**PROCESO DE GESTIÓN DE FORMACIÓN PROFESIONAL INTEGRAL**

**FORMATO GUÍA DE APRENDIZAJE**

**IDENTIFICACIÓN DE LA GUIA DE APRENDIZAJE**

* **Denominación del Programa de Formación:** Tecnólogo En Análisis y Desarrollo de Software(ADSO)
* **Código del Programa de Formación:** 228118
* **Nombre del Proyecto:** Desarrollo de software como recurso para la gestión de información de las organizaciones en el contexto de la innovación, transformación digital y en el marco de políticas y requisitos de la empresa.
* **Fase del Proyecto:** Fase I - Análisis
* **Actividad de Proyecto:**  Estructurar procesos lógicos para la construcción de algoritmos
* **Competencia:** Análisis de la especificación de requisitos del software.
* **Resultados de Aprendizaje:** Desarrollar procesos lógicos a través de la implementación de algoritmos.
* **Duración de la Guía:** 100 horas.

**2. PRESENTACIÓN**

El origen del término 'algoritmo' surgió a mediados del siglo IX cuando el matemático y astrónomo Mohammed Ibn Musa-al-Khwarizmi desarrolló en su obra "Al-yebr-mugabata" fórmulas para posibilitar que con un número limitado de procesos fuese posible resolver ecuaciones de primer y segundo grado.

Mohammed fue el primero en pensar en modo algorítmico explicó que, mediante una especificación clara y concisa de cómo calcular sistemáticamente, se podrían definir algoritmos que fueran usados en dispositivos mecánicos similares a un ábaco en vez de las manos.

Formuló las reglas de las cuatro operaciones aritméticas con varios dígitos. Posteriormente este concepto comenzó a utilizarse en general para designar las secuencias de operaciones que conducen a la solución de cualquier tarea matemática.

Con el tiempo el proceso de búsqueda y formalización de algoritmos dejó de ser tarea sólo de matemáticos y se obtuvieron diferentes tipos de algoritmos. Así surgieron algoritmos para juegos como damas y ajedrez, donde los objetos son figuras y posiciones en los que se requiere seleccionar el próximo paso. En todos los casos los algoritmos no trabajan con objetos del mundo real sino con representaciones, abstracciones de éstos. Por ello para designarlos se utilizan variables, símbolos, codificaciones.

Así es fue como el algoritmo formó parte de todas las soluciones lógicas y cotidianas a los problemas, cálculos matemáticos, instrucciones de procesos administrativos, un manual de usuario. Muchos autores los señalan como listas de instrucciones para resolver un problema abstracto, es decir, que un número finito de pasos convierten los datos de un problema (entrada) en una solución (salida)

En resumen, un algoritmo es cualquier cosa que funcione paso a paso, donde cada paso se pueda describir sin ambigüedad y sin hacer referencia a una computadora en particular, y además tiene un límite fijo en cuanto a la cantidad de datos que se pueden leer/escribir en un solo paso. El algoritmo se representa a través de un pseudocódigo, que significa falso, ya que se asemeja a un lenguaje de programación, pero no lo es ya que un pseudocódigo no es legible desde una aplicación o programa de un equipo de cómputo, a menos que se desarrolle una aplicación que transforme dicho pseudocódigo en lenguaje entendible por la máquina.

**Lenguaje algorítmico**

Cuando se describen algoritmos, tanto en prosa como por medio del uso de pseudocódigo, se hace más fácilmente legible y sirve de puerta de entrada a la programación directa en lenguajes de programación como Java, C#, C, C++ o Pyhton, lo que le permite a los aprendices transferir los conocimientos a otros lenguajes de programación.

El enfoque no se orienta a garantizar la escritura de código sintácticamente correcto, lo que importa es la claridad en las ideas y la selección de las instrucciones apropiadas.

En el programa Técnico en Programación de Software, los algoritmos son la base para que los aprendices puedan desarrollar programas en un lenguaje, según una secuencia lógica y son utilizados para la resolución de problemas cotidianos.

En la presente guía, los aprendices podrán encontrar una serie de actividades propuestas, que le permiten identificar los conceptos básicos que se utilizan en la escritura de las estructuras algorítmicas como:

* **Lógica matemática:** fundamentos, lógica proposicional.
* **Algoritmo:** concepto, tipos, técnicas para la formulación de algoritmos (pseudocódigo, diagramas de flujo), entidades primitivas, jerarquía de operadores, estructuras secuenciales, condicionales, cíclicas, arreglos, funciones, procedimientos, prueba de escritorio.
* **Herramienta para creación y prueba de algoritmos:** Python.

