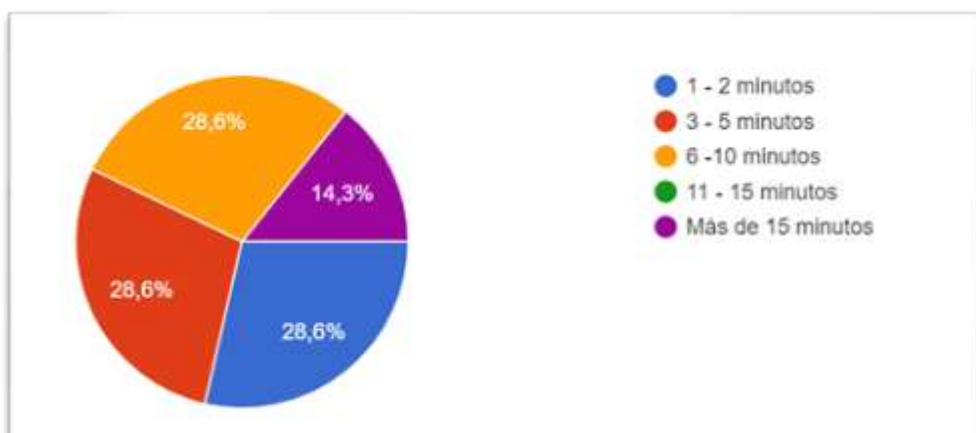
Imagen 21: ¿Cuánto tiempo aproximadamente en minutos le tomó resolver cada uno de los ejercicios?



Fuente: Elaboración propia

En el caso de los evaluados de manera tradicional, el escenario se muestra diferente, es bastante disperso el tiempo promedio consumido en desarrollar, no teniendo una notoriedad significativa alguno de los intervalos de tiempo, con lo cual podemos indicar claramente, que la herramienta *GitHub Copilot* ayuda a mejorar los tiempos de desarrollo de programación.

Imagen 22: *¿Cuánto tiempo aproximadamente en minutos le tomó resolver cada uno de los ejercicios? - Método tradicional*

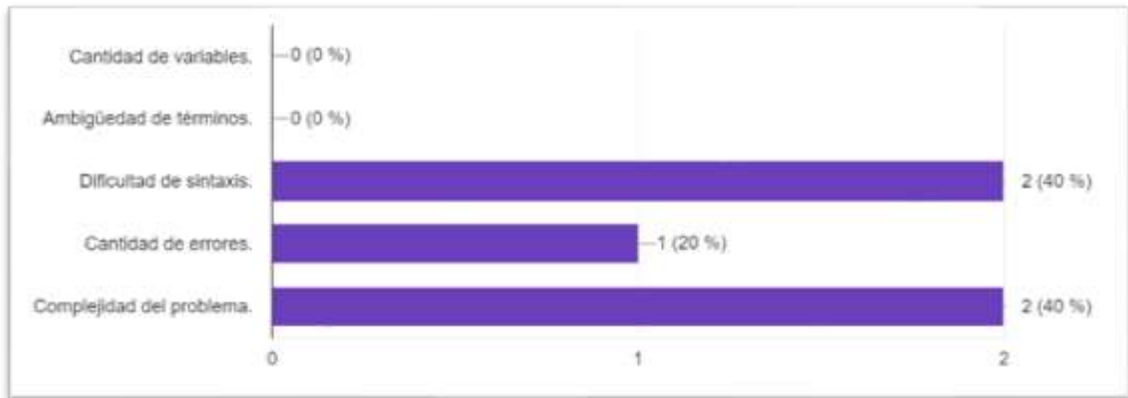


Fuente: Elaboración propia

En la séptima pregunta, se pretende conocer las dificultades que tuvo al plantear las soluciones a los programas propuestos, siendo la complejidad del problema y la dificultad de sintaxis las que tuvieron mayor incidencia, debido al análisis previo que se debe desarrollar para encontrar una solución aceptable y posteriormente óptima al desarrollo de un caso. Como tercer problema se encuentra la dificultad que el estudiante halla en la corrección de errores, los cuales se presentan al momento de

realizar la programación, cabe mencionar que la herramienta de I.A. *GitHub Copilot* también autocompleta código facilitando la corrección de este.

Imagen 23: ¿Cuáles fueron las principales dificultades que tuvo?



Fuente: Elaboración propia

En el caso de los estudiantes que realizaron la práctica de manera tradicional, se nota la diferencia ya que encuentran un problema adicional muy significativo que es la ambigüedad de términos con un 57.1%, el segundo problema es compartido entre la dificultad de sintaxis y la complejidad del problema con un 28.6% y el tercer problema la cantidad de errores con un 14.3%.

