



## Elemento de competencia 1: Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

### Criterios de desempeño

Los criterios de desempeño esperados para el logro de la(s) competencia(s) formulada(s) son:

- Selecciona los tipos de datos y variables a partir de un problema específico para dar una solución óptima.
- Utiliza las sentencias de entrada, de salida y asignación de forma adecuada para dar solución a problemas planteados.
- Realiza la validación o prueba e escritorio de un algoritmo secuencial para comprobar que los resultados arrojados son los esperados.

### Tema 1: Definición, características, partes y tipos de algoritmos

Para poder realizar programas de computación eficientes, eficaces, flexibles, confiables, portables y escalables que permitan entregar a los clientes (personas o empresas) software de alta calidad, es indispensable alcanzar habilidades en el desarrollo de algoritmos, base de la programación en lenguajes como Java, CC++, PHP, .Net, entre otros.

Para iniciar este módulo es necesario tener una base conceptual común en la cual podamos comprender el significado de los términos utilizados en el análisis de problemas y desarrollo de algoritmos. Por ello abordaremos los principales conceptos antes de pasar a dar soluciones a problemas mediante el desarrollo de algoritmos. Pasemos entonces al primer tema de aprendizaje, para obtener una base conceptual común.

**Algoritmo: concepto, tipos y partes.**

Lisandro Peralta define un algoritmo en los siguientes términos: “Un **algoritmo** es una serie de pasos organizados que describe el proceso que se debe seguir para dar solución a un problema específico.” (Peralta, 2002, 1).

Para la Facultad de Ciencias de la Computación de la Benemérita Universidad Autónoma de Puebla (s.f., en línea), todo algoritmo debe ser:

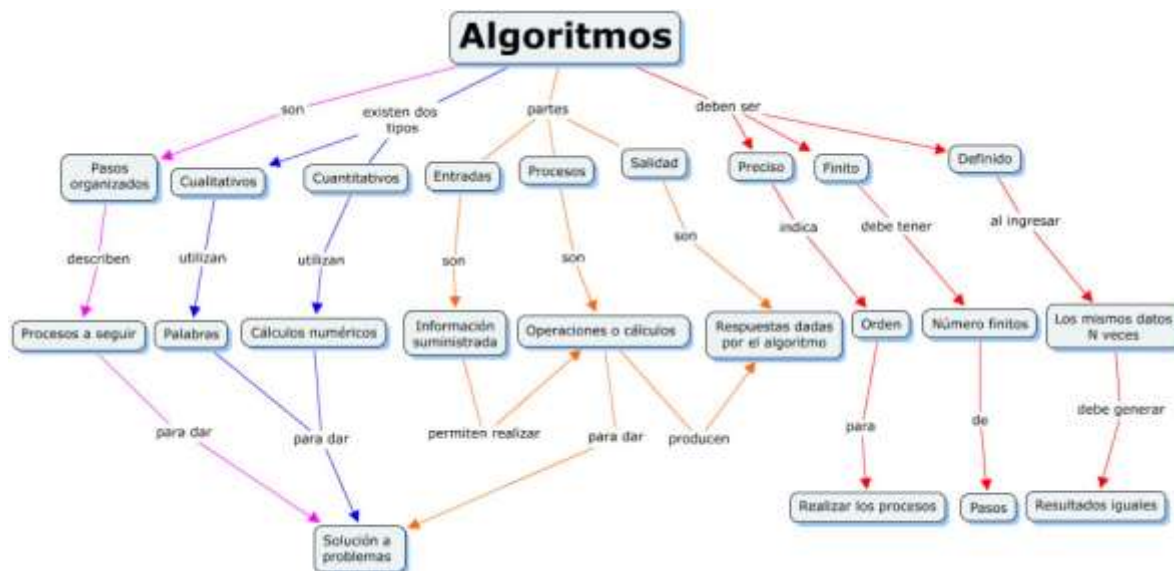
## Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

- **Preciso.** Indicando el orden de realización de cada uno de los pasos.
- **Definido.** Si se sigue el algoritmo varias veces proporcionándoles los mismos datos, se debe obtener siempre los mismos resultados.
- **Finito.** Al seguir el algoritmo, éste debe terminar en algún momento, es decir tener un número finito de pasos.

