



Actividad 1 Algoritmos

Materia: Introducción al Desarrollo de
Software

Ingeniería en Desarrollo de Software



TUTOR: Sandra Lara Devora

ALUMNO: Luis David Ruiz Villanueva

FECHA: 05 de agosto de 2025

Índice

Contenido

Introducción:	1
Descripción:	3
Justificación:	5
Desarrollo:	7
1. Números primos:	7
2. Número par e impar:	9
3. Números invertidos	11
Conclusión:	13

Introducción:

Es un conjunto finito de pasos o instrucciones ordenadas que se utilizan para resolver un problema o realizar una tarea específica.

Es una secuencia lógica de operaciones que, al ejecutarse, produce un resultado determinado. En esencia, un algoritmo es una receta o procedimiento para lograr un objetivo.

Los algoritmos son herramientas fundamentales para resolver problemas y realizar tareas de manera sistemática y eficiente, tanto en el ámbito computacional como en la vida diaria.

Conceptos clave sobre los algoritmos:**Secuencia:**

Los algoritmos deben seguir un orden específico para garantizar que la tarea se complete correctamente.

Finitud:

Un algoritmo debe tener un número limitado de pasos, asegurando que eventualmente termine.

Definición:

Cada paso del algoritmo debe ser claro y preciso, sin ambigüedades.

Entrada y salida:

Los algoritmos toman datos de entrada (input) y producen datos de salida (output).

Aplicaciones:

Los algoritmos se utilizan en una amplia gama de campos, desde la informática y la programación hasta la vida cotidiana.

Descripción:

Es un conjunto de instrucciones o pasos definidos y ordenados que se utilizan para resolver un problema, realizar una tarea o lograr un objetivo específico.

En esencia, es una receta para la computación.

Los algoritmos pueden ser tan simples como una receta de cocina o tan complejos como los que impulsan motores de búsqueda o sistemas de inteligencia artificial.

Características principales de un algoritmo:

- . **Definido:** Cada paso debe ser claro y preciso, sin ambigüedades.
- . **Finito:** Debe tener un principio y un fin, es decir, un número limitado de pasos.
- . **Ordenado:** Los pasos deben seguir una secuencia lógica y específica.
- . **Preciso:** La ejecución de cada paso debe producir resultados predecibles.

Importancia de los algoritmos:

- . **Base de la computación:** Permiten a las computadoras realizar tareas específicas.
- . **Automatización:** Facilitan la ejecución de tareas repetitivas y complejas.
- . **Resolución de problemas:** Brindan soluciones a problemas de diversas áreas.
- . **Desarrollo de software:** Son la base para la creación de programas y aplicaciones.

Son herramientas fundamentales en la informática y en muchos aspectos de nuestra vida diaria, permitiendo la automatización de procesos, la resolución de problemas y la creación de nuevas tecnologías.

Justificación:

Es el proceso de demostrar que el algoritmo resuelve el problema para el que fue diseñado de manera correcta y eficiente.

¿Por qué es importante la justificación de un algoritmo?**Garantiza la fiabilidad:**

Al justificar un algoritmo, se asegura que este funcione correctamente y produzca resultados precisos, lo cual es crucial en aplicaciones donde la exactitud es fundamental.

Facilita la optimización:

Al entender cómo funciona un algoritmo y dónde puede haber ineficiencias, se puede trabajar en su optimización, mejorando su rendimiento.

Permite comparar diferentes algoritmos:

La justificación facilita la comparación de diferentes algoritmos para un mismo problema, lo que permite elegir el más eficiente y adecuado para cada situación.

Es fundamental para el desarrollo de software:

En el desarrollo de software, la justificación de algoritmos es esencial para asegurar que las soluciones implementadas sean correctas y eficientes, lo que impacta directamente en la calidad del producto final.

Es un proceso riguroso que asegura la fiabilidad y eficiencia del mismo, siendo fundamental para el desarrollo de soluciones informáticas correctas y optimizadas.