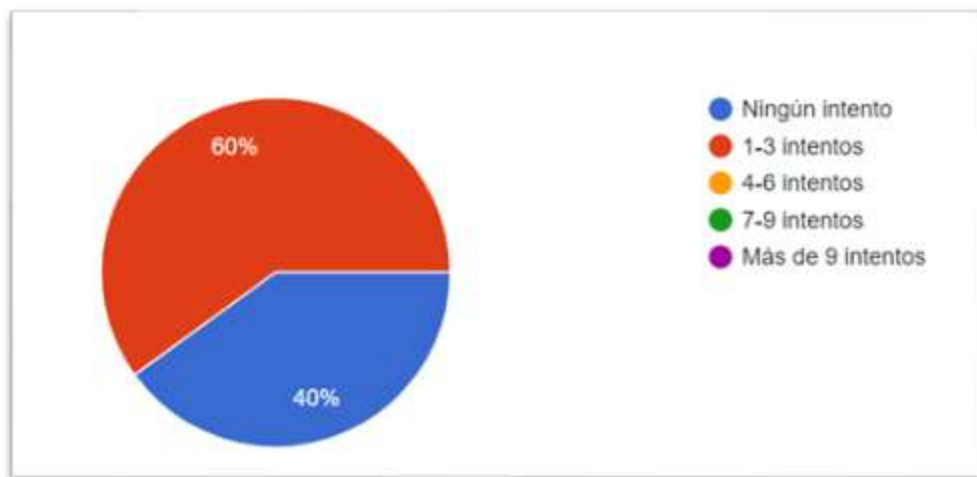
Imagen 24: ¿Cuáles fueron las principales dificultades que tuvo? - Método tradicional



Fuente: Elaboración propia

En la octava pregunta se busca saber cuántos intentos hicieron los estudiantes para lograr la solución óptima de programación. En su mayoría con un 60% requirió entre 1 y 3 intentos previos a la solución óptima y solo el 40% logró tener la solución óptima sin ningún intento previo.

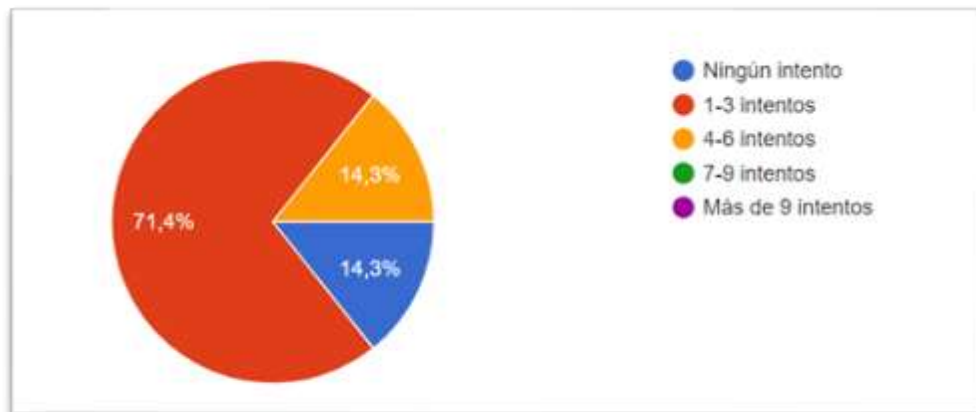
Imagen 25: *¿Cuántos intentos erróneos tuvo antes de generar la solución final con GitHub Copilot?*



Fuente: Elaboración propia

Al realizarlo de forma tradicional se observa que la cantidad de intentos incrementa teniendo un 14.3% de estudiantes que necesitaron entre 4 a 6 intentos; y el número de estudiantes que no requirieron mayor cantidad de intentos disminuye a 14.3%, esto nos lleva a indicar que la herramienta *GitHub Copilot* ayuda en el aprendizaje de la programación, ya que disminuye los intentos para tener la opción óptima de programación.

Imagen 26: ¿Cuántos intentos erróneos tuvo antes de generar la solución final con *GitHub Copilot*? - Método tradicional



Fuente: Elaboración propia

La novena pregunta se plantea para conocer: ¿Cuál es la principal ventaja que encontró el estudiante con la herramienta *GitHub Copilot*? En las respuestas de los encuestados la más destacada se tiene: “Gracias a *Copilot* aceleré el tiempo de trabajo de los ejercicios, ayudando a expandir nuestra opción de soluciones”, lo que evidencia que es de gran ayuda, para plantear nuevas propuestas de soluciones ampliando el rango de respuestas a los problemas de programación planteados.

Por otra parte, a la muestra de estudiantes que desarrollaron la práctica de manera tradicional, se les consultó sobre algunas herramientas de generación de código que conocen, teniendo como respuestas las siguientes: *Copilot*, *Jhipster*, *ChatGPT*, *CodeMaker*, *BlackBox*, *Replit GhostWriter*.

En la décima pregunta tiene por fin conocer las desventajas que presenta la herramienta *GitHub Copilot*, la más destacada es la dificultad de plantearle problemas gráficos, es decir que conlleven a la complejidad intermedia de los ejercicios planteados donde es necesario realizar el uso de anidamiento con bucles y una

descripción textual de la imagen que se quiere proyectar; así mismo el discernir la mejor opción que les puede presentar la herramienta, según se formule el problema o inicie la programación.

En el caso de los estudiantes que desarrollaron la práctica de manera tradicional, se les consultó sobre la principal desventaja al momento de desarrollar código, siendo esta la posibilidad de cometer errores al escribir el código, ya que estos no siempre son fácilmente visibles. Esto implica la necesidad de revisar el código línea por línea para asegurarse de que todo esté correcto, lo cual conlleva a indicar que la herramienta *GitHub Copilot* ayuda en mejorar el tiempo de desarrollo invertido en la programación.

Finalmente, es de interés de los estudiantes conocer herramientas de I.A. que le ayuden a optimizar su programación como la mostrada en la prueba de investigación y otras tales como *intelliCode*.