Además en un algoritmo se debe de considerar tres partes:

1. **Entradas.** Información dada al algoritmo.
2. **Procesos.** Operaciones o cálculos necesarios para encontrar la solución del problema.
3. **Salidas.** Respuestas dadas por el algoritmo o resultados finales de los procesos.

La siguiente figura sintetiza los aspectos fundamentales del concepto de algoritmo:



Para comprender a cabalidad el concepto de algoritmo es importante que observe el siguiente video en el que se explica en qué consiste un algoritmo junto con un ejemplo de cómo freír un huevo siguiendo un proceso algorítmico: [http://youtu.be/YnMMY8Nnj\\_I](http://youtu.be/YnMMY8Nnj_I)

Al terminar el estudio del tema 1 debe tener claro los conceptos de algoritmos, tipos de algoritmos y partes de un algoritmo. Una vez dominadas las anteriores temáticas, abordaremos los conceptos de variables, tipos de datos y expresiones; aprenderemos a identificar de forma correcta las variables y tipos de datos que usaremos en el análisis y el diseño de un algoritmo cuantitativo. Finalmente, incursionaremos en el estudio de los operadores matemáticos, relacionales y lógicos, expresiones y las diferentes formas de representar un algoritmo.



## Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

### Tema 2: Tipos de datos, operadores, variables, constantes y expresiones

Habrás visto que una calculadora sencilla solo recibe números y operadores (suma, resta, multiplicación, división, igual), pero que un computador puede recibir muchos tipos de datos: números, letras, palabras, códigos. La “magia” de estos consiste en que se puede elegir qué información entra, qué operación vamos a realizar y cómo queremos obtener la información luego de un proceso. Uno puede colocar números diferentes cada vez que va a realizar una suma, o colocar palabras diferentes cada vez que va a redactar una carta. Si no pudiéramos tener esta posibilidad de ingresar información diferente cada vez, una calculadora o un computador no servirían de mucho. Para poder tener esta flexibilidad, necesitamos tener algo que pueda recibir múltiples valores, y esto es una **variable**.

Diremos pues que una **variable** es un **espacio** de memoria reservado para almacenar **un valor** que corresponde a un tipo de dato. Es representada y usada a través de una etiqueta (un nombre) que le asigna un programador.

Para dar soluciones a problemas planteados mediante el diseño de algoritmos cuantitativos es preciso identificar y declarar variables con las cuales operará el algoritmo y establecer el tipo de datos de cada una de estas. En este tema se presenta una descripción de los tipos de datos, expresiones, variables, operadores matemáticos, operadores relacionales, operadores lógicos para conformar expresiones. Pasemos entonces al desarrollo de los temas para conocer la forma de identificar y declarar variables y sus tipos de datos, así como de crear expresiones que son utilizadas en el diseño de los algoritmos.

A continuación observaremos un video que nos permitirá aclarar los conceptos acerca de variables y constantes:

<http://youtu.be/-ayE4egxRQ4>

En el siguiente video se aborda el tema de los tipos de datos:

[http://youtu.be/5CBoxm\\_L38Y](http://youtu.be/5CBoxm_L38Y)

Ahora que ya tenemos claros los conceptos básicos para desarrollar algoritmos cuantitativos, pasemos entonces al último tema de la unidad donde veremos las sentencias de lectura, escritura y asignación. Diseñaremos los primeros algoritmos en pseudocódigo y realizaremos la validación o prueba de escritorio de estos para verificar si arroja los resultados esperados y comprobar si el desarrollo dado es correcto de no ser así, se deben hacer los ajustes y correcciones necesarias.





# Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

## Tema 3: Instrucciones de lectura, salida y asignación

Al momento de desarrollar un algoritmo, se deben **identificar los datos** de entrada que generalmente son ingresados por el usuario desde el teclado u otro dispositivo de entrada mediante la sentencia de lectura. Igualmente debemos **identificar los procesos** (fórmulas o cálculos matemáticos) que se requieren para dar solución al problema planteado y finalmente **visualizar o mostrar los resultados** arrojados con la sentencias de salida.


