**Corrección Prueba 1 Bimestre**

**Miembros:**

* Mark Jonathan Hernández Almeida.
* Ozzy James Lachamin Martinez.
* Luis Enrique Pérez Señalin.
* Martin Sebastian Zambonino Gabela.

1. **Analice diferencias funcionales que puede existir entre los compiladores e intérpretes de los lenguajes de programación actuales.**

**Traducción**:

* **Compiladores**: **Traducción completa** antes de la ejecución a código máquina.
  + **Ejemplo**: Un programa en C es compilado usando gcc, generando un archivo ejecutable (a.out).
* **Intérpretes**: **Traducción línea por línea** durante la ejecución.
  + **Ejemplo**: Un script en Python es ejecutado usando python script.py, traduciendo y ejecutando el código en tiempo real.

**Tiempo de Ejecución**:

* **Compiladores**: **Más rápido** en tiempo de ejecución (ejecuta código maquina).
  + **Ejemplo**: Un juego compilado en C++ ejecuta rápidamente porque ya está en código máquina.
* **Intérpretes**: **Más lento** en tiempo de ejecución (interpreta código a máquina en tiempo real).
  + **Ejemplo**: Un programa en Ruby puede ser más lento porque se traduce y ejecuta línea por línea.

**Errores**:

* **Compiladores**: Detecta **errores en compilación** (antes de ejecutar).
  + **Ejemplo**: El compilador gcc detecta errores antes de ejecutarse al momento de intentar compilar.
* **Intérpretes**: Detecta **errores en ejecución** (mientras se ejecuta).
  + **Ejemplo**: En PHP, un error de sintaxis en una línea se detecta solo cuando esa línea es ejecutada.

**Memoria**:

* **Compiladores**: **Uso eficiente** de memoria.
  + **Ejemplo**: Un programa en C optimizado usa menos memoria durante la ejecución porque el código ya está en un formato optimizado.
* **Intérpretes**: **Mayor uso** de memoria (código y entorno en memoria).
  + **Ejemplo**: Un script en JavaScript puede usar más memoria porque el motor V8 de Node.js mantiene el código y el entorno de ejecución en memoria.

**Compatibilidad**:

* **Compiladores**: **Específico por plataforma**, necesita compilación para cada plataforma.
  + **Ejemplo**: Un programa en C++ compilado en Windows necesita recompilarse para ejecutarse en Linux.
* **Intérpretes**: **Altamente portátil**, ejecuta en cualquier sistema con el intérprete adecuado.
  + **Ejemplo**: Un script en Python puede ejecutarse en Windows, Mac y Linux sin cambios si el intérprete Python está instalado.

**Portabilidad**:

* **Compiladores**: Solo necesita ser compilado 1 vez y su archivo puede .exe o .out puede ejecutarse sin necesidad de otros archivos o programas.
  + **Ejemplo**: Un programa en C++ compilado en Windows puede ejecutarse en cualquier otro dispositivo Windows sin necesidad de tener gcc.
* **Intérpretes**: Necesitan siempre de su programa de interprete para ejecutar, lo que no permite utilizar el código fuente sin el intérprete del lenguaje correspondiente.
  + **Ejemplo**: Un script en Python necesita el intérprete de Python para ser ejecutado, por lo que el código fuente por sí mismo no es suficiente para ejecutar o utilizar el programa.

**Almacenamiento**:

* **Compiladores**: Los lenguajes compilados necesitan crear los archivos en código maquina para ejecutarse lo cual genera más consumo de almacenamiento durante la creación del código.
  + **Ejemplo**: Un programa en C++ compilado va a crear archivos .exe o .out cada vez que compila y que el programador necesite ejecutar el código fuente, lo cual generará un archivo extra.
* **Intérpretes**: Los intérpretes no crean un archivo extra para poder ejecutar el código.
  + **Ejemplo**: Un script en Python al ejecutarse no generará un .exe y solo necesitará que el interprete esté instalado.

**Depuración**:

* **Compiladores**: Los errores de tipo compilación son más difíciles de entender al momento de depurar
  + **Ejemplo**: Un programa en C++ compila el código por completo y puede llegar a dar un error de compilación no entendible.
* **Intérpretes**: Los intérpretes no compilan el código y solo se necesita depurar los errores de lógica.
  + **Ejemplo**: Un script en Python al no tener un correcto funcionamiento se puede depurar y ver el manejo de variables y resultados en tiempo de ejecución, no tendrá errores de compilación.

**Desarrollo y mantenimiento:**

* **Compiladores**: El desarrollo de un compilador es más complicado debido a que debe asegurarse de la correcta traducción a código de destino (máquina) lo cual es complicado y más aún cuando debe compilarse para cada arquitectura de computador (x86, x32, arm, etc) y para cada sistema operativo.
  + **Ejemplo**: Un programa en C++ necesita de gcc y el compilador de gcc es muy robusto y complejo de entender debido a que crea código máquina a partir de código fuente.
* **Intérpretes**: Los intérpretes no necesitan preocuparse por generar código máquina directamente ni de la plataforma de destino.
  + **Ejemplo**: Un script en Python usando el intérprete de Python no se va a preocupar de que el sistema operativo sea distinto ni va a generar código destino (maquina) solo va a ejecutar el código fuente.

