Introducción

Lenguajes de Programación

¿Qué es un lenguaje de programación?

- En la programación de computadores, un lenguaje de programación sirve como medio de comunicación entre el individuo que desea resolver un determinado problema y el computador.
- El lenguaje de programación debe hacer la conexión entre el pensamiento humano (muchas veces de naturaleza no estructurada) y la precisión requerida para el procesamiento por el computador.

Introducción

- Lenguajes de Programación son usados por las personas para expresar un proceso a través del cual un computador puede resolver un problema.
- Comunicación entre las personas y el computador.
  - Para resolver problemas

¿Por qué estudiar lenguajes de programación?

- Mayor habilidad en resolver problemas

  Por ejemplo: conocimiento del concepto de TADs fomenta el uso de este método de programación incluso en LP que no cuentan con mecanismos específicos para su implementación.

- Mayor habilidad al usar un LP

  Por ejemplo: sabiendo cómo se implementan los LP, se puede entender por qué los algoritmos recursivos son menos eficientes que los correspondientes iterativos.

¿Por qué estudiar lenguajes de programación?

- Mayor capacidad para escoger LPs apropiadas

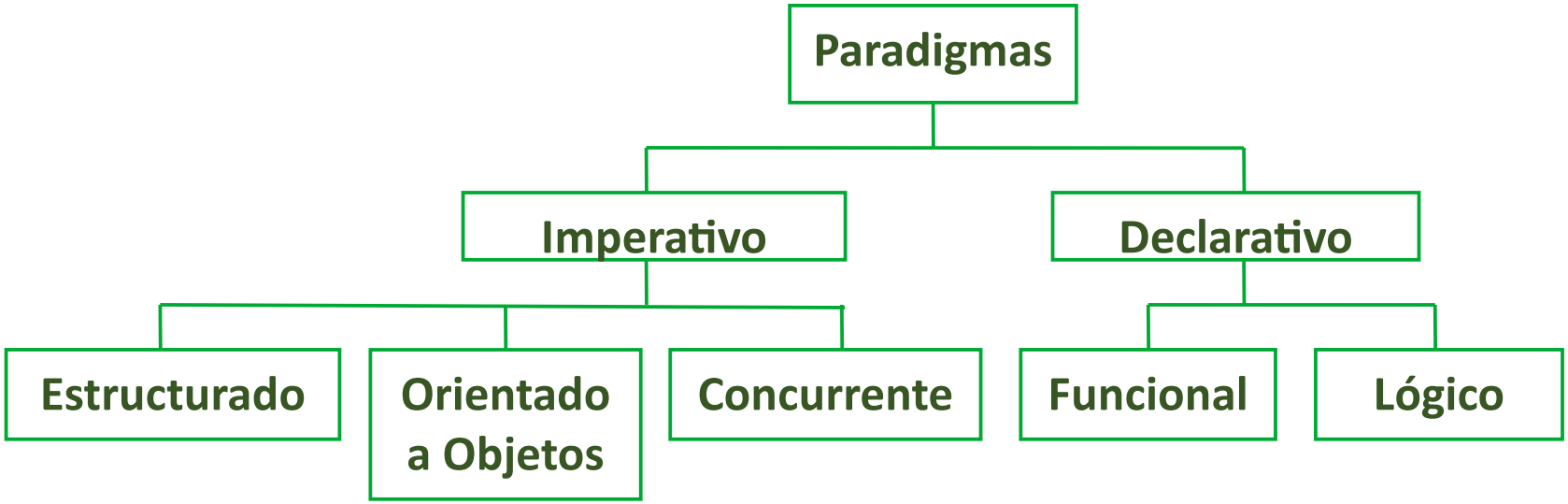
  Por ejemplo: saber que C no realiza comprobaciones dinámicas de índices de acceso a posiciones vectoriales puede ser determinante para elegir este lenguaje en aplicaciones de tiempo real que hacen uso frecuente de accesos vectoriales.

- Mayor habilidad en aprender nuevos Lps.

  Por ejemplo: a programadores que han aprendido conceptos orientados a objetos les resulta más fácil aprender C++ y JAVA.

- Mayor habilidad para diseñar nuevos LPs

Paradigmas de LPs



Paradigmas de LPs

- Imperativo
  - Proceso de Cambios de Estados
  - Variable, Valor y Asignación
  - Celda de Memoria

Paradigmas de LPs

- Estructurado
  - Refinamientos Sucesivos
  - Desestímulo al uso de desvio incondicional
  - Fomentando la división de programas en subprogramas.
  - Bloques Anidados de Comandos.
  - Ej.: Pascal y C

Paradigmas de LPs

- Orientado a Objetos
  - Abstracción de Datos
- Concurrente
  - Procesos Ejecutan Simultáneamente y Concurren por Recursos

Paradigmas de LPs

- Orientado a Objetos
  - Abstracción de Datos
- Concurrente
  - Procesos Ejecutan Simultáneamente y Concurren por Recursos

Paradigmas de LPs

- Declarativo
  - Especificaciones sobre la tarea a ser realizada
  - Se abstrae de Como el computador es implementado
- Funcional
  - Programa compuesto por Funciones
- Lógico
  - Predicados
  - Deducción Automática

Paradigmas de LPs

- Declarativo
  - Especificaciones sobre la tarea a ser realizada
  - Se abstrae de Como el computador es implementado
- Funcional
  - Programa compuesto por Funciones
- Lógico
  - Predicados
  - Deducción Automática

Paradigmas de LPs

- Declarativo
  - Especificaciones sobre la tarea a ser realizada
  - Se abstrae de Como el computador es implementado
- Funcional
  - Programa compuesto por Funciones
- Lógico
  - Predicados
  - Deducción Automática