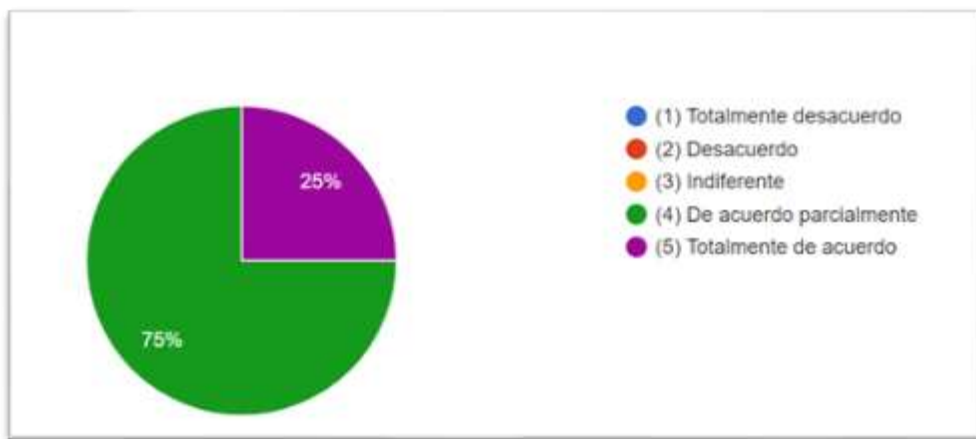
4.2.10. Análisis de entrevistas a docentes del curso de metodología de la programación

Se entrevistó a los docentes responsables del dictado de los cursos de programación del primer semestre de la carrera profesional de Ingeniería de Sistemas, en la UCSM, para conocer su percepción respecto a esta herramienta y ver su grado de pertinencia en el aprendizaje de la programación, lo que se detalla a continuación.

En la primera pregunta respecto a si considera que el uso de herramientas de Inteligencia Artificial es beneficioso en el aprendizaje de sus estudiantes, el 100% de

los catedráticos está a favor, es decir, parcialmente de acuerdo en un 75% y totalmente de acuerdo en un 25%, lo que hace concluir que se considera a las herramientas de I.A. , específicamente *GitHub Copilot*, bastante útiles para desarrollar sus labores académicas y mejorar así el aprendizaje en los estudiantes, si es bien enfocada. Tal como vemos en el grafico siguiente:

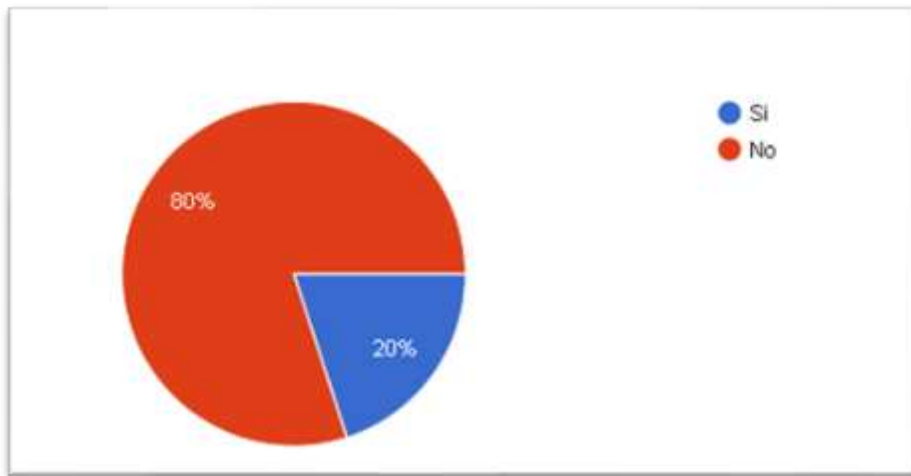
Imagen 27: *¿Considera que el uso de herramientas de Inteligencia Artificial es beneficioso en el aprendizaje de sus estudiantes?*



Fuente: Elaboración propia

En la segunda pregunta, que a la letra dice: “Usted como docente, ¿Ha utilizado la herramienta *GitHub Copilot* para elaborar su material de clase?”, se observa que es un tema aún rico en exploración, con muchas oportunidades de investigación, ya que el 80% no la conoce aún, pero como se ve en posteriores preguntas hay una percepción positiva para el uso de estas herramientas en el aprendizaje de la programación.

Imagen 28: *Usted como docente, ¿Ha utilizado la herramienta GitHub Copilot para elaborar su material de clase?*

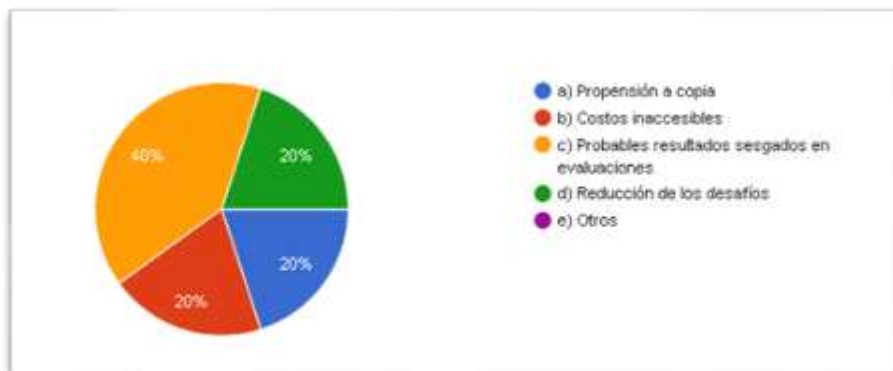


Fuente: Elaboración propia

En la tercera pregunta: ¿Usted integraría esta herramienta en el silabo de su asignatura?, de ser afirmativo, ¿Como sugiere llevar a cabo?, de lo contrario ¿Por qué no la integraría? Aquí, en su mayoría fue respondida que están de acuerdo en integrarla; pero la forma difiere, ya que algunos catedráticos sugieren que se aplique en la última fase del curso luego que el estudiante desarrolle su lógica al inicio, algunos docentes sugieren su uso solo para los proyectos finales; además se plantea que sea una alternativa más de solución y así consolidar una programación óptima al problema planteado.

