Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI3641 – Lenguajes de Programación I
Abril–Julio 2022

Profesor: Ricardo Monascal Eros Cedeño 16-10216

Examen 1

(25 puntos)

Primera Parte:

a)

i)

Se escogio ECMAScript 6 para la resolucion del examen. Esta es la especificacion de lenguaje de programacion que Javascript adopta. Ecmascript tiene una larga trayectoria, en 1996 se crearon las especificaciones con sintaxis inspirada en Java y C. Actualmente su popularidad esta creciendo dado que Javascript esta implementado en la mayoria de los navegadores web. Es un Lenguaje imperativo, orientado a objetos, en su implementacion: Javascript, ostenta alcance estatico (static scope) dentro de su sintaxis. Javascript es considerado un lenguaje interpretado dado que desde el punto de vista del programador objetivo, no es necesario compilar el codigo antes de ejecutarlo. Sin embargo esto no es asi del todo. Javascript tiene la particularidad de ser un lenguaje compilado, pero a diferencia de la mayoria de lenguajes compilados el se va compilando justo al momento de ejecucion. Es decir Javascript se compila Just in Time. Justo antes de ejecutar el codigo se procede a verificar el parsin (que la sintaxis sea correcta) y luego continua a la ejecucion-compilacion del codigo sin generar archivo ejecutable como es tradicional. Por lo tanto ¿En que momento ocurren las asociaciones? Javascript esta en esa linea delgada entre la asociacion temprana y la asociacion tardia debido a su compilacion JIT. Sin embargo como no son creadas las asignaciones de memoria ni la creacion de las variables ni las tables de valores sino hasta el momento en el que se esta ejecutando el programa se concluye que es de asociacion tardia. Aqui las funciones son consideradas de primera clase.

Los Diseñadores de Ecmascript dotaron al lenguaje de muchas ventajas para los usuarios objetivos, entre ellas tenemos su sintaxis muy similar a los lenguajes de programacion populares y tradicionales como C y Java por lo que la curva de aprendizaje es muy favorable. La compilacion JIT permite que el desarrollo de aplicaciones sea mas rapido porque quita la necesidad de compilar cada vez que se realiza una modificacion al codigo para poder ejecutarlo y dota a la ejecucion de mayor velocidad que los lenguajes interpretados. Otra ventaja es la capacidad de verificar errores de sintaxis antes de ejecutar el codigo. Por otro lado tambien existen desventajas en su diseño: Dado que se compila JIT no se generan archivos ejecutables con codigo fuente, lo que genera la necesidad de compilar cada programa cada vez que se ejecuta, incluso si acaba de ser ejecutado, esto acota su desempeño, a diferencia de los lenguajes compilados que una vez compilado exitosamente no es necesario compilar el mismo codigo otra vez, sino que basta volver a ejecutar el archivo resultado de la ultima compilacion. Otra desventaja para el programador es que el diseño no contempla la sobrecarga de funciones ni de operadores. Por otro lado, Anteriormente el lenguaje estaba aislado a la programacion web, especificamente para el scripting del FrontEnd que ejecutan los navegadores. Esto era una gran desventaja, sin embargo en los ultimos año la situacion viene cambiando y ahora es

posible ejecutar las especificaciones Ecmascript directamente en el servidor usando NodeJS y continua mas alla permitiendo interactuar con bases de datos con sintaxis Ecmascript gracias al manejador de base de datos no relacional MongoDB.

La popularidad de Ecmascript ha crecido considerablemente y ya empresas multinacionacionales de gran embergadura lo usan en sus productos y servicios.

ii)

Ecmascript permite la importacion y exportacion de modulos conocidos como modulos ES o ESM mediante las palabras claves import y export. Los modulos no se exportan por defecto. Para que un modulo pueda se exportado es necesario el uso de la instruccion *export*. En el caso de la exportacion es usando la palabra reservada *import.*

iii)

Es posible el uso de aliases con la instrucción *as.* Por lo que por ejemplo se puede crear un alias de un modulo con la instrucción:

import { myGoodAndAmazingVariable } from "./constants.js" as omg

En contraste con otros lenguajes de programacion ecmascript no soporta la sobrecarga de operadores ni de funciones, sin embargo existen lenguajes basados en Ecmascript que permiten justamente esto. Por ejemplo ExtendScript de Adobe permite la sobrecarga de funciones

