Ámbito

- Para poder utilizar un atributo se debe haber declarado antes dentro de la misma clase.
- Un método se puede llamar de forma recursiva.
- Cuando se declara un atributo dentro de una función utilizando el operador Let, éste es **Local.** Cuando se declara fuera de una función o método pero dentro de una clase, es **Global.**
- Sólo existe el acceso público para todas las clases y atributos.
- Si se definen dos variables, una con ámbito global y otra con ámbito local que tengan el mismo nombre, si no se define el ámbito, se toma el valor de la variable local.
- No pueden haber dos atributos o identificadores con el mismo nombre dentro de un mismo ámbito.
- Si B hereda de A y B sobrescribe un método de A, este debe ser llamado igual que como fue declarado en A y mantener la misma estructura.
- No se puede hacer herencia múltiple o herencia recursiva de clases.

Sistema de tipos

Valores Default:

Regla	Definición
$\frac{\Gamma \vdash a: String}{\Gamma \vdash a : String = ""}$	Los objetos creados a partir de la clase String, poseen valor default “” {cadena vacía}.
$\frac{\Gamma \vdash a: int}{\Gamma \vdash a : int = 0}$	Los objetos creados a partir de la clase Int, poseen valor default 0
$\frac{\Gamma \vdash a: Bool}{\Gamma \vdash a : Bool = false}$	Los objetos creados a partir de la clase Bool, poseen valor default false.

Casteo:

Regla	Definición
$\frac{\Gamma \vdash a: Bool \quad \Gamma \vdash b: int}{\Gamma \vdash a \to b: int \;\; if(a=false: b=0, a=true: b=1)}$	Es posible el casteo implícito de Bool a Int. (False es 0, True es 1)

$\frac{\Gamma \vdash a: int \quad \Gamma \vdash b: bool}{\Gamma \vdash a \rightarrow b: Bool \ \ if(a{=}0: b{=}false, a{>}0: b{=}1)}$	Es posible el casteo implícito de Int a Bool (0 es False, cualquier valor positivo es True)

Expresiones:

Regla	Definición
$\frac{\Gamma \vdash a: int \quad \Gamma \vdash b: int}{\Gamma \vdash (a \ \{+, -, *, / \} \ b): int}$	Los operadores aritméticos se aplican a Objetos creados a partir de la clase Int. El tipo de dato del resultado es del tipo Int.
$\frac{\Gamma \vdash a: A{<=}Static \quad \Gamma \vdash b: A{<=}Static}{\Gamma \vdash (a \ \{==, <, >, <=, >=\} \ b): Bool}$	Los operadores de comparación se aplican a Objetos que del mismo tipo de datos estático o que sean Objetos de clases heredadas de la misma clase. El tipo de dato del resultado es del tipo Bool.
$\frac{\Gamma \vdash a: int}{\Gamma \vdash {\sim}a: int}$	La operación unario ~ aplicado al tipo Int devuelve como resultado un valor del tipo Int
$\frac{\Gamma \vdash a, expr: Bool}{\Gamma \vdash not(a): Bool, if(a{=}true: false, a{=}false, true)}$	La operación unaria not aplicada a una expresión de tipo de dato Bool, devuelve una expresión de tipo de dato Bool.

Operaciones con enteros

Regla	Definición
$\frac{\Gamma \vdash a: Entero \quad \Gamma \vdash b: Entero}{\Gamma \vdash a + b: Entero}$	La suma entre 2 enteros da como resultado un entero.
$\frac{\Gamma \vdash a: Entero \quad \Gamma \vdash b: Entero}{\Gamma \vdash a - b: Entero}$	La resta entre 2 enteros da como resultado un entero.
$\frac{\Gamma \vdash a: Entero \quad \Gamma \vdash b: Entero}{\Gamma \vdash a * b: Entero}$	La multiplicación entre 2 enteros da como resultado un entero.
$\frac{\Gamma \vdash a: Entero \quad \Gamma \vdash b: Entero}{\Gamma \vdash a \ / \ b: Flotante \ if(b =/= 0)}$	La división entre 2 enteros da como resultado un número flotante.

Operaciones entre Strings/Cadenas:

Regla	Definición
$\frac{\Gamma \vdash a: String \quad \Gamma \vdash b: String}{\Gamma \vdash a + b: String}$	La suma de 2 strings equivale a la concatenación de esos 2 strings, resultando en un string.

Asignación:

- La asignación tiene la forma: $< id > \quad <- \quad < expr >$, en donde el valor de expr se convierte en el valor del objeto id.
- El tipo <expr> debe coincidir con el tipo declarado para el <id>, o ser un tipo heredado a partir del tipo de <id>. Es decir, no se puede asignar un expr de tipo String a uno de tipo Int.
- Si id es un atributo de una clase, dicho atributo debe de encontrarse definido dentro de la misma.
- Tanto el lado izquierdo como el derecho permiten identificadores recursivos.

Llamadas a métodos y valores de retorno:

- Los argumentos de los métodos que son de tipos básicos se pasan por valor.
- Todos los argumentos de métodos que sean de tipo básico (int, string, bool) deben ser pasados por valor y no por referencia.
- Los argumentos de los métodos que son tipos derivados se pasan por referencia.
- Los argumentos de un método pueden ser usados como variables locales dentro del método.
- Los argumentos son evaluados de izquierda a derecha.
- El tipo del valor de retorno del método debe coincidir con el tipo de método especificado.

Estructuras de control:

- El tipo de retorno de un expresión utilizada dentro de un if o while debe ser de tipo Bool.
- El tipo de dato del condicional if es el tipo de dato del bloque que sea un supertipo de ambas ramas del condicional.
- El tipo de dato de la estructura while es Object.

Clases especiales

- Existe una clase especial IO que define funciones de entrada y salida de valores tipo Int y Bool.
- La tabla de símbolos debe de contener de forma predefinida la definición de las clases IO, Int, String y Bool junto a sus métodos ya definidos.

Referencias:

Tucker Jr., A. (2004). CRC Handbook of Computer Science and Engineering. CRC.
- Se utilizó el capítulo 97 llamado Type Systems por Luca Cardelli.