En la cuarta pregunta, se pretende identificar las dificultades generadas al utilizar herramientas de I.A. En este análisis, los docentes reconocieron que las herramientas de I.A. podrían generar resultados sesgados en sus evaluaciones, un 40% indica esta preocupación, seguido de la propensión a copias, que podría generar, por eso, se recomienda orientar el tipo de preguntas planteadas por los docentes, para que evalúen el raciocinio y no la mera construcción de código fuente.

Imagen 29: *¿Cuál cree que sería la principal dificultad al utilizar las herramientas de Inteligencia Artificial en la educación?*

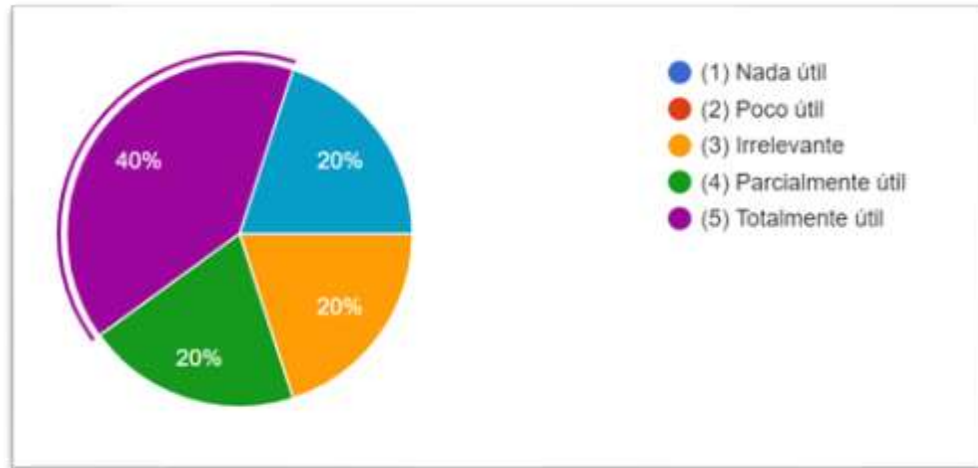


Fuente: Elaboración propia

En este caso vemos que los costos inaccesibles y reducción de desafío podrían ser otras preocupaciones que los docentes tienen en el aprendizaje de sus asignaturas.

En la quinta pregunta se evalúa la utilidad de la herramienta *GitHub Copilot* para enseñar la materia, aquí se destaca que más de la mitad de los catedráticos consideran que puede ser de utilidad, de ellos, un notable 40% indica que es totalmente útil y un 20% piensa que lo es parcialmente; también se puede observar que solo un 40% cree que es poco o nada útil, tal como se ve en la figura siguiente.

Imagen 30: En una escala que va del 1 al 5, donde 1 es Nada útil y 5 es totalmente útil, ¿Cómo respondería usted la siguiente pregunta: ¿Qué tan útil es la herramienta GitHub Copilot para enseñar su materia?



Fuente: Elaboración propia

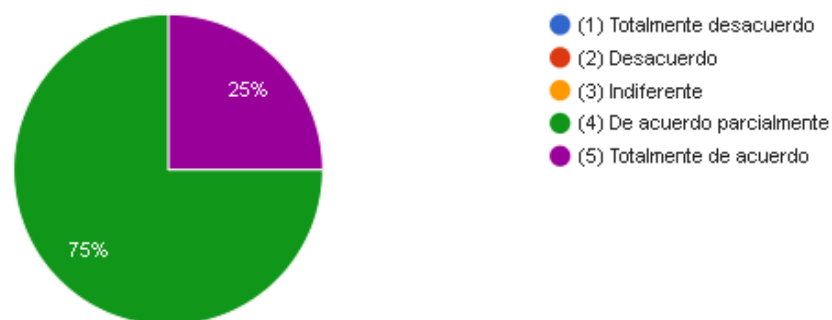
En la sexta pregunta referente a la utilidad de la herramienta en el impacto positivo del rendimiento académico, se indica que la herramienta de I.A. debe ser complementaria a los conocimientos que el estudiante alcance, primero a través de su razonamiento lógico, comprendiendo primero los conceptos teóricos prácticos asociados a la programación, asimismo sugieren que los alumnos deben verla solo como una herramienta alternativa, para analizar errores, optimizar código, y reducir los tiempos (tiempo vs costo), mas no como reemplazo del proceso de implementación de programas.

En la séptima pregunta, se quiere identificar las herramientas que el catedrático propone utilizar para minimizar el potencial de copia, ante ello, la respuesta más destacada fue solicitar a los estudiantes el análisis del código generado, es decir que el estudiante no se limite a ver el código sugerido sino también analizarlo profundamente, por otro lado, otra estrategia sugerida es solicitarle modificaciones en

tiempo real, para evaluar su nivel de comprensión, contribuyendo así al aprendizaje del estudiante.

