GUÍA DE APRENDIZAJE PARA CONCURSANTES ICPC Y IOI: VARIABLES, TIPOS DE DATOS, SENTENCIAS DE DECLARACIÓN Y ASIGNACIÓN

---

1. Introducción

Cuando se codifica un algoritmo independientemente del lenguaje que se use se tiene que hace uso de un concepto que tomamos de matemática como es variable.

Cuando en matemática hacemos uso de este concepto es para referirnos a un valor que desconocemos pero que trabajaremos con ella aún si conocer el valor real.

En programación tambien vamos a apoyarnos en ese concepto para poder trabajar con aquellos valores que desconocemos bien porque son valores introducidos por el usuario en la entrada del programa o valores que son resultados de operaciones realizadas con los datos introducidos por el usuario.

La gran diferencia entre una variable en matemática a una variable en programación es que la de matemática siempre va ser referencia a un valor númerico desconocido, mientras la de programación hace alusión a un valor desconocido pero este no tiene que ser necesariamente númerico.

2. Desarrollo

2.1. Variable

Una variable es un nombre que contiene un valor que puede cambiar a lo largo del programa. De acuerdo con el tipo de información que contienen, en C++ y Java hay dos tipos principales de variables:

1. Variables de tipos primitivos. Están definidas mediante un valor único que puede ser entero, de punto flotante, carácter o booleano.
2. Variables referencia. Las variables referencia son referencias o nombres de una información más compleja: arrays u objetos de una determinada clase.

Desde el punto de vista del papel o misión en el programa, las variables pueden ser:

1. Variables **miembro** de una clase: Se definen en una clase, fuera de cualquier método; pueden ser tipos primitivos o referencias.
2. Variables **locales**: Se definen dentro de un método o más en general dentro de cualquier bloque entre llaves . Se crean en el interior del bloque y se destruyen al finalizar dicho bloque. Pueden ser también tipos primitivos o referencias.
3. Variables **globales**: Caso especial para el tipo de estructura de solución en C++, se define fuera de cualquier método y siempre por encima de cualquiera de ellos; pueden ser tipos primitivos o referencias.

Los nombres de variables se pueden crear con mucha libertad. Pueden ser cualquier conjunto de caracteres numéricos y alfanuméricos, sin algunos caracteres especiales utilizados por los lenguajes de programación como lo operadores aritméticos.

Existe una serie de palabras reservadas las cuales tienen un significado especial tanto para C++ como para Java y por lo tanto no se pueden utilizar como nombres de variables.

2.2. Tipos Primitivos de Variable

Se llaman tipos primitivos de variables de Java a aquellas variables sencillas que contienen los tipos de información más habituales: valores boolean, caracteres y valores numéricos enteros o de punto flotante.

Java dispone de ocho tipos primitivos de variables: un tipo para almacenar valores true y false (boolean); un tipo para almacenar caracteres (char), y 6 tipos para guardar valores numéricos, cuatro tipos para enteros (byte, short, int y long) y dos para valores reales de punto flotante (float y double). Los rangos y la memoria que ocupa cada uno de estos tipos se muestran en la la siguiente tabla

Declaración	**Rango**
boolean	true - false
byte	[-128 .. 127]
short	[-32,768 .. 32,767]
int	[-2^{31} .. 2^{31}-1]
long	[-2^{63} .. 2^{63}-1]
float	[$\pm 3{,}4 * 10^{-38}$ .. $\pm 3{,}4 * 10^{38}$]
double	[$\pm 1{,}7 * 10^{-308}$ .. $\pm 1{,}7 * 10^{308}$]
char	[ú0000'.. úffff'] o [0 .. 65.535]
String	Para almacenar una secuencia de caracteres

En caso de C++ la tabla quedaría de la siguiente manera:

Declaración	**Rango**
bool	true - false
short	[-2^{15} .. 2^{15}-1]
int	[-2^{31} .. 2^{31}-1]
long long	[-2^{63} .. 2^{63}-1]
float	[$\pm 1{,}18e-38$ .. $\pm 3{,}40e38$] Precisión científica (7-dígitos)
double	[$\pm 2{,}23e-308$ .. $\pm 1{,}79e308$] Precisión científica (15-dígitos)
long double	[$\pm 3{,}37e-4932$ .. $\pm 1{,}18e4932$] Precisión científica (18-dígitos)
char	[-128 ... 127]
unsigned char	[0 ... 255]
string	Para almacenar una secuencia de caracteres

En casos de los tipos de datos que soportan datos númericos de tipo entero si se coloca delante del tipo dato el modificador **unsigned** indica que la variable solo soportará enteros neutros y positivos desplazando el rango de 0 a 2^{16}-1 , 2^{32}-1 o 2^{64}-1 según sea el caso.

2.3. Cómo se definen e inicializan las variables

Una variable se define especificando el tipo y el nombre de dicha variable. Estas variables pueden ser tanto de tipos primitivos como referencias a objetos de alguna clase perteneciente al lenaguaje de programación o generada por el usuario. Si no se especifica un valor en su declaración, las variable primitivas se inicializan a cero (salvo boolean y char, que se inicializan a false y $'$). Análogamente las variables de tipo referencia son inicializadas por defecto a un valor especial: null.

```
int x; //Declaracion de la variable x. Se inicializa a 0
int y = 5; //Declaracion de la variable y. Se inicializa a 5
```

2.4. Constante

Contrario a una variable, una constante es un determinado objeto cuyo valor no puede ser alterado durante el proceso de una tarea específica. En C, C++ para declarar variables no existe una palabra especial, es decir, las variables se declaran escribiendo el tipo seguido de uno o más identificadores o nombres de variables. Por otro lado, para declarar constantes existe la palabra reservada const, así como la directiva define. A continuación se muestran ejemplos de declaración de variables y constantes.

```
//Constante primera variante
#define A 100

//Constante segunda variante
const double b = 100;

//Variables
int a,c;
```

A diferencia de las constantes declaradas con la palabra const los símbolos definidos con define no ocupan espacio en la memoria del código ejecutable resultante. El tipo de la variable o constante puede ser cualquiera de los listados en Tipos primitivos, o bien de un tipo definido por el usuario.

En el caso de Java para definir una constante basta con usar la palabra reservada **final**

```
final double PI=3.14159265358979323846;
```

Las constantes son usadas a menudo con un doble propósito, el primero es con el fin de hacer más legible el código del programa

3. Aplicaciones

El uso de variables así como la definición del tipo de datos de cada una de ellas es primodial para la implementación de un algoritmo. El uso del correcto tipo de dato para una variable bien

puede provocar que la solución algorítmica sea correcta o no o que el trabajo con dicha variable tenga o no un mayor o menor nivel de complejidad.