Existen diversas técnicas para justificar un algoritmo, entre ellas:

Prueba formal:

Se utilizan métodos matemáticos como la inducción matemática para demostrar que el algoritmo produce la salida correcta para todas las entradas válidas.

Pruebas empíricas:

Se ejecutan el algoritmo con diferentes entradas y se comparan los resultados con los esperados, para verificar su funcionamiento y detectar posibles errores.

Análisis de complejidad:

Se analiza la eficiencia del algoritmo en términos de tiempo y espacio de ejecución, utilizando notación asintótica como la notación O .

Análisis de casos límite:

Se estudian los casos más complejos o extremos que pueden presentarse al ejecutar el algoritmo, para asegurar su correcto funcionamiento en estas situaciones.

Desarrollo:**1. Números primos:**

Algoritmo Primos

Escribir "Dame un numero entero"

Leer numeroIngresado;

iteracion = 1;

Mientras iteracion <= numeroIngresado Hacer

 sí (numeroIngresado % iteracion == 0) Entonces

 divisionResiduoCero = divisionResiduoCero + 1;

 Fin Si

 iteracion=iteracion + 1

Fin Mientras

sí(divisionResiduoCero==2) Entonces

 Escribir "Es Primo"

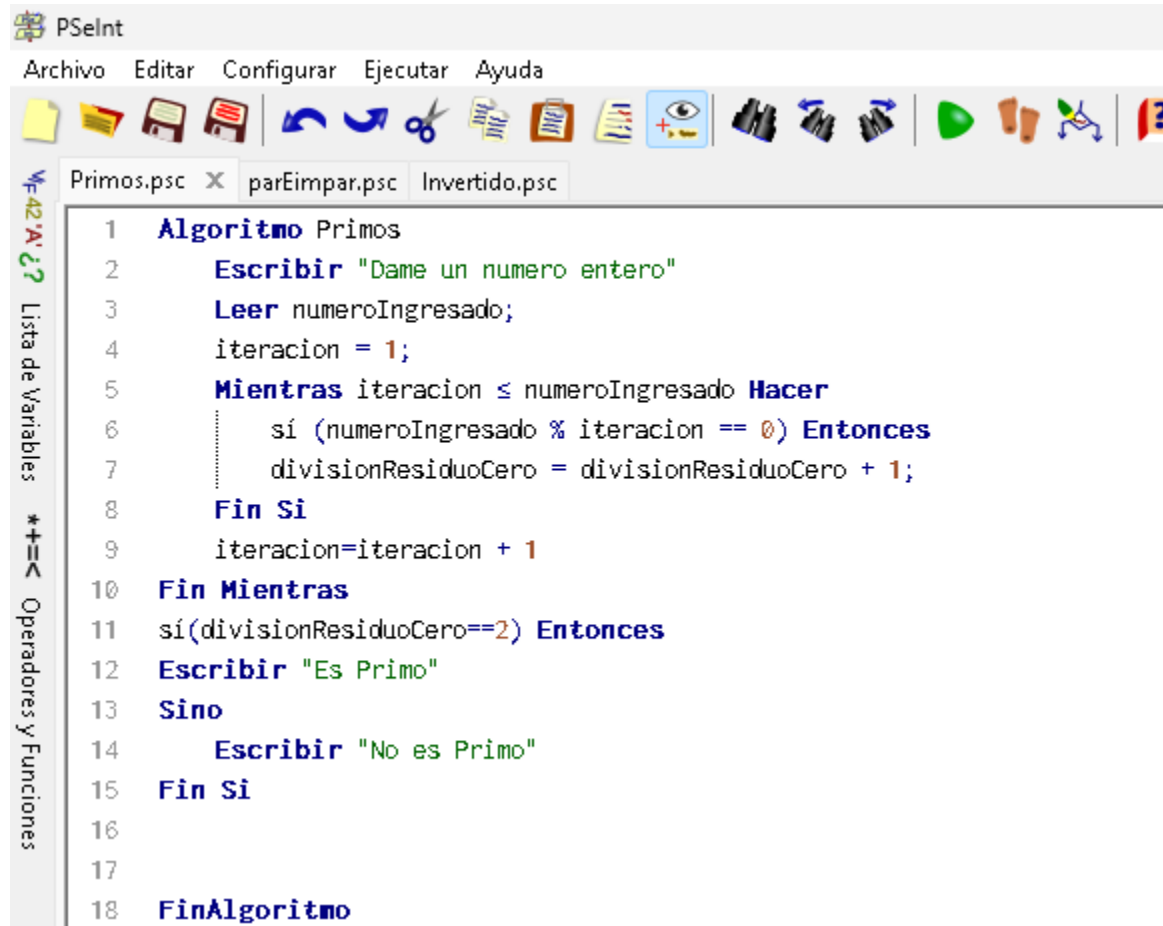
Sino

 Escribir "No es Primo"

Fin Si

Fin Algoritmo

Imagen del algoritmo de números primos:



The image shows the PSeInt software interface. The menu bar includes Archivo, Editar, Configurar, Ejecutar, and Ayuda. The toolbar contains various icons for file operations, editing, and execution. The window title bar shows 'PSeInt' and three open files: 'Primos.psc', 'parEimpar.psc', and 'Invertido.psc'. The main editor displays a Pascal program for finding prime numbers. The program uses standard Pascal syntax with keywords like 'Algoritmo', 'Escribir', 'Leer', 'Mientras', 'Hacer', 'Si', 'Entonces', 'Sino', and 'Fin'. The code is as follows:

```

1  Algoritmo Primos
2      Escribir "Dame un numero entero"
3      Leer numeroIngresado;
4      iteracion = 1;
5      Mientras iteracion ≤ numeroIngresado Hacer
6          .....  sí (numeroIngresado % iteracion == 0) Entonces
7              .....  divisionResiduoCero = divisionResiduoCero + 1;
8      Fin Si
9      iteracion=iteracion + 1
10 Fin Mientras
11 sí (divisionResiduoCero==2) Entonces
12 Escribir "Es Primo"
13 Sino
14     Escribir "No es Primo"
15 Fin Si
16
17
18 FinAlgoritmo
  
```

On the left side of the editor, there is a vertical toolbar with icons for variables, operators, and functions, and a section titled 'Lista de Variables'.

2. Número par e impar:

Algoritmo parEimpar

num=0;

Repetir

num=num+1;

Leer nro;

sí (nro mod 2) =0 Entonces

 Escribir "es par"