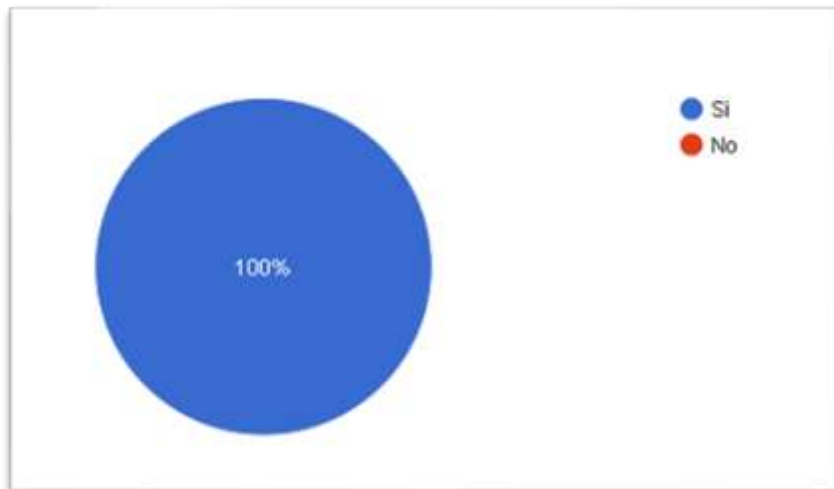
En la octava pregunta donde se le solicita al experto docente calificar en qué medida contribuye la herramienta *GitHub Copilot* al desarrollo de las habilidades de programación en una escala del 1 al 5, donde 1 es no contribuye y 5 contribuye positivamente, fue contundente la respuesta en la que todos coincidieron al 100% que la herramienta contribuye de manera parcial; ello debido a que el estudiante es quien finalmente debe tener la lógica, inclusive de formular la pregunta a la herramienta. Para obtener respuestas adecuadas es necesario que la formulación de la pregunta sea adecuada también.

Imagen 31: *En una escala del 1 al 5, donde 1 es no contribuye y 5 contribuye positivamente. ¿En qué medida considera que la herramienta GitHub Copilot contribuye al desarrollo de habilidades de programación de los estudiantes?*



En la novena pregunta ¿Cree que la herramienta es fácil de usar para los alumnos universitarios con diversos niveles de habilidad en programación?

Imagen 32: Considera usted que ¿La herramienta es fácil de usar para los alumnos que tienen diferentes niveles de habilidad en programación?



Fuente: Elaboración propia

Vemos que todos los catedráticos tienen una apreciación positiva y consideran que será una herramienta fácil de utilizar siendo esto un factor favorable para el aprendizaje de la programación en los estudiantes.

En la décima consulta se preguntó a los expertos docentes sobre la efectividad de la herramienta en el proceso de enseñanza-aprendizaje. Los comentarios recibidos han sido positivos, destacando ser una guía para el desarrollo del código, favoreciendo en el aprendizaje de los estudiantes, generando impacto en la mejora de su lógica de interpretación y optimizando el código de desarrollo.

4.3. Diseño de mejora

Para implementar el diseño de mejora se ha elegido un tema específico llamado bucles en programación, planteado en una guía de laboratorio, que contará con una sección teórica a la introducción al tema y el aporte será dado en los ejercicios propuestos a desarrollar, haciendo uso de la herramienta *GitHub Copilot*, en primer

lugar tenemos los ejercicios que deben realizar figuras gráficas con la aplicación de bucles, el nivel de complejidad de los ejercicios es similar; pero la lógica difiere en cada caso de esta manera se restringe la simple copia literal del enunciado a la herramienta de I.A. ; para ello se requiere un previo raciocinio para formular las especificaciones a la herramienta, esto ayuda al objetivo de mejorar el aprendizaje de los estudiantes:

1. Desarrollar un programa para imprimir la siguiente información:

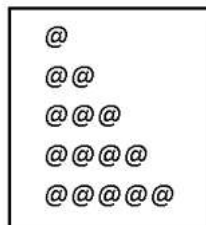
Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

Imagen 33: *Ejercicio propuesto 1*



Fuente: Elaboración propia

2. Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

Imagen 34: Ejercicio propuesto 2



Fuente: Elaboración propia

3. Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

Imagen 35: Ejercicio propuesto 3



Fuente: Elaboración propia

4. Desarrollar un programa para imprimir la siguiente información:

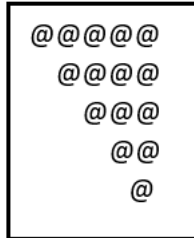
Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

Imagen 36: Ejercicio propuesto 4



Fuente: Elaboración propia

5. Desarrollar un programa para imprimir la siguiente información:

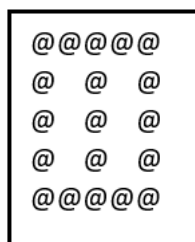
Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

Imagen 37: *Ejercicio propuesto 5*



Fuente: Elaboración propia

En segundo lugar se propuso el desarrollo de ejercicios de programación donde se tenga que completar el código, es decir en este caso la herramienta de *GitHub Copilot* también podrá aportar soluciones, pero sola alguna o algunas de las alternativas de soluciones, lo que implica que el estudiante puede obtener respuestas

parciales pero no en su totalidad, aportando de esa manera a la lógica en su aprendizaje y analizando opciones adicionales a las que le muestra la herramienta, estos ejercicios tienen varias opciones de solución.

