Ingeniería en Desarrollo de Software

Actividad | 1 | Algoritmos

Introducción al Desarrollo de Software

Ingeniería en Desarrollo de Software

TUTOR: Sandra Luz Lara Dévora.

ALUMNO: Rodrigo Flores Vázquez.

FECHA: 18 de noviembre 2024

Índice

1. Introducción.

En esta primera unidad desarrollaremos el algoritmo de los tres programas que se piden en nuestra unidad, como primer paso, se deberá de comprender la funcionalidad detrás de cada programa que se solicita y escribirlo en un algoritmo. Asimismo, en el siguiente trabajo pretendemos presentar una serie de concepto y definiciones propios del estudio de los algoritmos, su análisis y diseño.

En el mismo podremos encontrar los conceptos de algoritmo y algunos de sus componentes, análisis y diseño. También veremos los diferentes tipos de formas y tamaños o medidas en que se pueden almacenar y representar los datos y estructuras en un algoritmo o programa. De igual forma podremos ver las definiciones y algunas características, reglas, normas, tipos de algoritmos de búsqueda y ordenación, así como sus aplicaciones.

Finalmente, con el desarrollo del tema podremos comprender la creación de programas informáticos que satisfagan las necesidades de los usuarios y aprender a desarrollar pseudocódigo y algoritmos, plasmándolos en diagramas de flujo.

2. Descripción.

Es importante destacar que los algoritmos son importantes porque permiten resolver problemas y realizar tareas complejas de manera óptima. Son una pieza clave de la inteligencia artificial y tienen muchas aplicaciones, como automatizar tareas repetitivas.

Algunos de los beneficios de los algoritmos son:

- **Soluciones óptimas**

Los algoritmos ofrecen soluciones óptimas para cualquier tipo de problema que se procese en un ordenador.

- **Fiabilidad**

Los algoritmos son fiables porque siempre producen el mismo resultado para la misma entrada.

- **Automatización**

Los algoritmos permiten automatizar tareas repetitivas y complejas, lo que libera tiempo para labores más estratégicas.

Recordemos que los algoritmos son una secuencia ordenada y finita de pasos que permiten resolver un problema. Se caracterizan por ser precisos, definidos, finitos, completos, exactos, abstractos y tener inicio y fin. Es por ello que, para comenzar con el desarrollo de nuestra actividad, se solicita que se sigan los siguientes pasos para la elaboración de dichos programas:

Para cada problema matemático, se deberá:

1. Realizar el algoritmo de cada programa 2. Realizar su diagrama de flujo por cada uno 3. Codificarlo finalmente en lenguaje C.

3. Justificación.

Podemos entender que un algoritmo es un conjunto de reglas definidas que permite solucionar un problema, de una determinada manera, mediante operaciones sistemáticas (no necesariamente ordenadas) y finitas. Estas instrucciones, definidas y ordenadas en función de los datos, resuelven el problema o la tarea. Entiendo que, en realidad los algoritmos se adaptan, transforman y están por todas partes.

Las características de un algoritmo es entender que:

- Las instrucciones o reglas son **finitas**, es decir, hay un número determinado de ellas.
- Son **pasos elementales**. ¿En serio? ¡Claro! Si existen algoritmos complicados es porque estos pasos son muy, muy numerosos (no porque sean enrevesados)
- Se aplican de **forma ordenada**.
- Siempre dan **un resultado** al final.

De las características del algoritmo se deduce que tiene **tres partes**:

- La **entrada o input**: son los datos sobre los que aplica las instrucciones.
- **Procesamiento o instrucciones** que lleva a cabo: con lo recibido en la entrada o input, el algoritmo realizará una serie de cálculos lógicos para resolver el problema.
- **Salida o resultado obtenido**. En la receta, el postre rico, rico; en la SERP de Google, los contenidos que muestra.

Por último, un algoritmo nos ayuda a resolver un problema de forma sistemática e inequívoca. Como, por ejemplo, cada vez que tenemos que lavar, basta con introducir la ropa en la lavadora (más el detergente y el suavizante) y el algoritmo de esta ya se ocupa él solito de dejarla como nueva. O cuando queremos saber la cifra exacta que esconde un porcentaje, regla de tres ¡y listo!

Entiendo totalmente que, con la llegada de los ordenadores, el algoritmo cobra más importancia. Y es que permite obtener un resultado a partir de un gran volumen de datos, y hacerlo en tan solo un segundo (o incluso menos).

4. Desarrollo.

- ***Realizar los algoritmos solicitados en un documento de Word.***

1. Números primos
2. Número par e impar
3. Números invertidos

- ***Explicar la lógica utilizada en cada uno de ellos. Cada algoritmo deberá ir en la sección correspondiente de acuerdo con el índice.***

4.1. Números primos.

Para crear el algoritmo de la calculadora "Primos" que determine si un número es primo o no, podemos seguir una serie de pasos lógicos. Entendemos que un número primo es aquel que solo tiene dos divisores positivos: 1 y él mismo.