**Uso:**

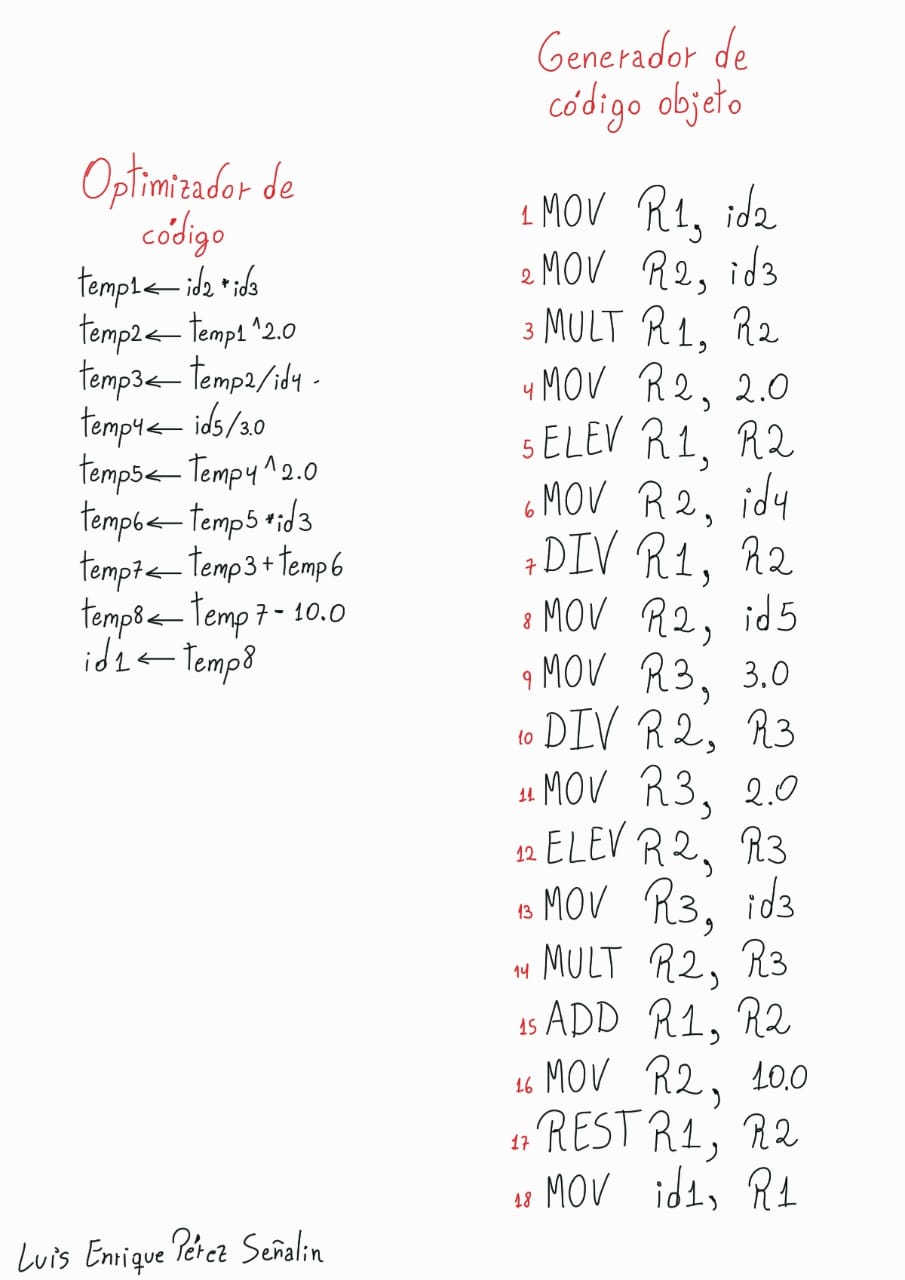
* **Compiladores**: Los compiladores son más usados para programas que necesitan mucha eficiencia y rápidez.
  + **Ejemplo**: El lenguaje C++ puede usarse para crear un videojuego que necesita mucha eficiencia.
* **Intérpretes**: Los interpretes ofrecen rapidez para desarrollar pero son lentos en ejecución, de cierta manera no son usados para programas que necesitan aprovecharla mayor cantidad de recursos posibles como servidores o al menos no a gran escala, por eso algunos utilizan lenguajes con una sintaxis y beneficios cercanos a los del interprete pero con una eficiencia cercana a la de un compilador como el lenguaje Go.
  + **Ejemplo**: Un script en Python va a usar más recursos, más memoria y será más lento que un código C++ o Go, por lo que no serán usados para creación directa de videojuegos, pero existen soluciones como traductores de código Python a C/c++ y demás.

1. **Simule la compilación de la siguiente instrucción**A white paper with a diagram

   Description automatically generated**.**

A diagram of a garden

Description automatically generated

**Grabación de reunión:** <https://epnecuador-my.sharepoint.com/:v:/g/personal/marck_hernandez_epn_edu_ec/EZhzxwcieYZBsPDnfrOuHzcBeVkr7fWsC_OvN7uW1aJLHw?referrer=Teams.TEAMS-ELECTRON&referrerScenario=MeetingChicletGetLink.view.view>