6. Responde ¿Cuántas veces se imprime la palabra 'hi' en el siguiente código?

Imagen 38: Ejercicio propuesto 6

```
i=5
while(i>0) :
    print('hi');
    i=i+1;
```

Fuente: Elaboración propia

- a) 0
- b) 1
- c) 2
- d) Infinito
- e) N.A.

7. Responder, ¿Cuántas posibles condiciones simples hacen que se escriba "hi" 3 veces en el siguiente código?

Imagen 39: Ejercicio propuesto 7

```
i=2;
while( ):
    print("hi");
    i=i+2;
```

Fuente: Elaboración propia

- a) 0
- b) 1
- c) 9 a más
- d) 3 a 8
- e) N.A.

8. Responder, ¿cuántas posibles sentencias simples hacen que se escriba hi,3 veces?

Imagen 40: Ejercicio propuesto 8

```
i=2;
while(i<=6):
    print("hi");
    i= ;
```

Fuente: Elaboración propia

- a) 0
- b) 1
- c) 9
- d) 3 a 8
- e) N.A.

9. Complete el código que implemente un programa que permita ingresar y sumar indefinidos números hasta que se ingrese el número 0 y se detenga.

Imagen 41: Ejercicio propuesto 9

```
n = int(input('ingrese un numero'));
s=0;

while( ):
    s=s+n;
    n = int (input ('ingrese un numero' ));

print(s)
```

Fuente: Elaboración propia

Complete: _____

10. Complete el código que implemente un programa que permita ingresar y sumar indefinidos números hasta que se ingrese el número 0 y se detenga.

Imagen 42: Ejercicio propuesto 10

```
n = int(input('ingrese un numero'))
s=0;

while(n!=0):
    s= ;
    n = int(input('ingrese un numero'));

print(s)
```

Fuente: Elaboración propia

Complete: _____

4.4. Definición de módulos de control

Para mejorar el aprendizaje de los estudiantes haciendo uso de la herramienta *GitHub Copilot* es necesario que los ejercicios que se planteen en las guías de laboratorio conlleven al aprendizaje con raciocinio, planteando ejercicios para completar código de programación o mostrando otros que puedan optar por múltiples soluciones.

Una alternativa para mejorar la enseñanza es contar con herramientas especializadas en detección de código similar como lo es *Plagiarism Detection Tools*. Asimismo, para la evaluación práctica se puede utilizar herramientas, anticopia, como *Respondus LockDown Browser*, donde permitan tener únicamente una pantalla activa, con las opciones de desarrollo y el bloqueo de las demás herramientas a excepción lógicamente de *GitHub Copilot*, este sería un instrumento adicional, donde el estudiante puede hacer uso de este, evitando acceso a otros recursos.

5. SUGERENCIAS

5.1. Conclusiones

La herramienta *GitHub Copilot* favorece al aprendizaje de la lógica de la programación, lo cual ha sido demostrado a través de los puntajes obtenidos por los estudiantes evaluados, los cuales fueron superiores a los obtenidos de la manera tradicional en los estudiantes del curso de Metodología de la Programación en la carrera profesional de Ingeniería de Sistemas de la UCSM.

Existe una fuerte correlación entre el rendimiento académico y la utilización de la herramienta *GitHub Copilot*, ya que no solo se logró un código optimizado en menor tiempo y con respuestas adecuadas, las cuales repercutieron notablemente sus calificaciones.

La aplicación de esta herramienta conlleva nuevos beneficios, como mejora del rendimiento, incremento de posibles soluciones, pero genera también dificultades, como propensión a copias y sesgos en sus calificaciones, que podrían ser abordadas en futuras investigaciones.

Los docentes expertos en el dictado de la materia concluyen que es beneficioso el uso de la herramienta como parte de sus estrategias de aprendizaje ya que ayuda a optimizar los códigos de programación y disminuir los tiempos invertidos en el desarrollo de soluciones a los problemas de lógica y programación.

5.2. Recomendaciones

Dado que el curso de Metodología de la Programación se dicta en tres fases en la Escuela Profesional de Ingeniería de Sistemas, se sugiere utilizar la herramienta *GitHub Copilot* en la tercera fase, solo luego que los estudiantes ya hayan aprendido el razonamiento básico en el inicio de la asignatura, para que solo sea una herramienta de complemento al aprendizaje y no haya sesgos en sus evaluaciones.

También se sugiere reorientar la tarea del catedrático, a no solo ser un evaluador de aprendizajes basado en resolución de ejercicios, sino también ser un crítico del código, cuestionando al estudiante con preguntas puntuales y reflexivas, que lleven al estudiante a un nivel más alto de raciocinio.

Uno de los aspectos más importantes de la herramienta *GitHub Copilot* radica en las múltiples soluciones que se pueden obtener a través de ella, dadas las consultas que pueda formular el estudiante, haciendo que él mismo de acuerdo con su dominio y lógica que va madurando durante el desarrollo de su carrera pueda elegir la solución óptima al problema planteado.

Se sugiere implementar esta herramienta como parte de la estrategia de aprendizaje para mejorar resultados en otras asignaturas similares; puesto que en el estudio fue rotundo el incremento de promedio de notas obtenidas con el uso de la herramienta de I.A., siendo superior al desarrollado de manera tradicional.

Se recomienda que los ejercicios planteados en las guías de práctica tengan cierta dificultad, con un grado de raciocinio que permita no ser resueltos de una

manera sencilla aplicando la técnica *copy-paste* a la herramienta de I.A., puesto que ello distorsionaría el objetivo de su uso.