Ecmascript admite polimorfismos de subtipos como por ejemplo:

```
class Animal {
	habla() {
		return "Los Animales hablan de diferentes formas"
	}
}

class Pollito extend Animal {
	habla() {
		return  "pio pio"
	}
}

class Gallo extend Anima {
	habla() {
		return "Kikiriki"
	}
}
```

En el ejemplo anterior el metodo habla() es sobre-escrito por los subtipos de la clase Animal. Y su resultado varia dependiendo del subtipo de la clase.

iv) Ecmascript tiene una amplia variedad de profillers y debuggers asi como un gran ecosistema de frameworks. Entre ellos tenemos Chrome Profiller nativo en el navegador web, extensiones como spy-js, los frameworks Nodejs, Express, Angular.js entre otros

b)

link del repositorio git:

https://github.com/Eycer-usb/ci3641/tree/main/Pregunta_1

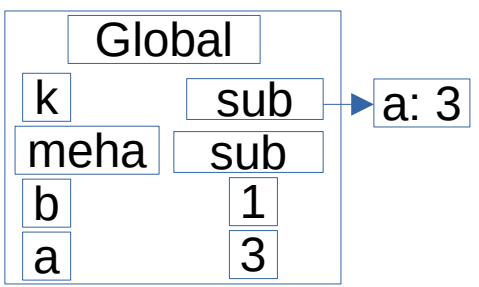
Parte 2:

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

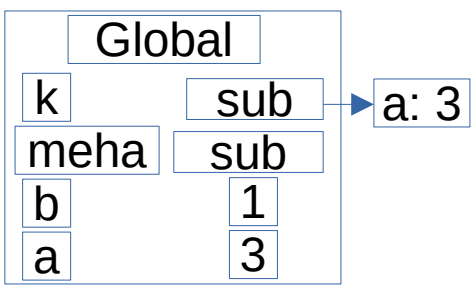
Alcance Estatico y Clausura con Asociacion Superficial

Inicialmente las variables Globales son declaradas.

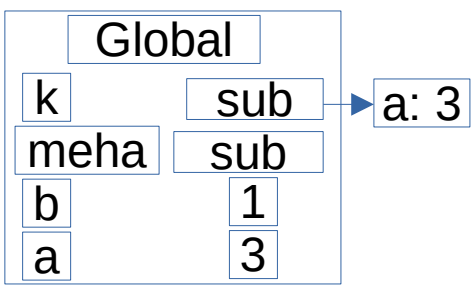
Las clausuras se realizan Al Ejecutar las subrutinas



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

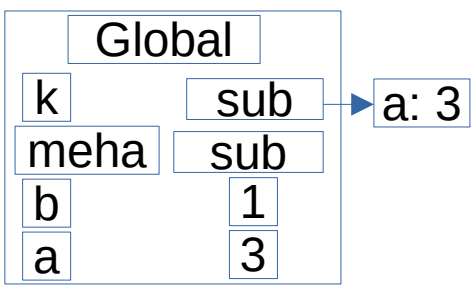


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



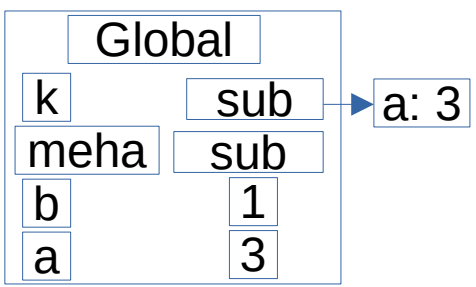
k0
4 a: 7
3 meha: sub
2 ku: meha(global)
1 go: meha(global)
0 b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k0
4 a: 7
3 meha: sub
2 ku: meha(global)
1 go: meha(global)
0 b: 3