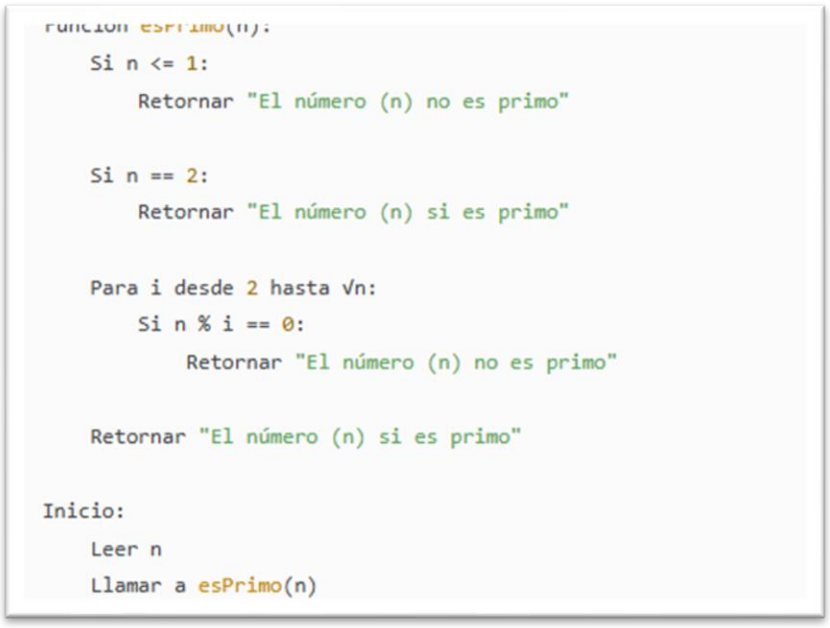
Algoritmo números primos.

1. Entrada: Debemos ingresar un número n.

2. Proceso:

-Si n <= 1, el número no es primo.
-Si n == 2, el número es primo (el único número par primo).
-Para n > 2, debemos verificar si x es divisible entre cualquier número desde 2 hasta la raíz cuadrada de n. Si se encuentra un divisor, el número no es primo.

2. Salida: Mostrar si el número es primo o no.



```
Función esPrimo(n):
    Si n <= 1:
        Retornar "El número (n) no es primo"

    Si n == 2:
        Retornar "El número (n) si es primo"

    Para i desde 2 hasta √n:
        Si n % i == 0:
            Retornar "El número (n) no es primo"

    Retornar "El número (n) si es primo"

Inicio:
    Leer n
    Llamar a esPrimo(n)
```

Explicación del código:

1. Función es_primo(n):

-Si x es menor o igual a 1, retorna que el número no es primo.
-Si x es igual a 2, lo considera como primo (es el único número par primo).
-Luego, usa un ciclo for para verificar si x es divisible por algún número desde 2 hasta la raíz cuadrada de n (esto optimiza el proceso, ya que no es necesario verificar divisores mayores que la raíz cuadrada de n).

2. Entrada y salida:

El programa nos solicitara ingresar un número, luego llama a la función es_primo() y muestra el resultado.

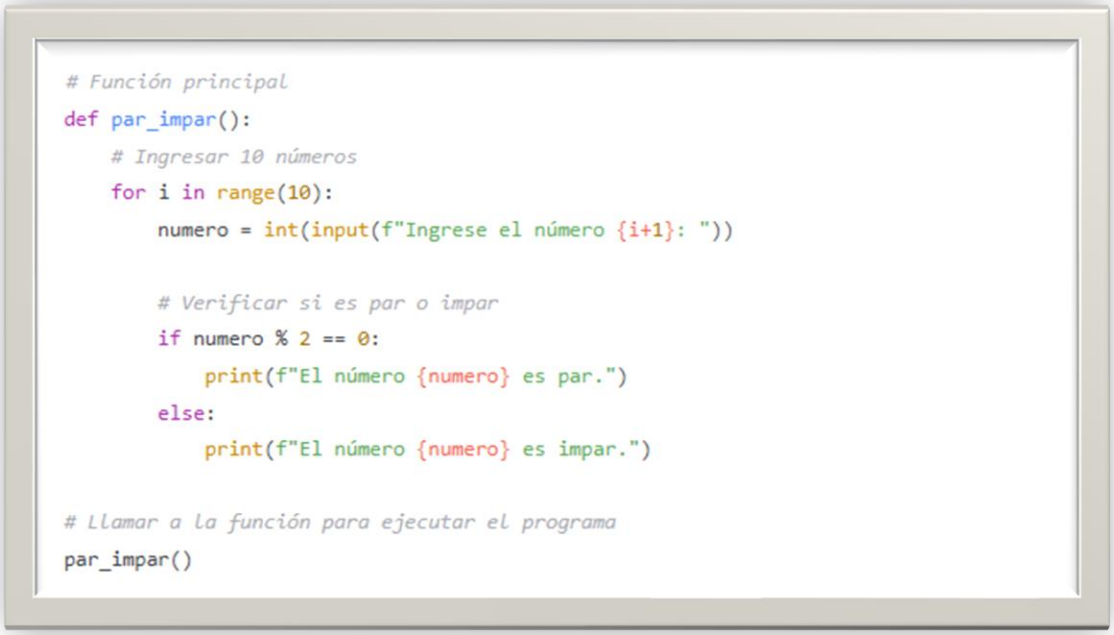
4.2. Número impar e impar.

