PUCE Virtual
EDUCACIÓN FLEXIBLE

Pensamiento Computacional

Algoritmos y programación: Cadenas de caracteres

Clase 15

Ingeniería en ciberseguridad

La excelencia no se improvisa

16.1 Algoritmos y programación: Cadenas de caracteres

Una **cadena de caracteres** es una secuencia de caracteres que puede incluir letras, números, símbolos y espacios. A continuación, se presentan algunos ejemplos en pseudocódigo.

Creación y manipulación de cadenas

- **Concatenación**: Unir dos o más cadenas es una operación básica. En pseudocódigo, se podría representar de la siguiente manera:

```
cadena1 ← "Hola, "
cadena2 ← "Mundo"
saludo ← concatenar(cadena1, cadena2)
imprimir(saludo)  // Output: Hola, Mundo
```

- **Subcadenas**: Extraer una parte específica de una cadena es útil en muchos contextos. En pseudocódigo, podemos definirlo así:

```
texto ← "Algoritmos"
subcadena ← subcadena(texto, 1, 5)  // Del carácter 1 al 5
imprimir(subcadena)  // Output: Algor
```

- **Longitud de una cadena**: Conocer la longitud de una cadena es esencial en muchas operaciones. En pseudocódigo, se podría hacer de esta manera:

```
texto ← "Algoritmos"
longitud ← longitud(texto)
imprimir(longitud)  // Output: 10
```

- **Iteración sobre cadenas**: Recorrer una cadena carácter por carácter permite realizar diversas operaciones, como buscar un carácter específico o contar cuántas veces aparece un carácter.

```
texto ← "Algoritmos"
para cada caracter en texto hacer
    imprimir(caracter)
fin para
```

Algoritmos comunes para la manipulación de cadenas

1. **Búsqueda de subcadenas**: Un algoritmo básico es buscar la primera aparición de una subcadena dentro de otra cadena.
2. **Reversión de cadenas**: Invertir una cadena es un ejercicio común para entender la manipulación de cadenas (strings).
3. **Conteo de palabras**: Este algoritmo cuenta cuántas palabras hay en una cadena.

16.1 Algoritmos y programación: Funciones

Una **función** es un conjunto de instrucciones que realiza una tarea específica. Las funciones pueden recibir entradas, llamadas **parámetros**, y devolver un valor como resultado. Esto permite estructurar los programas en bloques más manejables, facilitando la resolución de problemas complejos mediante su división en subproblemas más simples (Martínez & Hernández, 2019).

En pseudocódigo, la definición de una función incluye su **nombre**, los parámetros que recibe y las instrucciones que ejecuta. A continuación, se presenta un ejemplo básico.

Ejemplo:

Definición de una función que suma dos números

función sumar (a, b)

  resultado ← a + b

  retornar resultado

fin función

Invocación de funciones

Una función definida puede ser invocada en cualquier parte del programa. La invocación implica proporcionar los valores para sus parámetros y, en caso de que la función devuelva un valor, capturar ese resultado.

Ejemplo:

Invocación de la función sumar

x ← 10

y ← 20

resultado ← sumar (x, y)

imprimir (resultado)

Salida: 30

Tipos de funciones

Las funciones pueden clasificarse en varios tipos según su propósito y estructura. Algunos de los tipos más comunes incluyen:

1. **Funciones sin retorno (procedimientos)**: Estas funciones realizan una tarea, pero no devuelven un valor. Se utilizan principalmente para ejecutar acciones que no requieren un resultado que deba ser procesado posteriormente.
2. **Funciones con retorno**: Estas funciones realizan una tarea y devuelven un valor que puede ser utilizado posteriormente en el programa.
3. **Funciones con parámetros**: Son funciones que reciben uno o más parámetros para personalizar su comportamiento. Esto permite que la función sea más flexible y adaptable a diferentes situaciones.

