Informatorio Chaco I 2024 I Desarrollo web

Comenzamos a explorar este nuevo mundo con

Lógica de programación y algoritmos

SEMANA 1

La base para ser un buen programador - 2024

Aprenderás a pensar de forma lógica para generar soluciones concisas y eficientes.

> Conceptos básicos

SEMANA 0

Lógica de programación

Ya transcurrió una semana desde que empezaste con el Informatorio y diste los primeros pasos en el mundo de la programación. Hasta el momento viste conceptos fundamentales que a lo largo de este material complementario vamos a profundizar.

Entre los primeros conceptos encontramos a la lógica de programación y algoritmos, por lo que no hay mejor forma de comenzar este apunte profundizando en estos temas.

Para entender la lógica de programación, lo primero que vamos a hacer es definirla y lo vamos a hacer de la forma más fácil posible.

Definición

La lógica de programación es la base del pensamiento computacional. Es la capacidad de organizar ideas y pasos de manera clara y concisa para que una computadora pueda entenderlas y ejecutarlas.

¿Cuál es el propósito?

Permite resolver problemas de manera sistemática y eficiente utilizando lenguajes de programación.

¿Y cuáles son sus elementos claves?

- **Variables**: Representan entidades que almacenan información.
- **Operadores**: Realizan operaciones aritméticas, lógicas y relacionales sobre variables.
- **Expresiones**: Combinan variables y operadores para obtener resultados.
- **Sentencias de control**: Determinan el flujo de ejecución del programa (if-else, while, for, etc.).
- **Estructuras de datos**: Organizan y almacenan conjuntos de datos (arreglos, listas, diccionarios, etc.).

¿Existen diferentes tipos de lógica?

Si, de hecho, existen varios tipos y vamos a nombrarlos y explicarlos brevemente.

- **Lógica proposicional**: Trabaja con valores de verdad (verdadero o falso) para representar proposiciones y relaciones entre ellas.

- **Lógica de predicados**: Amplía la lógica proposicional permitiendo el uso de cuantificadores (para todo, existe algún) y variables.

- **Lógica difusa**: Maneja conceptos imprecisos o probabilísticos, utilizando valores de verdad difusos (por ejemplo, 0.75 verdadero).

¿Cómo manejo el enfoque correcto para resolver problemas?

- **Análisis**: Descomponer el problema en partes más pequeñas y manejables.

- **Diseño**: Plantear un algoritmo que describa la solución paso a paso (algoritmo, pseudocódigo o diagramas de flujo).

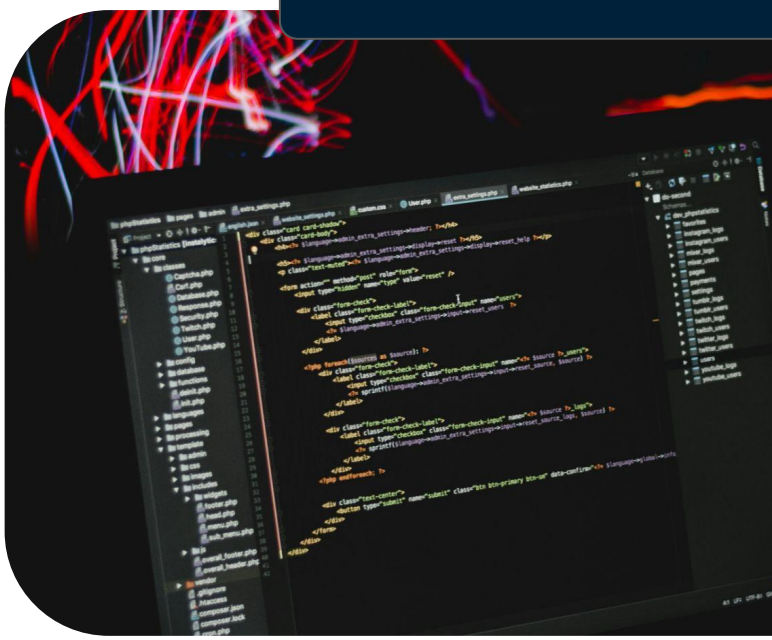
- **Implementación**: Codificar el algoritmo en un lenguaje de programación específico

- **Prueba y depuración**: Ejecutar el programa, identificar y corregir errores.

> Conceptos básicos

SEMANA 0

Algoritmo



Hablamos de algoritmos para resolver problemas o buscar soluciones, pero...

¿Qué es un algoritmo?

Definición

Un algoritmo es una secuencia ordenada de pasos finitos y precisos que describe cómo resolver un problema en particular.