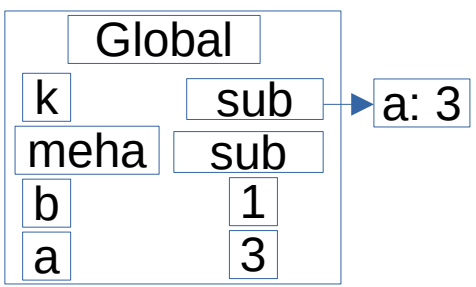
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12

k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

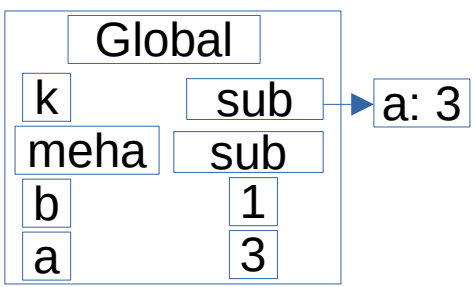
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12

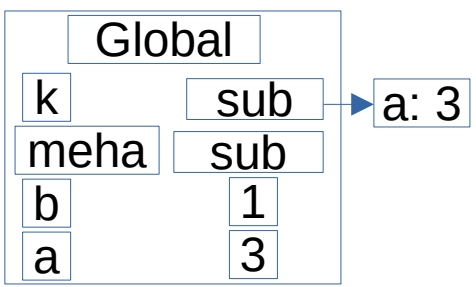
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k2	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

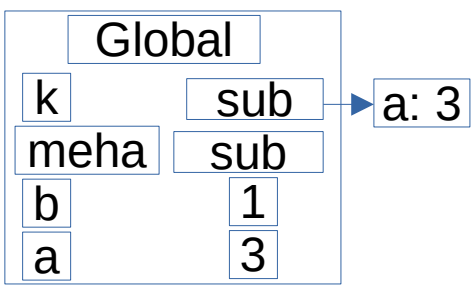
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k2	15	b: -15
	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

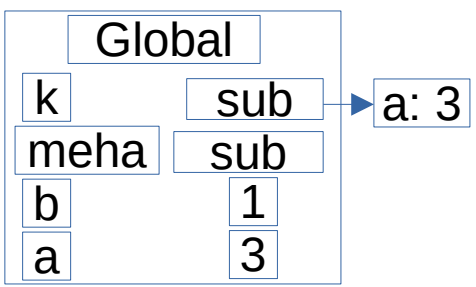
Como se esta ejecutando go se le crea la clausura
Recordando que go11 = ku7= meha(global)

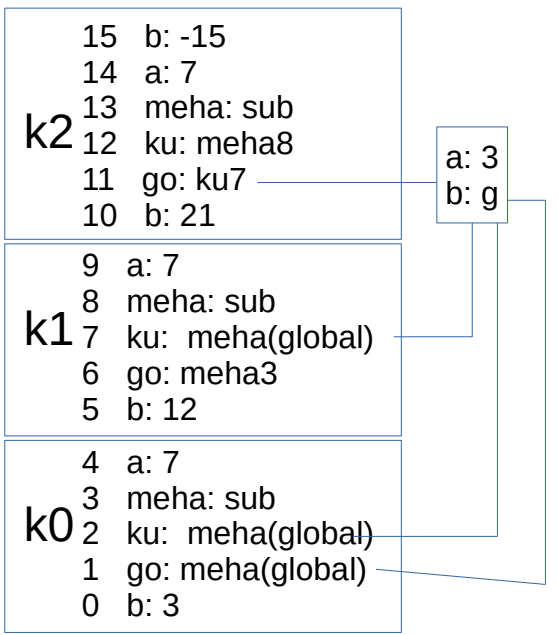


k2	15	b: -15
	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

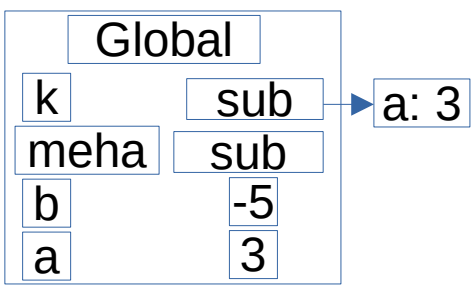
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