6. BIBLIOGRAFÍA

1. A. S. Al-Sadoon y S. M. Al-Aqeeli. (2022). Using GitHub Copilot to Improve the Learning of Programming. *IEEE Access*.
2. Bisquerra, R. (2009). *Evaluación de la calidad educativa*. Madrid, España: La Muralla.
3. Chadwick, C. B. (1979). La psicología del aprendizaje desde el enfoque constructivista. *Revista Latinoamericana de Psicología*.
4. G. Van Rossum & F.L. Drake. (2011). *Python Tutorial Release 3.2.3*. Python Software Foundation.
5. GitHub. (2023). <https://github.com/features/copilot>. Obtenido de <https://github.com/features/copilot>.
6. Gross, Z., Rutland, S. (2017). Experiential learning in informal educational settings. 63, 1-8. doi:<https://doi.org/10.1007/s11159-017-9625-6>
7. J. A. Pérez-González y M. A. Jiménez-Sánchez. (2022). The Impact of GitHub Copilot on the Learning of Programming: A Mixed-Methods Study. *Computers & Education*.
8. J.A. Rodríguez, M.P. Sánchez & M.J. García. (2018). Rendimiento académico en estudiantes universitarios factores asociados y medidas de intervención. *Revista Complutense de Educación*, 685-704.
9. Kaczynska, M. (1986). *El rendimiento escolar y la inteligencia*. Madrid, España: Espasa Calpe.

10. Knewton. (19 de 12 de 2023). *Getting started with Knewton*. Obtenido de knewton.com: <https://support.knewton.com/s/article/Getting-Started-with-Knewton-Alta-for-Students>
11. Knuth, D. E. (1997). *The art of computer programming* (3ra Edición ed.). Addison-Wesley.
12. M. A. Al-Khafaji y A. S. Al-Sadoon. (2022). The Use of GitHub Copilot for Teaching Programming: A Case Study. *Journal of Educational Technology and Society*.
13. Mathieu, M. J. (2014). *Introducción a la Programación*. Patria.
14. Microsoft. (2023). *Visual Studio Code - Code Editing. Redefined*. Obtenido de www.microsoft.com
15. OpenAI. (20 de diciembre de 2015). *About OpenAI*. Obtenido de openai.com
16. Pérez S. Antonio y Poveda S. Patricia. (2008). *Efectos del aprendizaje cooperativo en la adaptación escolar*. Alicante, España.
17. Pizarro, R. (1985). Rasgos y Actitudes del Profesor Efectivo. En *Tesis para optar el Grado de Magíster en Ciencias de la Educación*. Chile: Pontificia Universidad Católica de Chile.
18. S. Russell & P. Norving. (2018). *Inteligencia Artificial: Un Enfoque Moderno* (3ra Edición ed.). Pearson.
19. S. S. Al-Sharif, A. S. Al-Sadoon y M. A. Al-Khafaji. (2023). GitHub Copilot as a Tool for Supporting Different Learning Styles in Programming Education. *Education and Information Technologies*.
20. Sebastián A. Furtado, Rafael Furtado & Gustavo Machado. (2018). *A comparative study of different approaches to teaching algorithms*. ACM SIGCSE Bulletin.

21. Sebesta, R. W. (2015). *Conceptos de programación: Guía del Estudiante en C++* (9na Edición ed.). Addison-Wesley.
22. Sommerville, L. (2011). *Ingeniería de Software* (9na Edición ed.). Pearson.
23. T. H. Corme, C. E. Rivest & C. Stein. (2019). *Introduction to alghoritms*. MIT Press.
24. Tejedor Tejedor, F. J. (2003). Poder explicativo de algunos determinantes del rendimiento en los estudios universitarios.
25. Thomas H. Cormen, C. Leiserson, R. Rivest & C. Stein. (s.f.). *Introduction to Algorithms* (4ta edición ed.).
26. UCSM. (2023). *Universidad Católica de Santa María*. Obtenido de www.ucsm.edu.pe

7. ANEXOS

Anexo 1: Cuestionario complejidad 1

1. Desarrollar un programa que pida al usuario un número entero positivo y muestre por pantalla todos los números impares desde 1 hasta ese número separados por comas.

Ejemplo:

Entero positivo: 18

Salida por Pantalla:

1, 3, 5, 7, 9, 11, 13, 15, 17.

2. Desarrollar un programa que pida al usuario un número entero positivo y muestre por pantalla la cuenta atrás desde ese número hasta cero separados por comas.

Ejemplo:

Entero positivo: 8

Salida por Pantalla:

8,7,6,5,4,3,2,1,0.

3. Desarrollar un programa que muestre por pantalla la tabla de multiplicar del 1 al 10.

Salida por Pantalla:

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

4. Desarrollar un programa que permita al usuario ingresar dos años y luego imprima todos los años en ese rango, que sean bisiestos. Nota: para que un año sea bisiesto debe ser divisible por 4 y no debe ser divisible por 100, excepto que también sea divisible por 400.

Ejemplo:

Año Inicio: 2000

Año Final: 2024

Salida por Pantalla:

2000, 2004, 2008, 2012, 2016, 2020, 2024

5. Escriba un programa que pida tres números y diga si el tercero está más cerca del primero o del segundo.

Ejemplo:

Primer número:1

Segundo número:7

Tercer número:9

Salida por pantalla:

El tercer número está más cerca del segundo.