Origen de los LPs

- Dificuldatad de Programación en Lenguajes de Máquina
- Foco de Primeros LPs era Eficiencia de Procesamiento y Consumo de Memoria
- Baja Produtividad de Programación
  - Programación Estructurada
  - Tipos Abstratos de Datos
  - Orientación a Objetos

Origen de los LPs

- FORTRAN (1957)
  - aplicaciones numéricas
- LISP (1959)
  - programación funcional
- ALGOL (1960)
  - programación estruturada
- COBOL (1960)
  - aplicaciones comerciales

Origen de los LPs

- BASIC (1964)
  - enseñar a iniciantes
- PASCAL (1971)
  - enseñanza de programación estructurada
  - simplicidad
- C (1972)
  - implementación de UNIX
- PROLOG (1972)
  - programación lógica

Origen de los LPs

- SMALLTALK (1972)
  - programación orientada a objetos
- ADA (1983)
  - programación concurrente
- C++ (1985)
  - diseminación de programación orientada a objetos
- JAVA (1995)
  - más simple y confiable que C++
  - Internet

Propiedades Deseables en LPs: Legibilidad

- Facilidad con que el programador lee y entiende el código de un programa.
- Cuanto más fácil leer, más fácil entender y descubrir errores. Ej:
  - Desvíos Incondicionales reducen la legibilidad: goto.
  - Duplicación de Significado de Vocablos

    this (en JAVA) *p = (*p)*q;
  - Marcadores de Bloques: begin/end o { y }. Ausencia o exceso.

```
if (x>1)
    if (x==2)
        x=3;
else
    x=4;
```

Propiedades Deseables en LPs: Legibilidad

- Efectos Colaterales: variable global alterada en función

```
int x = 1;
int retornaCinco( ) {
    x = x + 3;
    return 5;
}
void main( ) {
    int y;
    y = retornaCinco( );
    y = y + x;
}
```

- Ausencia del concepto de palabras reservadas:

  if (if > then) then else (en FORTRAN)

Propiedades Deseables en LPs: Facilidad de Escritura/Codificación de Programas

- Facilidad de codificar programas.
- Tipos de Dados Limitados (FORTRAN) requieren estructuras complejas, dificultando la codificación
- Puede entrar en conflicto con Legibilidad. Ej. C permite codificación de comandos complejos, pero sin dejar muy claro su función

```
void f(char *q, char *p) {
      for(;*q=*p; q++,p++);
}
```

- Ausencia de Tratamiento de Excepciones

Propiedades Deseables en LPs: Confiabilidad

- Mecanismos proveídos por el lenguaje para creación de programas confiables.
  - Declaración de Tipos

```
boolean u = true;
int v = 0;
while (u && v < 9) {
   v = u + 2;
   if  (v == 6) u = false;
}
```

  - Tratamiento de Excepciones

```
try {
        System.out.println(a[i]);
} catch (IndexOutofBoundsException) {
        System.out.println("Error de Indexación");
}
```

Propiedades Deseables en LPs: Eficiencia

- Atender a una determinada demanda por recursos
- Ej:
  - Automatización en tiempo real: lenguajes que minimicen tiempo de ejecución
  - Java verifica índices de vectores a cada acceso, C y C++ no
    - Código generado por C tiene desempeño mejor
    - En contrapartida causa perdida de confiabilidad
  - Verificación Dinámica de Tipos

Propiedades Deseables en LPs: Facilidad de Aprendizaje

- Programador debe ser capaz de aprender el lenguaje con facilidad.
- Lenguajes con muchas formas de realizar la misma cosa tienden a ser más difíciles de aprender. Exceso de características es perjudicial.

```
c = c + 1;
c+=1;
c++;
++c;
```

Propiedades Deseables en LPs: Ortogonalidad

- Capacidad del programador de poder combinar conceptos básicos sin producir efectos anómalos.
- Programador puede prever con seguridad el comportamiento de una determinada combinación de conceptos.
- Falta de ortogonalidad disminuye el aprendizaje y estimula la ocurrencia de errores.

```
int x, y = 2, z = 3;
byte a, b = 2, c = 3;
x = y + z;
a = b + c;
```

Propiedades Deseables en LPs: Reusabilidad

- Posibilidad de reutilizar el mismo código en varios aplicaciones.
- Cuanto más reusable, mayor la productividad.
- Ej:
  - Funciones:

```
void intercambio(int *x, int *y) {
      int z = *x;
      *x = *y;
      *y = z;
}
```

  - Importación de clases en el lenguaje Java

Propiedades Deseables en LPs: Modificabilidad

- Facilidad de alterar el código del programa en función de nuevos requisitos.
- Ej:
  - Uso de constantes simbólicas
  - Separación entre interfaz e implementación

```
const float pi = 3.14;
```

Propiedades Deseables en LPs: Portabilidad

- Programas se comportan de la misma forma independiente de la herramienta usada para traducirlos para lenguaje de máquina o arquitectura en que ejecutan
- Mismo programa puede ser usado en varios ambientes.
- Ej:
  - Programas Java y máquinas virtuales
- Un diseño riguroso ayuda a la portabilidad

Propiedades Deseables en LPs

- Legibilidad: ¿La lectura del programa es fácilmente comprendida?
- Facilidad de escritura: ¿La implementación refleja el algoritmo? La redacción es sucinta?
- Confiabilidad: ¿Es fácil detectar "errores" del programador?
- Eficiencia: ¿Ejecuta rápido?
- Facilidad de Aprendizaje: ¿Es simple?
- Ortogonalidad: ¿Conceptos pueden ser combinados libremente?
- Reusabilidad: ¿Es posible aprovechar partes en otros programas?
- Modificabilidad: ¿Es fácil alterar programas?
- Portabilidad: ¿Corre de la forma esperada en diferentes plataformas?