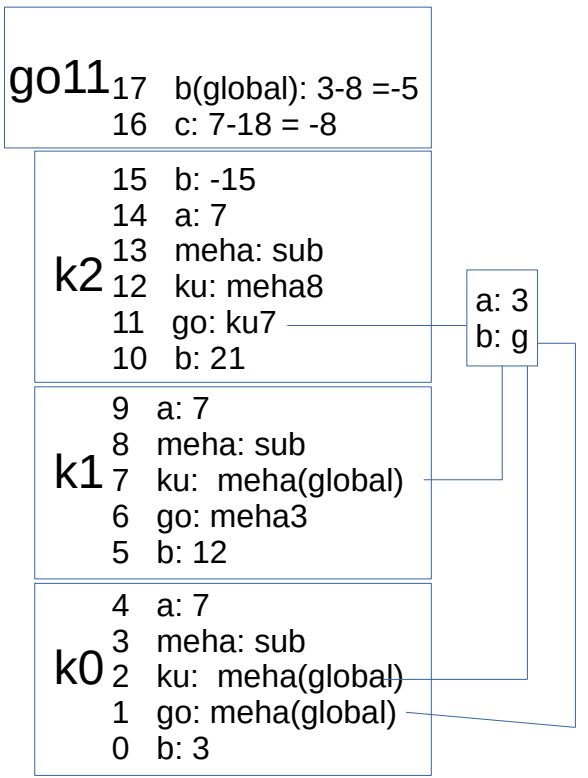
Como hay alcance estatico, en la clausura
Tenemos… a es 3 y b es global





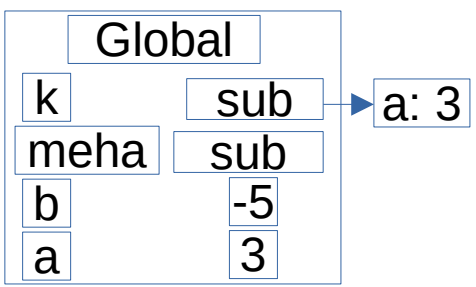
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

