Universidad Nacional Autónoma de México

Facultad de Ingeniería

Ingeniería en Computación

Estructuras de Datos y Algoritmos I

Actividad Asíncrona 2

“Acordeón sobre lenguajes de programación”

Alumno: Carmona García Gabriel Alexander

Profesor: Marco Antonio Martínez

28/02/2021

En este trabajo daremos un repaso al lenguaje de programación C, explicaremos brevemente su historia, su funcionalidad, algunas ventajas y desventajas y conceptos sobre este. También daremos las mismas propuestas, pero con otro lenguaje de programación en este caso trabajaremos sobre Groovy, sin nada más que agregar comenzaremos con el trabajo.

Lenguaje de programación C

El lenguaje C fue desarrollado en el año 1972 por Dennis Ritchie para UNIX un sistema operativo multiplataforma. Este tiene instrucciones que son muy parecidas a otros lenguajes incluyendo sentencias como if, else, for, do y while. A este lenguaje se le considera de un nivel alto (puesto que es estructurado y posee sentencias y funciones que simplifican su funcionamiento) aunque también es posible usarlo a un nivel bajo. Para simplificar el funcionamiento en el lenguaje C tiene incluidas librerías de funciones para ayudarnos.

Tenemos que el C nos beneficia en que:

- No depende del hardware, por lo que se puede migrar a otros sistemas.
- Objetivos generales. No es un lenguaje para una tarea específica, pudiendo programar tanto un sistema operativo, una hoja de cálculo o un juego.
- Ofrece un control absoluto de todo lo que sucede en el ordenador.
- Organización del trabajo con total libertad.
- Rico en tipo de datos, operadores y variables en C.

Pero por otro lado tenemos que sus desventajas:

- No es un lenguaje visual, no puede ser deducido de forma intuitiva, como por ejemplo el Visual Basic.
- Encapsulación.
- Para el uso de funciones anidadas necesita de extensiones.
- No cuenta con instrucciones para programación dirigida a objetos.

Su sintaxis es muy práctica, ya que primero deberemos incluir las bibliotecas que queremos usar y después depende del programa que vayamos a realizar, pero en general hablamos de poner la función principal, definir variables, la entrada y salida. A continuación, daremos un ejemplo sencillo:

Al hablar de las bibliotecas en C nos referimos a una serie de librerías que son usadas en este lenguaje de programación y están formadas por una serie de código con el que podemos, en términos prácticos son aquellas que nos permiten realizar más acciones dentro del código. Entre las librerías más destacadas encontramos **stdio.h** (de entrada y salida), **conio.h** (potente gestión de textos), **math.h** (encargada de operaciones matemáticas), **time.h** (obtienen fecha y hora actual para poder utilizarla según sus necesidades), etcétera.

Ahora comenzaremos a dar unos conceptos de las funcionalidades que usamos en este lenguaje de programación:

Algoritmo: es un método para resolver problemas mediante una serie de pasos precisos, definidos y finitos.

Diagrama de Flujo: Un diagrama de flujo es la representación gráfica de un algoritmo.

Comentarios: Estos se usan para comodidad del programador y hacer anotaciones importantes, se escribe de la sig. Forma: // buenas o en su defecto para varias líneas: /*buenas*/.

Variables: Identificador utilizado para representar un cierto tipo de información, cada variable es de un tipo de datos determinado.

Punto y coma: Indica que la línea de instrucción termina.

Función printf: Permite imprimir información por la salida estándar(monitor).

Función scanf: Permite leer datos del usuario.

Sentencias de control: Su función es determinar el orden en que se irán ejecutando las diferentes sentencias dentro de un programa.

Instrucción de selección if: Si la condición definida por la expresión1 es verdadera, entonces se ejecuta la o las sentencias siguientes.

Instrucción de selección if-else: Si la condición definida por la expresión 1mes verdadera, entonces se ejecuta la o las sentencias siguientes. Si la condición resulta falsa, se ignora la o las sentencias siguientes, y se ejecuta el siguiente fragmento de código.

Instrucción switch: En forma ocasional, un algoritmo contendrá una serie de decisiones, en las cuales una variable se probará por separado contra cada uno de los valores constantes enteros que puede asumir, y se tomarán diferentes acciones.

Sentencia for: Es una sentencia que implementa un bucle, es decir, que es capaz de repetir un grupo de sentencias un número determinado de veces.