Todas estas estructuras le permitirán al aprendiz, plasmar soluciones bajo un razonamiento lógico a los diferentes planteamientos de la vida cotidiana y que permitirán crear estructuras que podrán traducir posteriormente a un lenguaje de programación.

**3. FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE**

* Descripción de la(s) Actividad(es)
* Ambiente Requerido
* Materiales

Tomando como referencia la planeación pedagógica y las orientaciones para elaborar guías de aprendizaje citado en la guía de desarrollo curricular

**3.1 Actividades de Reflexión inicial**

**Técnica didáctica**: Lectura, discusión, conversatorio

**Descripción**: Frente al concepto de algoritmo se realiza una lluvia de ideas para construir el concepto de manera particular y luego general mediante una socialización grupal.

Al finalizar el instructor precisa y amplia el concepto aplicando casos prácticos de la vida cotidiana, utilizando como parte de los ejemplos la construcción de origami como la serie sucesiva de pasos que conduce a un resultado.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 1 hora.

**Resultado esperado de esta actividad:**

Reflexión sobre la manera en que los aprendices construyen algoritmos desde su vida cotidiana.

**3.2 Actividades de contextualización e identificación de conocimientos necesarios para el aprendizaje**

3.2.1 Explorar varios de los siguientes sitios, que utilizan el juego y herramientas lúdicas para acercar a los aprendices al mundo de los algoritmos:

[https://codecombat.com/](https://www.topcoder.com/)

[https://www.codingame.com/start](https://www.topcoder.com/)

[http://flexboxfroggy.com/#es](https://www.topcoder.com/)

[https://flukeout.github.io/](https://www.topcoder.com/)

[https://checkio.org/](https://www.topcoder.com/)

[http://play.elevatorsaga.com/](https://www.topcoder.com/)

[https://www.codewars.com/](https://www.topcoder.com/)

[https://vim-adventures.com/](https://www.topcoder.com/)

[http://www.dungeonsanddevelopers.com/](https://www.topcoder.com/)

<https://www.topcoder.com/>

<https://www.hackerearth.com/>

<https://coderbyte.com/>

Una vez comprendido el concepto, se realizan construcciones de algoritmos de la vida cotidiana, como la construcción de una receta de cocina, para acercar a los aprendices a los elementos de entrada, proceso salida.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 2 horas

**3.3 Actividades de apropiación del conocimiento**

3.3.1 Identificar los conceptos fundamentales relacionados con algoritmos

El instructor, explica los conceptos asociados a:

* Fundamentos de lógica
* Lógica proposicional
* Algoritmo
* Lenguajes algorítmicos
* Operadores y jerarquia
* Variables
* Tipos de Datos
* Técnicas para la formulación de algoritmos
* (pseudocódigo, diagramas de flujo)
* Prueba de escritorio
* Entidades primitivas
* Etapas del desarrollo de algoritmos (análisis, diseño y codificación)
* Algoritmos con manejo de archivos
* Operadores relacionales y lógicos
* Estructuras de control: Secuenciales, condicionales, iterativas o cíclicas (Case, While, For)

Luego, el instructor y los aprendices, resuelven las actividades prácticas asociadas a cada uno de los conceptos desarrollados apoyadas en la herramienta Python .

**Técnica Didáctica**: Taller.

**Actividad entregable:** Taller de Estructuras secuenciales y Estructuras de control.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 17 horas

3.3.2 Reconocer elementos de lógica matemática y algoritmos con sus diferentes estructuras

El instructor, apoyado en la herramienta Python y en los documentos: Algoritmos, lenguajes y estructuras de control, explica los conceptos:

* **Lógica matemática:** fundamentos, lógica proposicional.
* **Algoritmo:** concepto, tipos, técnicas para la formulación de algoritmos (pseudocódigo, diagramas de flujo), entidades primitivas, jerarquía de operadores, estructuras secuenciales, condicionales, cíclicas, prueba de escritorio.

Luego, el aprendiz, resuelve el taller Condicionales, aplicando los conceptos explicados por el instructor. Este taller esta disponible en Territorium. Use la herramienta Python o un editor en linea para la solución de cada algoritmo.