Anexo 2: Cuestionario complejidad 2

1. Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

```
@
@@
@@@
@@@@
@@@@@
```

2. Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

```
@@@@@
@@@@
@@@
@@
@
```

Salida por pantalla:

3. Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

```

  @
  @@
 @@@
@@@@
@@@@@
```

4. Desarrollar un programa para imprimir la siguiente información:
Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

```
@@@@@
 @@@@
  @@@
   @@
    @
```

5. Desarrollar un programa para imprimir la siguiente información:
Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

```
@@@@@
@ @ @
@ @ @
@ @ @
@@@@@
```

Anexo 3: Cuestionario complejidad 3

1. Escribir cuantas veces se imprime la palabra 'hi' en el siguiente código:

```
i=5  
while(i>0) :  
print('hi');  
i=i+1;
```

- a) 0
 - b) 1
 - c) 2
 - d) Infinito
 - e) N.A.
2. Responder, ¿cuántas posibles condiciones simples hacen que se escriba "hi" 3 veces en el siguiente código?

```
i=2;  
while( ):  
print("hi");  
i=i+2;
```

- a) 0
 - b) 1
 - c) 9
 - d) 3 o más
 - e) N.A.
3. Responder, ¿cuántas posibles sentencias simples hacen que se escriba hi,3 veces?

```
i=2;  
while(i<=6):  
print("hi");  
i= ;
```

- a) 0
- b) 1
- c) 9
- d) 3 o más
- e) N.A.

4. Complete el código que implemente un programa que permita ingresar y sumar indefinidos números hasta que se ingrese el número 0 y se detenga.

```
n = int(input('ingrese un número'));
s=0;

while( ):
    s=s+n;
    n = int (input ('ingrese un número' ) );

print(s)
```

Complete: _____

5. Complete el código que implemente un programa que permita ingresar y sumar indefinidos números hasta que se ingrese el número 0 y se detenga

```
n = int(input('ingrese un número'))
s=0;

while(n!=0 ):
    s= ;
    n = int(input('ingrese un número'));

print(s)
```

Complete: _____

Anexo 4: Cuestionario aplicado a los estudiantes (una pregunta aleatoria por cada nivel de complejidad)

Pregunta 1

Desarrollar un programa que muestre por pantalla la tabla de multiplicar del 1 al 10.

Salida por Pantalla:

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Pregunta 2

Desarrollar un programa para imprimir la siguiente información:

Ejemplo:

Ingresar un numero n: 5

Ingresar una letra: @

Salida por pantalla:

@	@	@	@	@
@	@	@		
@	@	@		
@	@	@		
@	@	@	@	@

Pregunta 3

Escribir cuantas veces se imprime la palabra 'hi' en el siguiente código:

```
i=5  
while(i>0) :  
    print('hi');  
    i=i+1;
```

- a. 1
- b. Infinito
- c. 0
- d. Ninguna de las anteriores.
- e. 2

Anexo 5: Opinión de los catedráticos

1.- En una escala del 1 al 5, donde 1 es desacuerdo total y 5 es totalmente de acuerdo, cómo calificaría usted la siguiente pregunta, ¿Considera que el uso de herramientas de Inteligencia Artificial es beneficioso en el aprendizaje de sus estudiantes?

- (1) Totalmente desacuerdo
- (2) Desacuerdo
- (3) Indiferente
- (4) De acuerdo parcialmente
- (5) Totalmente de acuerdo

2.- Usted como docente, ¿Ha utilizado la herramienta *GitHub Copilot* para elaborar su material de clase?

- Si
- No

3.- ¿Usted integraría esta herramienta en el silabo de su asignatura? De ser afirmativo, ¿Cómo sugiere llevar a cabo?, de lo contrario ¿Por qué no la integraría?

Texto de respuesta corta

4.- ¿Cuál cree que sería la principal dificultad al utilizar las herramientas de Inteligencia Artificial en la educación ?

- a) Propensión a copia
- b) Costos inaccesibles
- c) Probables resultados sesgados en evaluaciones
- d) Reducción de los desafíos
- e) Otros

5.- En una escala del 1 al 5, donde 1 es nada útil y 5 es totalmente útil, cómo respondería usted la siguiente pregunta, ¿Qué tan útil es la herramienta *GitHub Copilot* para enseñar su materia?

- (1) Nada útil
- (2) Poco útil
- (3) Irrelevante
- (4) Parcialmente útil
- (5) Totalmente útil

6.- ¿Cree usted que la nueva herramienta podría ayudar a mejorar el rendimiento académico de sus estudiantes? ¿Cómo la utilizaría? De ser afirmativo. ¿Cómo la utilizaría?

Texto de respuesta larga

7.- Teniendo en cuenta que la herramienta es una fuente potencial de copia ¿Cómo la utilizaría usted para minimizarlo?

Texto de respuesta larga

8.- En una escala del 1 al 5, donde 1 es no contribuye y 5 contribuye positivamente.

¿En qué medida considera que la herramienta *GitHub Copilot* contribuye al desarrollo de habilidades de programación de los estudiantes?

- (1) No contribuye.
- (2) Contribuye muy poco
- (3) Indiferente
- (4) Contribuye parcialmente
- (5) Contribuye positivamente

9.- Considera usted que ¿La herramienta es fácil de usar para los estudiantes con diferentes niveles de habilidad en programación?

Si

No

10. ¿Cómo evaluaría usted personalmente la efectividad de la herramienta en el proceso de enseñanza? ¿por qué?

Texto de respuesta corta
