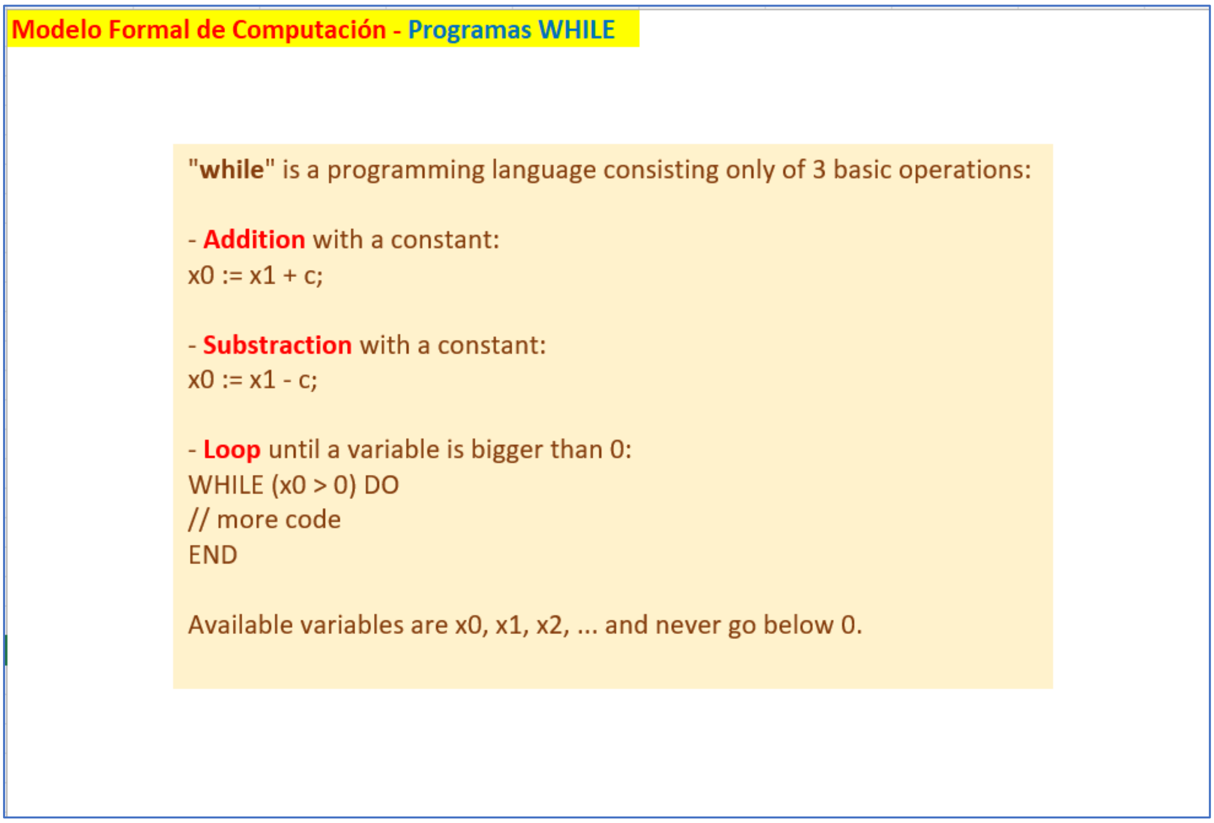
CC184 - Complejidad Algorítmica

Tema:	Computabilidad – Modelo Formal: Programas WHILE
Formato:	Esquema de Aprendizaje
Elaborado por:	Robert Zubieta
Fuente:	Propia

Modelo Formal: Programas WHILE

I. Alcance



Modelo Formal de Computación - Programas WHILE

"**while**" is a programming language consisting only of 3 basic operations:

- **Addition** with a constant:

```
x0 := x1 + c;
```

- **Substraction** with a constant:

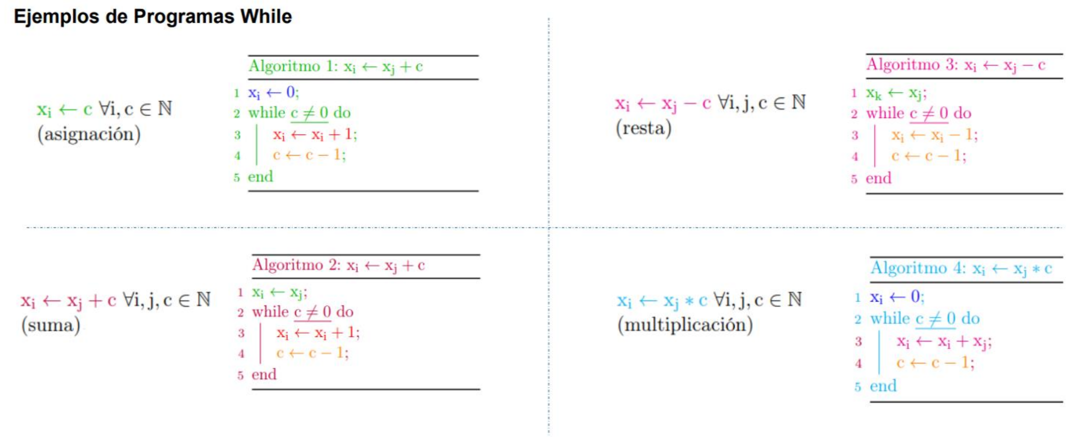
```
x0 := x1 - c;
```

- **Loop** until a variable is bigger than 0:

```
WHILE (x0 > 0) DO
// more code
END
```

Available variables are x0, x1, x2, ... and never go below 0.