Algoritmo: Par/Impar

1. Inicio
2. Inicializar un contador i en 1.
3. Repetir el siguiente proceso 10 veces (para ingresar 10 números):

-Ingresar un número entero.
-Solicitar si el número ingresado es divisible por 2 (es decir, número % 2 == 0):
-Imprimir "El número X es par", donde X es el número ingresado.
-Si no es divisible por 2 (es decir, número % 2 != 0):
-Imprimir "El número X es impar", donde X es el número ingresado.

4. Finalizar.



```python
# Función principal
def par_impar():
    # Ingresar 10 números
    for i in range(10):
        numero = int(input(f"Ingrese el número {i+1}: "))

        # Verificar si es par o impar
        if numero % 2 == 0:
            print(f"El número {numero} es par.")
        else:
            print(f"El número {numero} es impar.")

# Llamar a la función para ejecutar el programa
par_impar()
```

Explicación:

1. Usamos un bucle for que se ejecuta 10 veces para ingresar los 10 números.

2. Dentro del bucle, usamos la condición if numero % 2 == 0 para verificar si el número es divisible entre 2. Si es cierto, el número es par; si no, es impar.

3. Se imprime el resultado indicando si el número es par o impar.

Este algoritmo y código permiten ingresar 10 números y determinan si cada uno es par o impar.

4.3. Números invertidos.

Algoritmo "Al Revés":

1. **Entrada:**
   - Ingresar un número entero de 4 dígitos.

2. **Validación:**
   - Verificar que el número ingresado tenga exactamente 4 dígitos. (nuestro rango será entre 1000 y 9999)
   - Si el número no cumple con esta condición, mostrar un mensaje de error e ingresar un número válido.

3. **Proceso:**
   - Convertir el número a una cadena de texto (esto es necesario para manipular los dígitos).
   - Invertir la cadena de texto.

4. **Salida:**
   - Mostrar el número invertido.

```
Inicio
    // Paso 1: Pedir al usuario que ingrese un número de 4 dígitos
    Escribir "Por favor ingrese un número de 4 dígitos:"
    Leer numero

    // Paso 2: Validar que el número tenga 4 dígitos
    Si numero >= 1000 y numero <= 9999 Entonces
        // Paso 3: Convertir el número a texto y luego invertirlo
        numeroInvertido <- Convertir_a_texto(numero)  // Convierte el número a texto
        numeroInvertido <- Invertir_cadena(numeroInvertido)  // Invierte la cadena

        // Paso 4: Mostrar el número invertido
        Escribir "El número invertido es: ", numeroInvertido
    Sino
        Escribir "Error: El número ingresado no es válido. Debe tener 4 dígitos."
    Fin Si
Fin
```

Explicación:

1. Entrada: Se pide al usuario que ingrese un número entero de 4 dígitos.

2. Validación: Se comprueba si el número está en el rango de 1000 a 9999. Si no es así, se muestra un mensaje de error.

3. Proceso: Si el número es válido, se convierte en texto y luego se invierte la cadena de texto.

4. Salida: Finalmente, el programa muestra el número invertido.

5. Conclusión.

Luego de realizar este trabajo hemos visto como los algoritmos son una de las herramientas más complejas y aplicables en el área de la informática y el mundo de los computadores.

Pudimos comprobar que mientras más potente, completo y eficiente es el computador o la aplicación que corre sobre el mismo más grande, complejo y exacto es el algoritmo que utiliza.

Las técnicas de desarrollo de algoritmos nos permiten encontrar la mejor solución a los problemas que se nos presentan y deben ser solucionados por el computador, estas técnicas están orientadas para utilizarse en cada uno de los niveles de complejidad y variedad o alternativas para las cuales se aplican los algoritmos. Un algoritmo es el conjunto de operaciones y procedimientos que deben seguirse para resolver un problema, es por ellos que debemos estudiarlos y conocerlos.

6. Referencias.

Academia Global (s.f.). Introducción al desarrollo de Software. Unidad 1 “Resolución de problemas computacionales”

Correa Uribe, Guillermo. «Desarrollo de Algoritmos y sus aplicaciones», Editora MacGraw – Hill Inc. USA, III Edición. Abril/1992, Colombia. pp. 251.

Algoritmos: qué son y qué tipos existen - Ferrovial. (2024, 16 septiembre). Ferrovial. https://www.ferrovial.com/es/stem/algoritmos/

Ingeniería y Tecnología. (2022, 22 junio). Algoritmo: qué es, para qué sirve, ejemplos de algoritmos y cómo funciona. Unir, Formación profesional. https://unirfp.unir.net/revista/ingenieria-y-tecnologia/que-es-algoritmo/#:~:text=es%20un%20algoritmo?-,Un%20algoritmo%20es%20un%20conjunto%20de%20reglas%20definidas%20que%20permite,el%20problema%20o%20la%20tarea.