Nombre del alumno: Johanne Joaquín Arriaga Díaz

Nombre del profesor: María Isabel Roblero Ordoñez.

Nombre del trabajo: Mapa conceptual de unidad I.

Materia: Programación lógica.

Grado: Octavo cuatrimestre

Grupo: ISC13SDC0119-F

Frontera Comalapa, Chiapas a 08 de Marzo de 2022

CONCEPTOS FUNDAMENTALES

La programación lógica y la funcional, forman parte de programación declarativa. La programación tradicional indica cómo resolver un problema mediante sentencias; la programación lógica, trabaja de forma descriptiva, estableciendo relaciones entre entidades, indicando no cómo, sino qué hacer.

La ecuación de Robert Kowalski

Establece la idea esencial de la programación lógica: algoritmos = lógica + control. Es decir, un algoritmo se construye especificando conocimiento en un lenguaje formal (lógica de primer orden), y el problema se resuelve mediante un mecanismo de inferencia (control) que actúa sobre aquél.

Estilos de programación.

También llamado estándares de código o convención de código: describe convenciones para escribir código fuente en ciertos lenguajes de programación y es frecuentemente dependiente del lenguaje de programación que se haya elegido.

Portabilidad

Para asegurar portabilidad, se recomienda usar el estándar de Fortran 77.

Estructura del Programa

La estructura total deberá ser modular. Cada subprograma deberá resolver una tarea bien definida.

Comentarios

Escribir código legible, y agregar comentarios para explicar lo que se está haciendo.

Variables

Declarar todas las variables. No usar declaración implícita. Compactar a 6 caracteres máximo nombres de variables.

Subprogramas

Nunca permitir que las funciones tengan "efectos laterales". En las declaraciones separar parámetros, bloques comunes y variables locales. Minimizar el uso de bloques comunes.

Arreglos

Declarar arreglos grandes en el programa principal y pasarlos como argumentos a las subrutinas. Evitar el innecesario "redimensionamiento de matrices".

Identificadores significativos

Un nombre asociado a un objeto de programa. Debe identificar claramente al objeto que identifica. Debe empezar por una letra, no contener espacios ni símbolos.

Estructura del programa

Un programa debe ser claro, estar bien organizado y que sea fácil de leer y entender.

Evaluación de

¿Que son las expresiones?

* Método de expresar computaciones.
* Compuestas de operadores, operandos, paréntesis y llamadas a funciones.

Los operadores pueden ser:

* Unarios: Solo tienen un operando. Son operadores prefijos.
* Binarios: 2 Operandos. Son operadores infijos.
* Ternarios: 3 operandos.

Orden de la evaluación de operadores.

* Reglas del orden:
* Reglas de procedencia
* Reglas de asociatividad
* Uso de paréntesis

Evaluación de expresiones

Toda expresión regresa un valor. Si hay más de un operador, se evalúan primero operadores mayor precedencia, en empates, se aplica regla asociatividad
Reglas de prioridad:
* Primero, los paréntesis (si tiene)
* Después, orden de prioridad de operadores
* Por último, si aparecen dos o más operadores iguales, se evalúan de izquierda a derecha.

Sentencias de Control de Flujo

Determinan el orden en que se ejecutarán las otras sentencias dentro del programa.

Tipos de expresiones:

Distinguimos dos clases según el tipo de datos que devuelven al evaluarlas.
*Aritméticas: devuelven un valor numérico
*Lógicas: las que devuelven true o false.

Definición de funciones.

Se sigue un estilo consistente de ubicación de llaves y de indentación.

Definición de Funciones

Una aplicación que toma uno o más argumentos y devuelve un valor. Es una correspondencia en la que cada elemento del dominio está relacionado con un único elemento de la imagen.

Las definiciones se incluyen en ficheros de texto. Se acostumbra a identificar dichos ficheros mediante el sufijo. hs.

Los nombres de funciones tienen que empezar por una letra en minúscula.

En Haskell la disposición del texto del programa delimita las definiciones mediante la regla:

“Una definición acaba con el primer trozo de código con un margen izquierdo menor o igual que el del comienzo de la definición actual.”

Toda definición de función tiene por tanto la siguiente forma:

- El nombre de la función
- Los nombres de los parámetros
- El símbolo =
- Una expresión, que puede contener los parámetros, las funciones estándar y otras funciones definidas.

Disciplina de tipos.

—Los tipos se infieren, es decir se comprueban, de forma estática, en tiempo de compilación.

En lenguajes de programación con disciplina de tipos, cada tipo representa una colección de valores o datos similares. El conocerlos tipos ayuda a documentar los programas y evitar errores en tiempo de ejecución.
Es necesario determinar los tipos de los operandos en tiempo de compilación o de ejecución.

Pascal

Cercano a tener disciplina de tipos pero no realiza comprobación de tipos en los registros variantes.

Ada

- Resuelve el problema de los registros variantes realizando comprobación dinámica de tipos (sólo en este caso.
- Tiene una función de biblioteca que permite extraer un valor de una variable de cualquier tipo y usarlo como un tipo diferente.

C

No tiene disciplina de tipos:
* No se realiza comprobación de tipos sobre las uniones
* Permite funciones con parámetros sobre los que no se realiza comprobación de tipos.

Java

- Tiene disciplina de tipos (no hay uniones)

ML y Haskell

- Poseen disciplina de tipos
- Los tipos de los parámetros de las funciones se conocen en tiempo de compilación.

Tipos de datos.

En lenguajes de programación un tipo de dato es un atributo de una parte de los datos que indica al ordenador (y/o al programador) algo sobre la clase de datos sobre los que se va a procesar.

Un tipo de datos define un conjunto de valores y las operaciones sobre estos valores. La mayor parte de los lenguajes de programación permiten al programador definir tipos de datos adicionales, normalmente combinando múltiples elementos de otros tipos y definiendo las operaciones del nuevo tipo de dato.

*Entero en computación es un tipo de dato que puede representar un subconjunto finito de los números enteros
*Datos de coma flotante: float para un número de 4 bytes y doublé para uno de 8 bytes.
*Carácter (Char): Cualquier signo tipográfico, como una letra, número, signo de puntuación o espacio.
*Lógico: O Booleano, representa valores de lógica binaria, esto es, valores que representen falso o verdadero.

Tipos de datos simples:

*Datos Numéricos: Permiten representar valores escalares de forma numérica. Permiten realizar operaciones aritméticas comunes.
* Datos Lógicos: Solo pueden tener dos valores (cierto o falso)
* Datos Alfanuméricos (String): Es una secuencia de caracteres alfanuméricos que permiten representar valores identificables de forma descriptiva.

Los Tipos de Datos En un sentido amplio:

Un tipo de datos define un conjunto de valores y las operaciones sobre estos valores. Casi todos los lenguajes de programación explícitamente incluyen la notación del tipo de datos, aunque lenguajes diferentes pueden usar terminología diferente.