II. Ejemplos



Ejemplos de Programas While

$x_i \leftarrow c \ \forall i, c \in \mathbb{N}$ (asignación)

```
Algoritmo 1: $x_i \leftarrow x_j + c$
$x_i \leftarrow 0$;
while $c \neq 0$ do
    $x_i \leftarrow x_i + 1$;
    $c \leftarrow c - 1$;
end
```

$x_i \leftarrow x_j - c \ \forall i, j, c \in \mathbb{N}$ (resta)

```
Algoritmo 3: $x_i \leftarrow x_j - c$
$x_k \leftarrow x_j$;
while $c \neq 0$ do
    $x_i \leftarrow x_i - 1$;
    $c \leftarrow c - 1$;
end
```

$x_i \leftarrow x_j + c \ \forall i, j, c \in \mathbb{N}$ (suma)

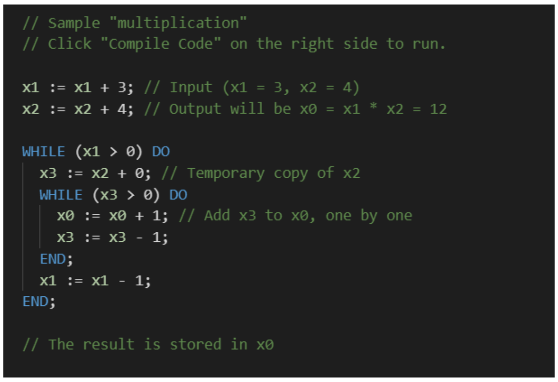
```
Algoritmo 2: $x_i \leftarrow x_j + c$
$x_i \leftarrow x_j$;
while $c \neq 0$ do
    $x_i \leftarrow x_i + 1$;
    $c \leftarrow c - 1$;
end
```

$x_i \leftarrow x_j * c \ \forall i, j, c \in \mathbb{N}$ (multiplicación)

```
Algoritmo 4: $x_i \leftarrow x_j * c$
$x_i \leftarrow 0$;
while $c \neq 0$ do
    $x_i \leftarrow x_i + x_j$;
    $c \leftarrow c - 1$;
end
```

III. Simulador - Multiplica

https://while.schnabel.io/



```
// Sample "multiplication"
// Click "Compile Code" on the right side to run.

x1 := x1 + 3; // Input (x1 = 3, x2 = 4)
x2 := x2 + 4; // Output will be x0 = x1 * x2 = 12

WHILE (x1 > 0) DO
  x3 := x2 + 0; // Temporary copy of x2
  WHILE (x3 > 0) DO
    x0 := x0 + 1; // Add x3 to x0, one by one
    x3 := x3 - 1;
  END;
  x1 := x1 - 1;
END;

// The result is stored in x0
```

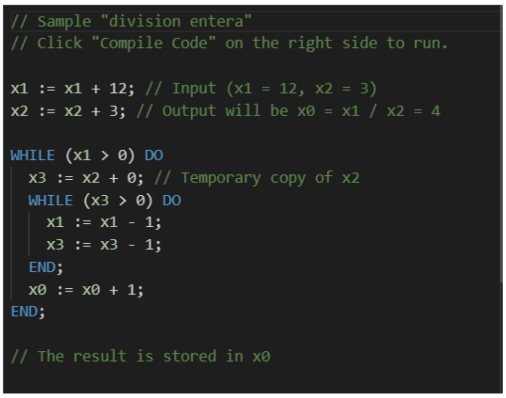
x1 = 3
x2 = 4

1er WHILE controla x1. En cada iteración:
Se asigna x2 a x3
x1 disminuye en 1
=> Habrán 3 iteraciones.

2do WHILE controla x2, a través de x3. En cada iteración
Se incrementa x0: El resultado
x3 disminuye en 1
=> Habran 4 iteraciones.

IV. Simulador- División Entera

https://while.schnabel.io/



```
// Sample "division entera"
// Click "Compile Code" on the right side to run.

x1 := x1 + 12; // Input (x1 = 12, x2 = 3)
x2 := x2 + 3; // Output will be x0 = x1 / x2 = 4

WHILE (x1 > 0) DO
  x3 := x2 + 0; // Temporary copy of x2
  WHILE (x3 > 0) DO
    x1 := x1 - 1;
    x3 := x3 - 1;
  END;
  x0 := x0 + 1;
END;

// The result is stored in x0
```

x1 = 12
x2 = 3

1er WHILE controla x1 y x0. En cada iteración:
  Se asigna x2 a x3
  x0 aumenta en 1 (cociente aumenta en 1)
=> Habrán 4 iteraciones.

2do WHILE controla x2, a través de x3. En cada iteración
  x1 disminuye en 1 (dividendo disminuye en 1)
  x3 disminuye en 1
=> Habran 3 iteraciones. (representa el divisor)