Una vez que se ha terminado de escribir un algoritmo es necesario comprobar que realiza las tareas para las que se ha desarrollado y produce el resultado correcto y esperado, mediante la **validación o prueba de escritorio**, que consiste en tomar datos específicos como entrada y seguir la secuencia indicada en el algoritmo hasta obtener un resultado. El análisis de estos resultados indicará si el algoritmo está correcto o si por el contrario hay necesidad de corregirlo o hacerle ajustes.

En el primer video que observa abajo se explican algunas definiciones de de algoritmos, las partes de un algoritmo (entradas, procesos y salidas); se expone de forma general los tipos de sentencias básicas utilizadas en el desarrollo de algoritmos en pseudocódigo soportados por los lenguajes de programación y se presenta un ejemplo de una lista de canciones. En el segundo video se explica la forma de verificar si el diseño del algoritmo es correcto o no (**prueba de escritorio**):

<http://youtu.be/6oWJ4eBKEU0>

<http://youtu.be/TvHqEhmXE4A>

El estudiante está invitado a profundizar en esta temática, para lo cual se sugiere estudiar las fuentes de estudio siguientes:

Fuentes de estudio	Formato
<b>Unidad I.</b> ( <a href="#">Clic aquí</a> ): En este enlace se debe descargar el documento Unidad I.pdf. Leer con mucha atención, analizar e interpretar los conceptos teóricos que dan la fundamentación para el desarrollo de algoritmos cualitativos y cuantitativos de forma exitosa, así como los ejemplos desarrollados, cada uno de ellos incluye el análisis del problema y el desarrollo del algoritmo en pseudocódigo. Al finalizar el estudio de este documento junto con las explicaciones e insumos que se han compartido previamente, estará en capacidad de identificar qué es un algoritmo, sus partes y tipos, las fases del	



# Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

proceso de creación de un programa. Así mismo, podrá realizar el análisis a un problema específico, diseñar el algoritmo cualitativo en pseudocódigo, identificar los tipos de datos, variables, operadores matemáticos, operadores relacionales, operadores lógicos, crear expresiones, conocer las diferentes formas de representar un algoritmo y las sentencias de lectura, salida y asignación, analizar e interpretar el análisis de los múltiples problemas planteado, el diseño de los algoritmos y la validación o prueba de escritorio de estos.

**Prueba de escritorio** ([Clic aquí](#)): En este enlace se debe descargar el archivo "Prueba de escritorio.ppt" en el que se muestra paso a paso la forma de hacerle seguimiento, validación o prueba de escritorio a un algoritmo que lee dos valores A y B y calcula el producto por sucesión de sumas. La presentación se abre con Microsoft PowerPoint. Seleccionar la opción presentación con diapositivas. Allí observará paso a paso el proceso, disponible en: [Descarga alterna de la presentación](#)

**Ejemplos\_U1** ([Clic aquí](#)): En este enlace debe descargar los ejemplos desarrollados en el documento Unidad I.PDF, realizados en el interpretador de pseudocódigo PSeInt. Estos ejemplos están comprimidos en WinZip por lo que debe descomprimirlos, abrirlos con el PSeInt, analizarlos y ejecutarlos para que verifique su funcionalidad. También los puede ejecutar paso a paso haciendo clic en el menú *Ejecutar - Ejecutar Paso a Paso* o pulsar la tecla de función *F5*, para que siga la secuencia de ejecución. Visitar la página <http://pseint.sourceforge.net/ejemplos.php> para descargar la última versión del interpretador de pseudocódigo, ejemplos desarrollados, documentación y una serie de información que te serán de gran ayuda. [Descarga alterna](#)

Una vez estudiados los anteriores materiales, continuaremos el desarrollo temático con la observación de un video para aprender a instalar el interpretador de pseudocódigo PSeInt.