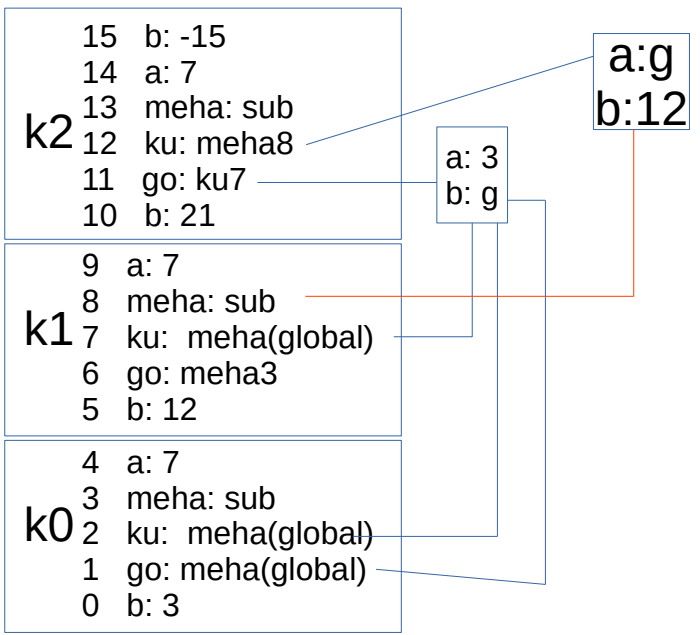




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

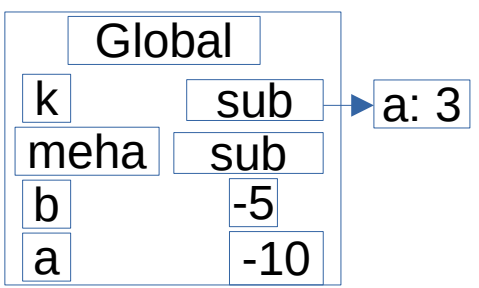
Hacemos la clausura de ku12

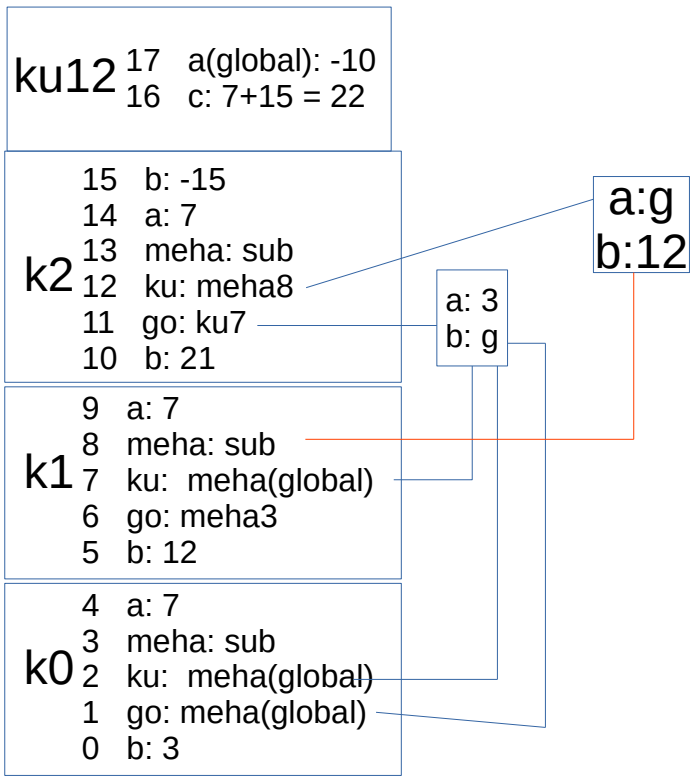




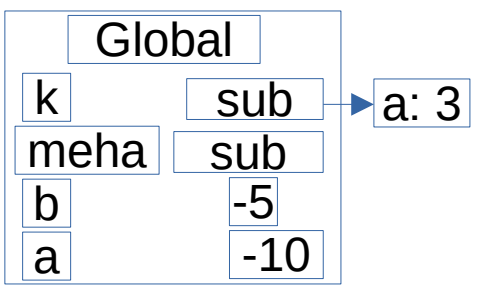
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

    sub meha(int c) {
            a := b - c;
    }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