**Técnica Didáctica:** Taller.

**Actividad entregable:** Taller de Estructuras secuenciales, Estructuras de control en herramientas para creación y prueba de algoritmos.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 20 horas

3.3.3 Identifcar estructuras algorítmicas como arreglos, estructuras y diccionarios en Python

El instructor, apoyado en el documento Algoritmos, estructuras y arreglos, explica los siguientes conceptos:

* Arreglos: Unidimensionales, multidimensionales
* Listas
* Diccionarios

Luego, el aprendiz, resuelve el taller llamado **Funciones y arreglos**, aplicando los conceptos explicados por el instructor. Este taller esta disponible en Territorium. Use Python para la solución de cada algoritmo.

**Técnica Didáctica:** Taller.

**Actividad entregable:** Taller de Estructuras secuenciales, Estructuras de control en herramientas para creación y prueba de algoritmos.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 20 horas

3.3.4 Identifcar estructuras algorítmicas como funciones y procedimientos en Python

El instructor, apoyado en el lenguaje Python, el documento Funciones\_Algoritmos\_Python, explica los siguientes conceptos:

* Funciones: Estructura, Argumentos y parámetros (por posición, por nombre, llamada sin argumentos, parámetros por defecto, argumentos indeterminados, sentencia return, retorno múltiple.
* Procedimientos

Luego, el aprendiz, resuelve el taller llamado **AlgoritmosFunciones**, aplicando los conceptos explicados por el instructor. Este taller esta disponible en Territorium. Use el lenguaje Python o un editor en linea basado en Python para la solución de cada algoritmo.

**Técnica Didáctica:** Taller.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 20 horas

**3.4 Actividades de transferencia del conocimiento**

3.4.1 Solucionar, aplicar y construir estructuras algorítmicas secuenciales, de control, cíclicas, funciones, procedimientos y arreglos con Pyhton.

De manera individual y/o grupal, los aprendices deben desarrollar, entregar y socializar diversos casos de estudio y ejercicios desarrollados con Python, que evidencie:

* Solucionar problemas de lógica proposicional.
* Aplicar estructuras secuenciales en la construcción de algoritmos.
* Aplicar estructuras de control en la construcción de algoritmos.
* Aplicar estructuras cíclicas en la construcción de algoritmos.
* Construir algoritmos con funciones y procedimientos.
* Construir algoritmos con arreglos.
* Construir algoritmos con manejo de archivos.

**Técnica Didáctica:** Caso de estudio y Socialización.

**Actividad entregable:** Taller de Estructuras secuenciales, Estructuras de control en herramientas para creación y prueba de algoritmos.

**Ambiente requerido**: Meet de Gmail, computadores con conexión a internet.

**Materiales**: Computador.

**Tiempo sugerido para la actividad**: 20 horas

**4. ACTIVIDADES DE EVALUACIÓN**

Tome como referencia la técnica e instrumentos de evaluación citados en la guía de Desarrollo Curricular

|  |  |  |
| --- | --- | --- |
| **Evidencias de Aprendizaje** | **Criterios de Evaluación** | **Técnicas e Instrumentos de Evaluación** |
| **Evidencias:**  Taller de Estructuras secuenciales, Estructuras de control.  Taller de Estructuras secuenciales, Estructuras de control en herramientas para creación y prueba de algoritmos.  Taller de Arreglos, Listas y diccionarios en Python.  Taller de Funciones y Procedimientos.  Caso de estudio y socialización de algoritmos. | Soluciona problemas de lógica proposicional incorporando habilidades propias en el oficio como programador.  Crea soluciones a problemas mediante algoritmos que incluyen estructuras secuenciales, condicionales y cíclicas.  Crea funciones y procedimientos en la solución de algoritmos para ordenar y simplificar los códigos.  Manipula arreglos en diferentes dimensiones para dar solución a problemas reales.  Administra la información de los usuarios por medio de archivos, permitiendo el ingreso, modificación y eliminación de los datos. | **Instrumento**: Lista de chequeo y/o Rúbrica  **Técnica**: Taller  **Instrumento**: Lista de chequeo y/o Rúbrica  **Técnica**: Observación |

**5. GLOSARIO DE TÉRMINOS**

**Algoritmo**: Un algoritmo es un conjunto de instrucciones que son tan precisas y tan bien hechas que basta con que una persona sepa leer para ejecutar correctamente una tarea que nunca antes había ejecutado.

**Estructura secuencial**: Conjunto de instrucciones que se ejecutan exactamente en el orden en que han sido escritas: primero se ejecuta la primera instrucción, luego la segunda instrucción, luego la tercera instrucción y por último la última instrucción.