<http://youtu.be/CRaIQPiwP5w>

A continuación observe otro video tutorial para aprender a instalar el interpretador de pseudocódigo PSeInt disponible en:




## Elemento de competencia 1

Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

[http://www.dailymotion.com/video/x76khr\\_videotutorial-pseint-1-instalacion\\_tech](http://www.dailymotion.com/video/x76khr_videotutorial-pseint-1-instalacion_tech)

Para aprender la sintaxis usada para escribir algoritmos en el interpretador de pseudocódigo PSeInt, debe observar:

[http://www.dailymotion.com/video/x77ss5\\_videotutorial-pseint-2-sintaxis\\_tech](http://www.dailymotion.com/video/x77ss5_videotutorial-pseint-2-sintaxis_tech)

Fuentes de estudio	Formato
<p>Los siguiente tutoriales en línea, <i>Aprenda a programar</i> (<a href="#">Clic aquí</a>), y Tutorial de diseño estructurado de algoritmos (<a href="#">Clic aquí</a>): Tutoriales en línea donde se abordan conceptos básicos de la lógica computacional, algoritmos, diagramas de flujo, pseudocódigo, entre otros. También puedes descargar el curso Aprende a programar en PDF (<a href="#">Clic aquí</a>).</p> <p>Del tutorial <i>Aprende a programar</i> debes estudiar del capítulo 1 al 9, del tutorial <i>diseño estructurado de algoritmos</i> debes estudiar desde la unida 1 a la Unidad 6, y de esta última sólo las estructuras secuenciales, ya que las estructuras condicionales las desarrollaremos en la siguiente unidad.</p>	

En este momento, usted está en capacidad de realizar análisis a problemas, desarrollar algoritmos en pseudocódigo con sentencias secuenciales (lecturas, asignación y salidas) así como realizar la prueba de escritorio para verificar si el algoritmo arroja los resultados esperados y de no ser así, realizar los ajustes y correcciones hasta que cumpla con los requerimientos establecidos en el enunciado del problema. El desarrollo de la siguiente actividad le permitirá afianzar las competencias adquiridas hasta el momento. Una vez entregada la evidencia, se le invita a continuar con el estudio de la unidad 2, en la cual aprenderá a desarrollar algoritmos en pseudocódigo utilizando instrucciones condicionales y ciclos.

### Guía de evidencias y actividades

<b>Competencia Global</b>	Realizar análisis a problemas y generar una solución a través del desarrollo de algoritmos en pseudocódigo aplicando el concepto de programación secuencial o modular de acuerdo a la complejidad del problema y hacer la validación o prueba de escritorio para verificar que el algoritmo arroja los resultados esperados.
---------------------------	--



## Elemento de competencia 1

Construir algoritmos sencillos y secuenciales utilizando estructuras de entrada, de salida y asignación para dar solución a problemas específicos.

<b>Criterios de Desempeño</b>	<ul style="list-style-type: none"><li>• Selecciona los tipos de datos y variables a partir de un problema específico para dar una solución óptima.</li><li>• Utiliza las sentencias de entrada, de salida y asignación de forma adecuada para dar solución a problemas planteados.</li><li>• Realiza la validación o prueba e escritorio de un algoritmo secuencial para comprobar que los resultados arrojados son los esperados.</li></ul>
<b>Las tareas, trabajos, prácticas y demás actividades que debe de realizar el estudiante para el logro de las competencias están relacionadas en la plataforma con sus respectivas instrucciones, criterios de valoración, fechas y tiempo de realización. Cualquier duda sobre las mismas la debe de consultar con su facilitador.</b>	

## Bibliografía

Efraín, O. (2000). Algoritmos estructurados. Colombia, Medellín: Fondo Editorial Cooperativo U de A.

Luis, J. A. (1996). Fundamentos de programación (Algoritmos y Estructuras de datos). (2a Ed.). España: McGraw – Hill

Gabriel, V. (1990). Lógica para Programación de Computadores. Colombia, Medellín: Ediciones Graficas.

Gregorio, M. Q., Francisco T. L. & Vicente, C. L. (2002). Fundamentos de Informática y Programación. [On line]. Disponible en internet: [\[Ver contenido\]](#)

Instituto Tecnológico de La Paz. (1999). Tutorial de diseño estructurado de algoritmos. . [On line]. Disponible en internet: [\[Ver contenido\]](#) (consulta, agosto 31 de 2011).

MailxMail.com. (s.f.). Curso Aprende a programar. [On line]. Disponible en internet: [\[Ver contenido\]](#) (consulta, agosto 31 de 2011).