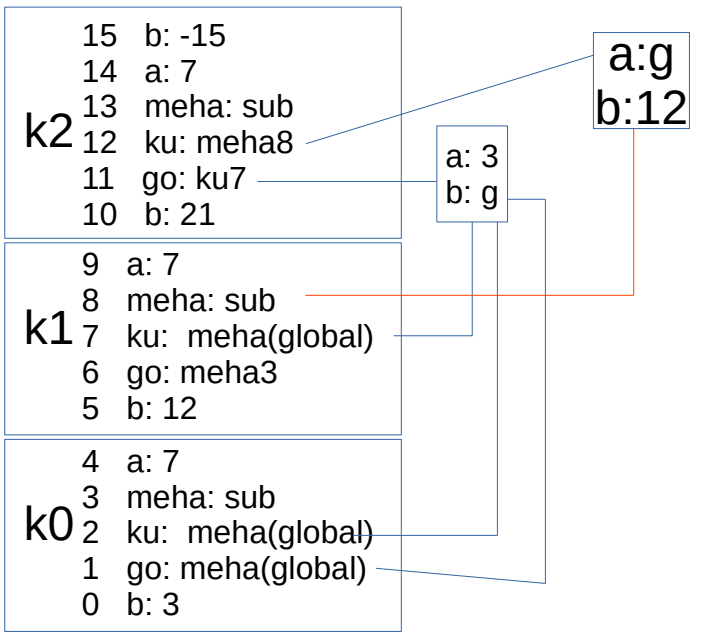
a(global) se convierte en -10





```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

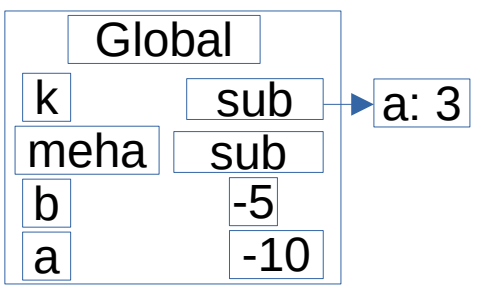


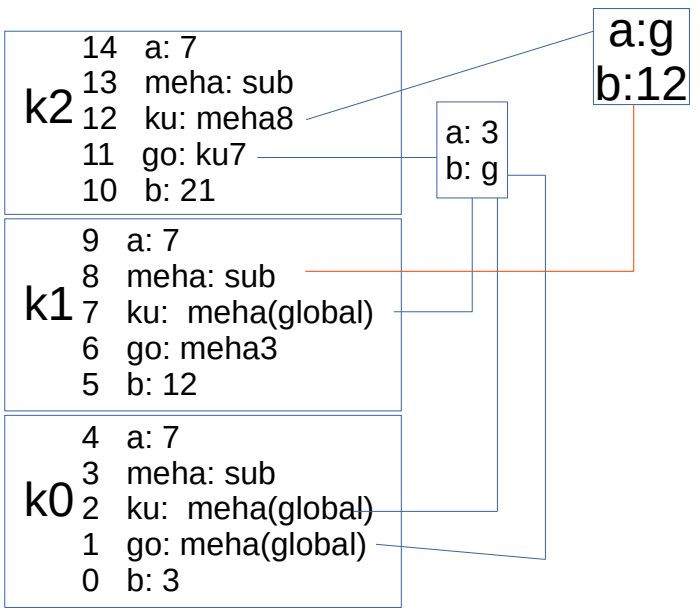


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

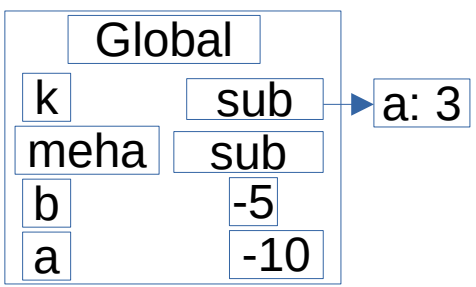


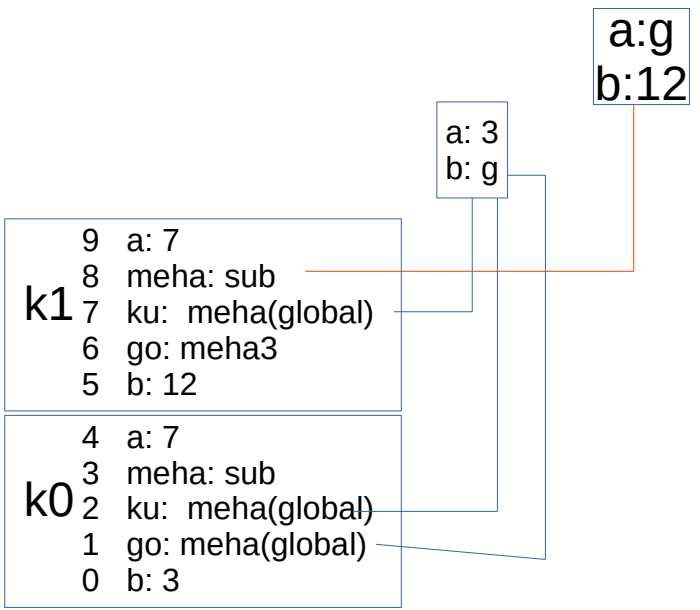


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

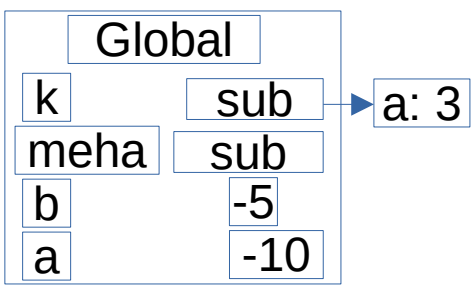


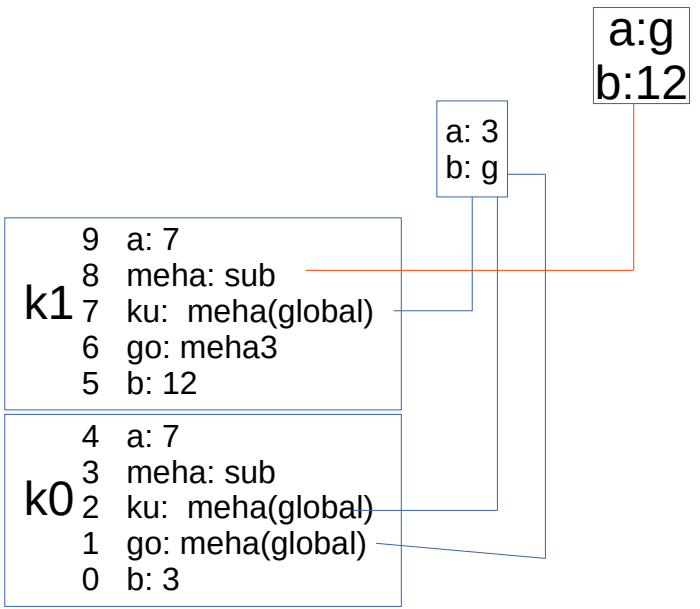


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7  21
7  12
```