Ejemplo:

Definición de una función que calcula el área de un rectángulo

función calcularArea (base, altura)

área ← base * altura

retornar área

fin función

Invocación de la función

base ← 5

altura ← 10

resultado ← calcularArea(base, altura)

imprimir(resultado)

Salida: 50

16.3 Aplicaciones con herramientas: listas y operaciones, listas de listas

Las listas son una de las estructuras de datos más fundamentales y versátiles en la programación. Permiten almacenar múltiples valores en una sola variable y realizar operaciones sobre esos valores de manera eficiente. Las listas de listas, también conocidas como listas multidimensionales, facilitan la representación de estructuras más complejas, como matrices o tablas. En este tema, exploraremos cómo trabajar con listas y listas de listas utilizando pseudocódigo, lo que facilita la comprensión de estos conceptos sin necesidad de un lenguaje de programación específico (Gómez & Morales, 2019). En pseudocódigo, una lista puede definirse de manera sencilla, asignando varios valores a una variable.

Ejemplo:

Definición de una lista de números enteros:

lista ← [10, 20, 30, 40, 50]

Acceso y modificación de elementos en una lista

Los elementos de una lista pueden ser accedidos y modificados utilizando su índice correspondiente. El índice de la primera posición de la lista es 0.

Ejemplo:

Acceso al tercer elemento de la lista:

tercerElemento ← lista [2]

imprimir(tercerElemento)

Salida: 30

Modificación del cuarto elemento de la lista:

lista [3] ← 45

imprimir(lista)

Salida: [10, 20, 30, 45, 50]

Operaciones comunes con listas

Las listas permiten realizar una variedad de operaciones, como agregar elementos, eliminar elementos, concatenar listas, entre otras (Gómez & Morales, 2019).

1. **Agregar elementos**: Se pueden añadir elementos al final de una lista mediante una operación de inserción.
2. **Eliminar elementos**: Los elementos pueden ser eliminados de una lista utilizando una operación de eliminación, ya sea por su valor o por su posición.
3. **Concatenar listas**: Dos listas pueden ser unidas para formar una lista más grande.

Listas de listas

Las listas de listas son estructuras que permiten almacenar listas dentro de otras listas, lo que facilita la representación de datos multidimensionales, como matrices o tablas. Cada elemento de una lista de listas puede accederse utilizando múltiples índices, uno para cada nivel de la estructura.

Creación y acceso a listas de listas

Ejemplo:

Definición de una lista de listas (matriz 2x3):

matriz ← [[1, 2, 3], [4, 5, 6]]

Acceso al elemento en la segunda fila y tercera columna:

elemento ← matriz [1][2]

imprimir(elemento)

Salida: 6

Operaciones en listas de listas

Las operaciones en listas de listas son similares a las que se realizan en listas simples, pero se aplican a cada una de las dimensiones de la estructura.

Ejemplo:

Modificación de un elemento en la matriz:

Matriz [0][1] ← 10

imprimir(matriz)

Salida: [[1, 10, 3], [4, 5, 6]]

Agregar una nueva fila a la matriz:

Agregar (matriz, [7, 8, 9])

Imprimir (matriz)

Salida: [[1, 10, 3], [4, 5, 6], [7, 8, 9]]

El manejo de listas y listas de listas es esencial en la programación, ya que estas estructuras de datos permiten representar y manipular colecciones de información de manera eficiente. Su comprensión es clave para desarrollar aplicaciones complejas y optimizar el procesamiento de datos (López & Pérez, 2020).

REFERENCIAS

Escudero, L., & Pérez, R. (2019). *Fundamentos de programación y estructuras de datos*. Editorial Tech.

García, M., & Ortiz, A. (2018). *Programación avanzada en Python*. Ediciones Profesionales.

Gómez, J., & Morales, A. (2019). *Estructuras de datos y algoritmos con pseudocódigo*. Editorial Ra-Ma.

Hernández, J. (2020). *Algoritmos y estructuras de datos en Python*. Editorial Académica.

López, A., & Pérez, D. (2020). *Programación avanzada con pseudocódigo*. Ediciones Académicas.

PUCE Virtual
EDUCACIÓN FLEXIBLE

La excelencia no se improvisa

síguenos

MDASC - 04 - 251