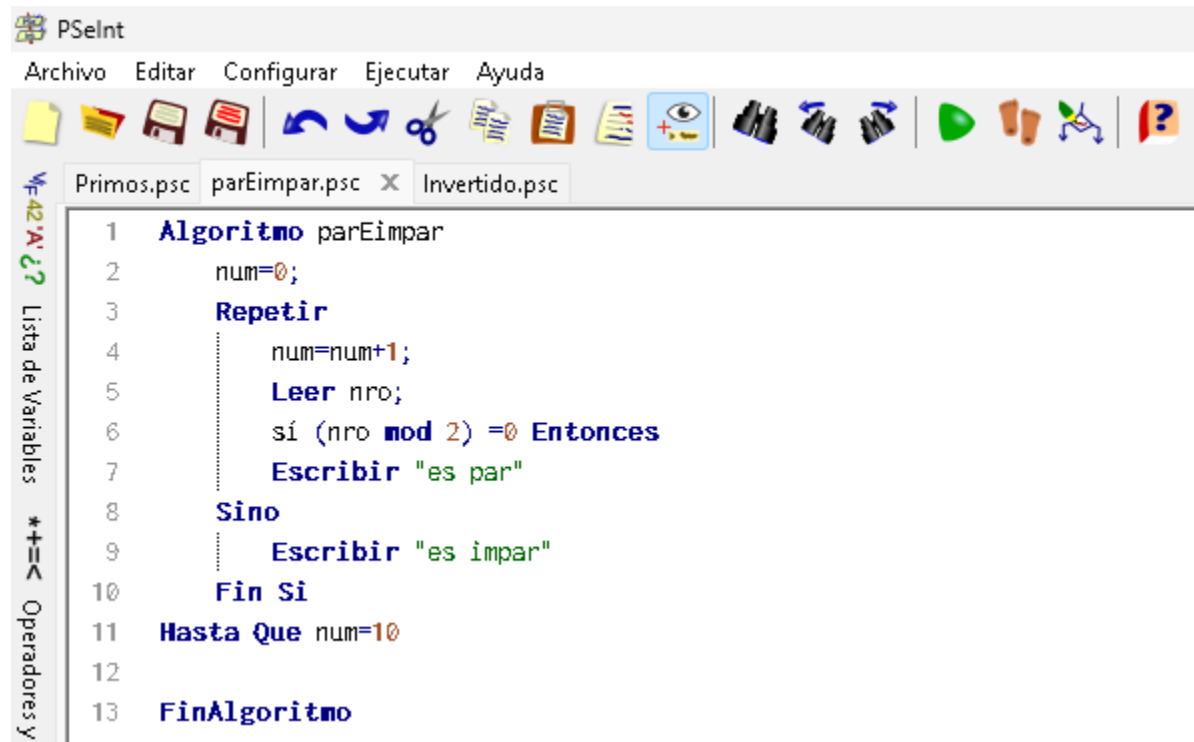
Sino

 Escribir "es impar"

Fin Si

Hasta Que num=10

Fin Algoritmo

Imagen del algoritmo de parEimpar:

The screenshot shows the PSeInt software interface. The title bar reads 'PSeInt'. The menu bar includes 'Archivo', 'Editar', 'Configurar', 'Ejecutar', and 'Ayuda'. The toolbar contains icons for file operations (new, open, save, print), editing (undo, redo, cut, copy, paste), and execution (run, stop, help). The window title bar shows three open files: 'Primos.psc', 'parEimpar.psc', and 'Invertido.psc'. The main editor displays the following pseudocode:

```
1  Algoritmo parEimpar
2      num=0;
3      Repetir
4          .....
4          num=num+1;
5          Leer nro;
6          .....
6          sí (nro mod 2) =0 Entonces
7              Escribir "es par"
8          Sino
9              Escribir "es impar"
10         Fin Si
11     Hasta Que num=10
12
13 FinAlgoritmo
```

On the left side of the editor, there is a vertical toolbar with icons for variables, lists, and operators, and a label 'Lista de Variables'.

3. Números invertidos

Algoritmo Invertido

inverso <- 0

Escribir "Ingresa número"