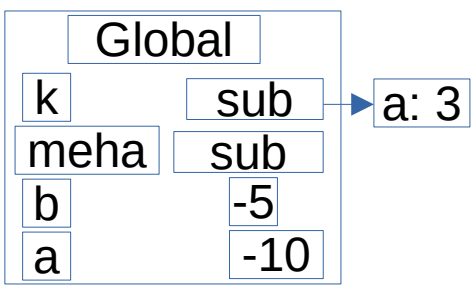


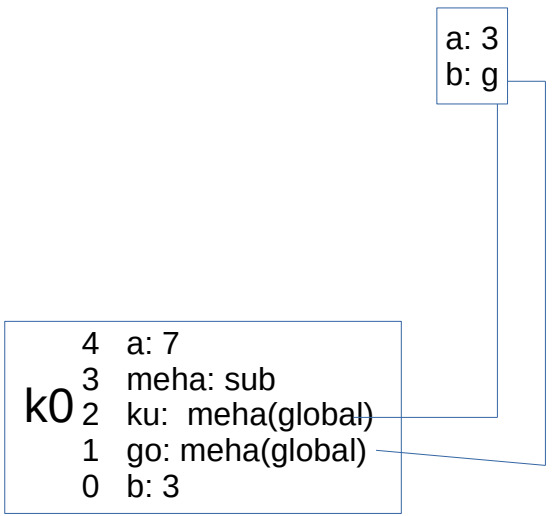


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7  21
7  12
```

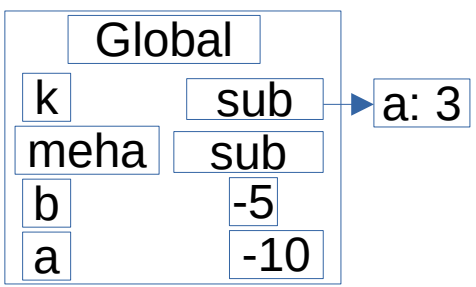


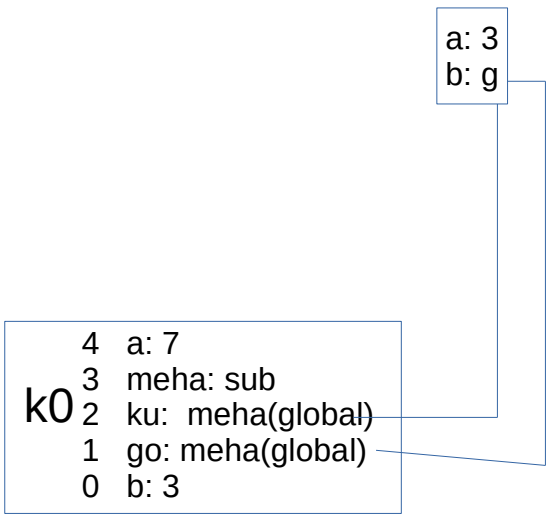


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7  21
7  12
7   3
```