Sencencia while: Estos bucles se utilizan cuando queremos repetir la ejecución de unas sentencias un número indefinido de veces, siempre que se cumpla una condición.

Sencencia do- while: Se utiliza generalmente cuando no sabemos cuántas veces se habrá de ejecutar el bucle, igual que el bucle while, con la diferencia de que sabemos seguro que el bucle por lo menos se ejecutará una vez.

Funciones: Una función es un bloque de código que realiza alguna operación. Una función puede definir opcionalmente parámetros de entrada que permiten a los llamadores pasar argumentos a la función. Una función también puede devolver un valor como salida.

Arreglos: Colección de datos o variables del mismo tipo, referenciados bajo un mismo nombre y almacenados en localidades consecutivas de memoria.

Arreglo unidimensional: Permite almacenar N elementos de un mismo tipo y acceder a ellos mediante un índice. El primer elemento del arreglo es el elemento cero.

Arreglo Cadena de caracteres: También se pueden declarar cadenas sin inicializarlas, pero en estos casos es indispensable indicar el tamaño que tendrán.

Apuntadores: Un apuntador es una variable que contiene la dirección de otra variable.

Memoria dinámica: Se refiere a aquella memoria que no puede ser definida ya que no se conoce o no se tiene idea del número de la variable a considerarse.

Estructuras: Conjunto de datos o variables de diferentes tipos.

Archivos: Concepto lógico que permite almacenar información de modo permanente y acceder y/o alterar la misma cuando sea necesario.

Sintaxis:

```
#include<stdio.h>
Int main(){
Printf("Hola mundo");
Return 0;
}
```

Lenguaje de programación Groovy

Groovy es un lenguaje orientado a objetos para la Plataforma Java, como alternativa al lenguaje de programación Java. Es un lenguaje dinámico, similar a Python, Ruby, Perl. Además, puede usarse como lenguaje de scripting dentro de la Plataforma Java. Siendo precisos, Groovy es un Lenguaje Dinámico Ágil.

Entre las características que distinguen a Groovy incluyen el tipado estático y dinámico, closures, sobrecarga de operadores, sintáxis nativa para la manipulación de listas y maps,soporte nativo para expresiones regulares, iteración polimórfica, expresiones embebidas dentro de strings.

Al igual que el lenguaje C en este los arreglos se declaran con corchetes[] en lugar de llaves{}; Los "punto y coma" (;) al final de una sentencia son opcionales. El return es opcional.

Closure: Una closure se podría describir como un bloque de código reutilizable, que retiene la información entorno donde fueron definidos.

.each{}: Itera por los elementos de una colección ejecutando el predicado de la closure.

.collect{}: Retorna el valor del uso de cada item de una colección.

.find{}: Busca el primer elemento que cumpla con el predicado de la closure.

.findAll{}: Busca todos los elementos que cumplan con el predicado de la closure.

.inject{}: Permite pasar un valor de una iteración dentro de otra y así sucesivamente..

.every{}: Verifica que todos los elementos de la lista cumplan el predicado de la closure.

.any{}: Retorna true si algún elemento de la colección cumple con el predicado de la closure.

.max / min{}: Retorna el max/min valor de una colección.

.join{}: Devuelve el valor en un string de la concatenación de los elementos de una colección.

Lenguaje es de tipado estático: Los tipos tienen que definirse en tiempo de compilación para que el programa funcione.

lenguajes de tipado dinámico: Los tipos de las variables se definen en tiempo de ejecución.

Sobrecarga de operadores: Cuando reutilizando el mismo operador con un número de usos diferentes, y el compilador decide como usar ese operador dependiendo sobre qué opera.

Sintaxis:

```
groovy> "¡Hola Mundo!"
groovy> go

============
¡Hola Mundo!
```

Otro ejemplo podría ser:

```
groovy> saludo = "Hola"
groovy> "${saludo}" Mundo
groovy> go

============
¡Hola Mundo!
```

Bibliografía:

Fundamentos de programación. Algoritmos y estructuras. Luis Jayanes Aguilar Editorial Mc Graw Hill.

El Lenguaje de Programacon C, Kernighan y Ritchie - Pearson education.

https://www.ecured.cu/C_(lenguaje_de_programaci%C3%B3n)

https://openwebinars.net/blog/que-es-c/

https://tecnoinformatic.com/c-programacion/que-es-groovy/

https://www.ecured.cu/Groovy

https://dosideas.com/cursos/course/introduccion-a-groovy/el-hola-mundo-en-groovy