**Ciclo**: Repetición de una o más instrucciones.

**Pseudocódigo**: es la descripción de un algoritmo que asemeja a un Lenguaje de programación, pero con algunas convenciones del lenguaje natural (de ahí que tenga el prefijo pseudo, que significa falso). Tiene varias ventajas con respecto a los diagramas de flujo, entre las que se destaca el poco espacio que se requiere para representar instrucciones complejas. El pseudocódigo no está regido por ningún estándar.

**Lenguaje de Bajo nivel**: como el código binario o los lenguajes ensambladores, que tienen un efecto directo sobre el hardware de la computadora que lo soporta. Además, tienen una escritura sencilla, pero es una ardua tarea el estructurar programas con ellos.

**Lenguaje de Alto nivel**: como los lenguajes estructurados en C, orientados a objetos en Java o Virtual Basic, o de programación científica en Matlab o Fortran. Estos se caracterizan por presentar los algoritmos perceptibles para la cognitiva humana, además de tener una escritura más compleja que los de bajo nivel.

**6. REFERENTES BILBIOGRÁFICOS**

Construya o cite documentos de apoyo para el desarrollo de la guía, según lo establecido en la guía de desarrollo curricular:

Aguilar, Joyanes (2014). “Fundamentos de programación. Algoritmos y Estructura de Datos”, McGrawHill. Capítulo 2.

Monroy, Samperio & Irasema, Theira (2015). Antología “Programación Estructurda”.

Olvaldo, Cairó (2012). Metodología de la programación (algoritmos, diagramas de flujo y programas), Editorial Alfaomega, Segunda edición.

https://www.python.org/downloads/

<https://tutorial.djangogirls.org/es/python_installation/>

https://codingornot.com/07pythonciclosejemplos

https://www.tutorialesprogramacionya.com/pythonya/detalleconcepto.php?punt o=35&codigo=35&inicio=30

https://uniwebsidad.com/libros/algoritmos python/capitulo 9/utilizandodiccionariosenpython

https://jdvelasq.github.io/courses/notebooks/python/205manejodelistastuplasydiccionarios.html

https://j2logo.com/python/tutorial/python while bucle/

https://pythones.net/bucles for while sintaxis ejemplos/

https://entrenamientopython

basico.readthedocs.io/es/latest/leccion4/bucle\_while.html

https://docs.hektorprofe.net/python/controlando el flujo/iteraciones/

Valdivia (2014). Python. Universidad Austral de Chile.

https://www.researchgate.net/publication/328928778\_Introduccion\_a\_la\_Programacion\_con\_Python\_Computacion\_Interactiva\_y\_Aprendizaje\_Significativo

<https://www3.uji.es/~vjimenez/AULASVIRTUALES/PL-0910/python-by-vilar.pdf>

Escobar, G. Lenguajes compilados e interpretados . Recuperado el 06 de febrero de 2020 de https://blog.makeitreal.camp/lenguajes compilados e interpretados/

González, V. Programación estructurada: subprogramas. Recuperado el 06 de febrero de 2020 de http://platea.pntic.mec.es/vgonzale/cyr\_0708/archivos/\_15/Tema\_3.6.htm

Carlospes.com. Codificación en programación Recuperado el 06 de febrero de 2016 de http://www.carlospes.com/minidiccionario/codificacion.php

Alegsa . Codificación . Recuperado el 06 de febrero de http://www.alegsa.com.ar/Dic/codificacion.php

Vila, I. Operadores , constantes , tipos de datos y expresiones Recuperado el 06 de febrero de http://mecanicositsu.blogspot.com/2012/02/unidadiioperadoresvariablesconstantes.html

DDR. Pseudocódigo . Recuperado el 06 de febrero de http://www.discoduroderoer.es/variables y constantes en pseudocodigo/

COVANTEC(2021). Programación en Python. Recuperado de: https://entrenamiento python basico.readthedocs.io

Gandia, C & Mulero, J (2020). La programación con Python.

**7. CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** |
| **Autor (es)** | **Fredy Sadder** | **Instructor** | **Centro de Servicios y Gestión Empresarial** | **Febrero 18 de 2022** |

**8. CONTROL DE CAMBIOS** (diligenciar únicamente si realiza ajustes a la guía)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** | **Razón del Cambio** |
| **Autor (es)** |  |  |  |  |  |