Características

- **Claridad**: Debe ser comprensible y fácil de seguir.
- **Precisión**: Cada paso debe estar definido de forma exacta y sin ambigüedades.
- **Generalidad**: Debe ser aplicable a un conjunto de casos o entradas similares.
- **Eficiencia**: Debe utilizar los recursos computacionales de manera óptima.
- **Corrección**: Debe proporcionar la solución correcta al problema planteado.

Tipos de algoritmos

- **Algoritmos lineales**: Ejecutan un número fijo de pasos en función del tamaño de la entrada.
- **Algoritmos logarítmicos**: El tiempo de ejecución crece de forma logarítmica con el tamaño de la entrada.
- **Algoritmos polinómicos**: El tiempo de ejecución crece de forma polinómica con el tamaño de la entrada.
- **Algoritmos exponenciales**: El tiempo de ejecución crece de forma exponencial con el tamaño de la entrada.

Representación de algoritmos

- **Diagramas de flujo**: Utilizan símbolos gráficos para representar los pasos y la secuencia del algoritmo.
- **Pseudocódigo**: Emplea lenguaje natural con estructuras de programación para describir los pasos del algoritmo.
- **Lenguajes de programación**: Se codifican los pasos del algoritmo en un lenguaje de programación específico.

Técnicas de análisis de algoritmos

• **Análisis de complejidad**: Evalúa el tiempo y los recursos computacionales que requiere el algoritmo en función del tamaño de la entrada.

• **Notación Big O**: Se utiliza para expresar el comportamiento del algoritmo en términos de su eficiencia a medida que crece el tamaño de la entrada.

Si necesitas profundizar más en la lógica de programación y algoritmos te recomendamos los siguientes libros:

• "Introducción a la programación lógica y diseño" de Joyce. y Farrell

• "Algoritmos: Diseño y análisis" de Donald E. Knuth

> Conceptos básicos

SEMANA 0

Primeros pasos con PSeint y Pseudocódigo

Esta herramienta grandiosa herramienta, nos facilita ejecutar pseudocódigo, por lo que es fundamental usarla cuando estás empezando en el mundo de la programación.

Con solo seguir la estructura básica del pseudocódigo se pueden lograr pruebas ejecutables que te van a servir para probar tus algoritmos resueltos en pseudocódigo.

Estructura base

La estructura básica del pseudocódigo implica:

- Nombre del algoritmo luego de la palabra inicial “Algoritmo” y al finalizar el algoritmo la palabra “FinAlgoritmo”.

```
1 Algoritmo nombre_del_algoritmo
2
3 FinAlgoritmo
```

- En la siguiente línea del pseudocódigo se definirán las variables que se utilizarán en el algoritmo, especificando el tipo de dato que se utilizará.

```
3     Definir variable_1, variable_2, aux Como Entero
4
```

- Luego se asignarán o se recolectarán los datos y se los guardará en la o las variables correspondientes.

```
6     Imprimir  'Ingreso de datos.'
7     Leer variable_1
8
```

- Una vez que tenemos datos para trabajar, es momento de comenzar con la parte lógica, donde se realizarán las operaciones que se requieran para dar solución al problema.

```
13      aux ← variable_1 + variable_2
14
```

- El último paso es la muestra de resultados.

```
16     Imprimir 'Resultado: ', aux
17
18
19 FinAlgoritmo
```

La facilidad que nos brinda PSeint es que sin haber escrito código en un lenguaje de programación, ya podemos lograr nuestro algoritmo presentado en pseudocódigo. Esto es realmente útil ya que luego de haber resuelto el problema con nuestro algoritmo, y habiendo realizado pruebas, podemos elegir el lenguaje de programación que más convenga para la solución al problema planteado.

PSeint cuenta con varias herramientas como la de “ejecución paso a paso”, donde podrás ir viendo como trabaja el algoritmo paso por paso.

También es muy útil para presentar el algoritmo mediante diagramas de flujo una vez realizado el pseudocódigo, o viceversa (realizar un diagrama de flujo y presentar el pseudocódigo desde el diagrama).

Durante las mentorías, tus mentores te estuvieron mostrando distintos ejercicios en pseudocódigo mediante PSeint, además de haberte dado un recorrido por el programa, por lo que en este material complementario vamos a seguir centrándonos en la profundización de conceptos fundamentales en la programación y, cómo nuestro lenguaje de programación a partir de esta semana será Python, vamos a seguir dándote explicaciones y conceptos, mediante Python.

En las siguientes páginas te hablaremos de Python, para que conozcas mejor el lenguaje de programación que vas a usar hasta el final del curso.

Luego seguiremos abordando los conceptos que utilizarás durante toda tu carrera como programador.

¡Ahora comienza lo mejor!