Leer numero

residuo <- numero

Mientras residuo >0 Hacer

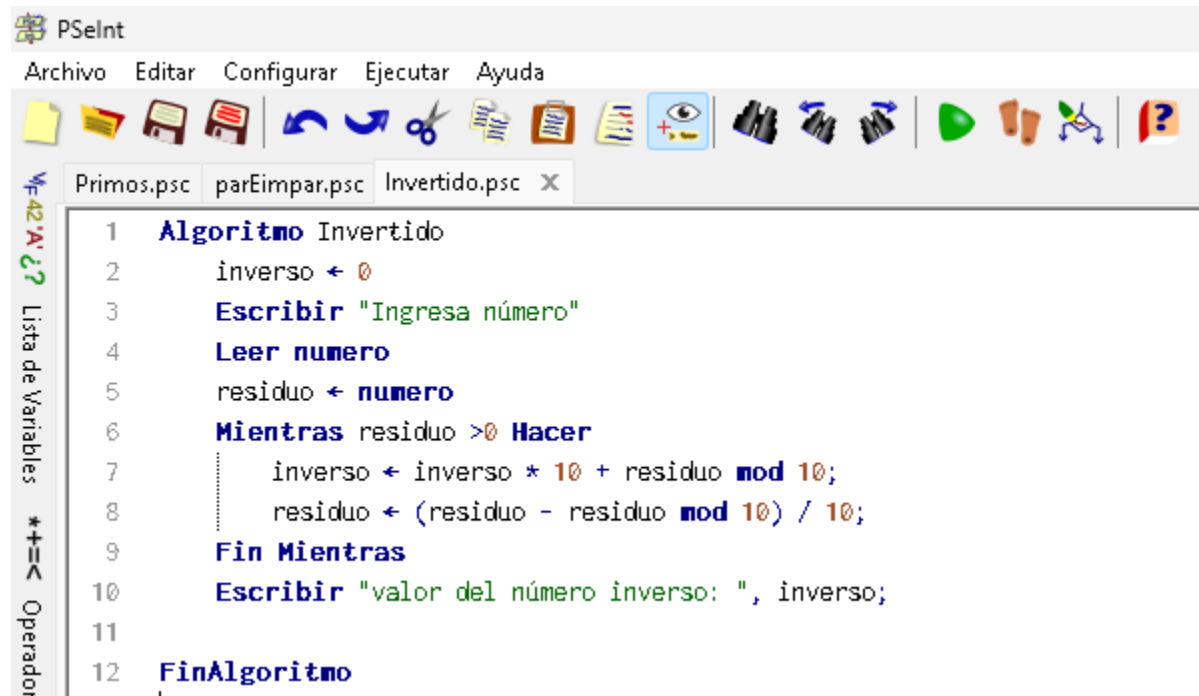
inverso <- inverso * 10 + residuo mod 10;

residuo <- (residuo - residuo mod 10) / 10;

Fin Mientras

Escribir "valor del número inverso", inverso;

Fin Algoritmo

Imagen del algoritmo de números inverso:

The image shows the PSeInt software interface. The menu bar includes Archivo, Editar, Configurar, Ejecutar, and Ayuda. The toolbar contains icons for file operations, editing, and execution. The file explorer on the left shows three files: Primos.psc, parEimpar.psc, and Invertido.psc. The main editor displays the following pseudocode:

```
1  Algoritmo Invertido
2      inverso ← 0
3      Escribir "Ingresa número"
4      Leer numero
5      residuo ← numero
6      Mientras residuo > 0 Hacer
7          .....
7          inverso ← inverso * 10 + residuo mod 10;
8          residuo ← (residuo - residuo mod 10) / 10;
9      Fin Mientras
10     Escribir "valor del número inverso: ", inverso;
11
12 FinAlgoritmo
```

Conclusión:

Es el resultado final o la solución que se obtiene después de ejecutar una serie de pasos definidos para resolver un problema. Este resultado puede ser un valor numérico, un conjunto de datos, una acción, o cualquier otra información relevante para el problema planteado. En esencia, la conclusión valida si el algoritmo logró su objetivo de resolver el problema de manera satisfactoria.

Análisis de la conclusión:

En algunos casos, se puede requerir un análisis más profundo de la conclusión para entender su significado y cómo se relaciona con el problema original.

Salida del algoritmo:

La conclusión es la salida o resultado final que produce el algoritmo después de procesar la información de entrada.

Importancia de la precisión:

Una conclusión precisa y correcta es crucial para la utilidad de un algoritmo, especialmente en aplicaciones donde la exactitud es fundamental.

Validación del proceso:

La conclusión sirve para verificar si el algoritmo funcionó correctamente y si la solución obtenida es la esperada.

Link de Github:

<https://github.com/deyvi23456/-Introduccion-Desarrollo-Software-.git>