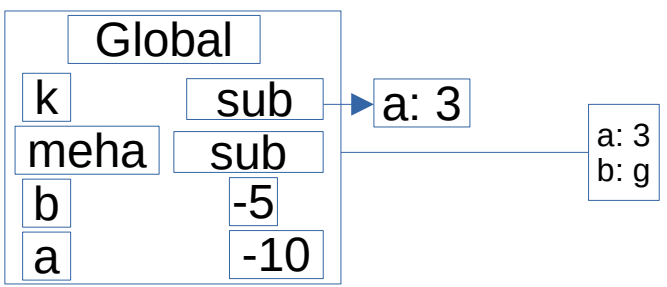




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7  21
7  12
7   3
```

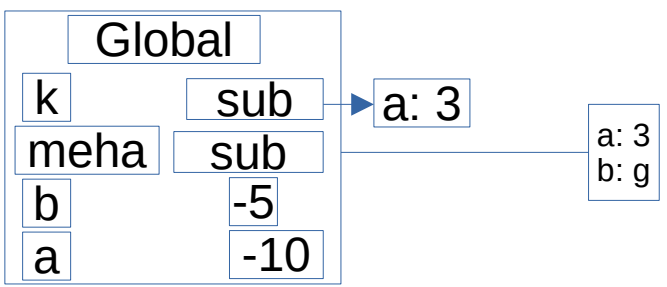


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
```

```
print(a, b)
```

Salida

```
7  21
7  12
7   3
-10 -5
```

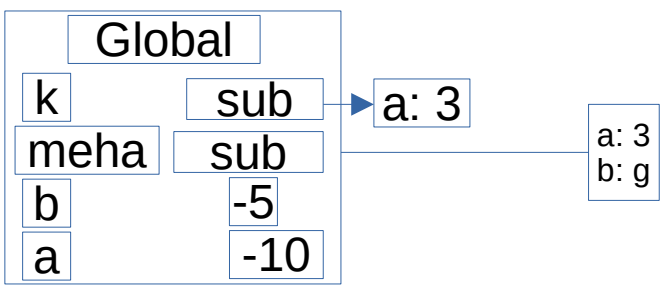


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
 7  21
 7  12
 7   3
-10 -5
```

Finaliza la ejecucion

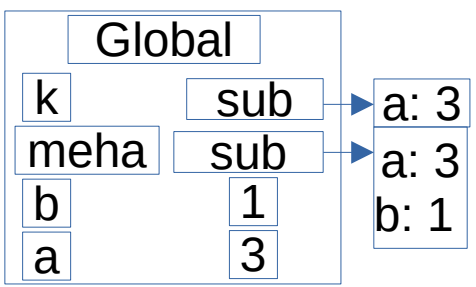


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

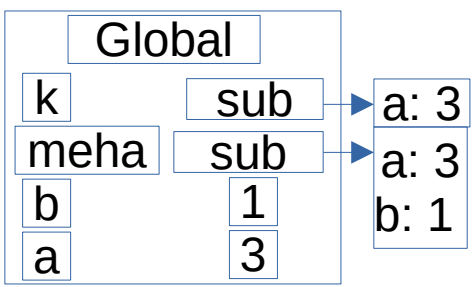
Alcance Estatico y Clausura con Asociacion Profunda

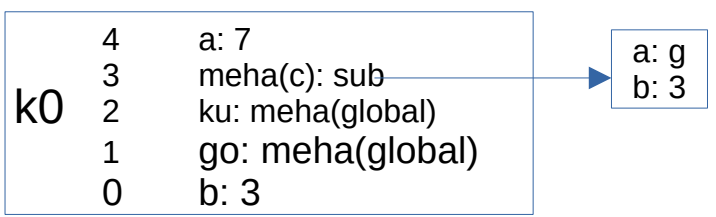
Inicialmente las variables Globales son declaradas.

Las clausuras se realizan Al definir las subrutinas

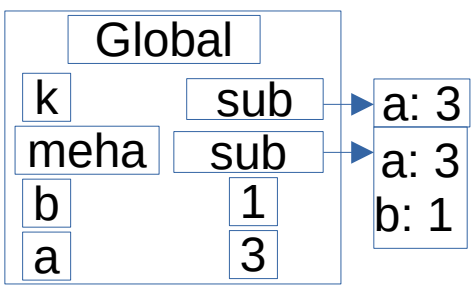


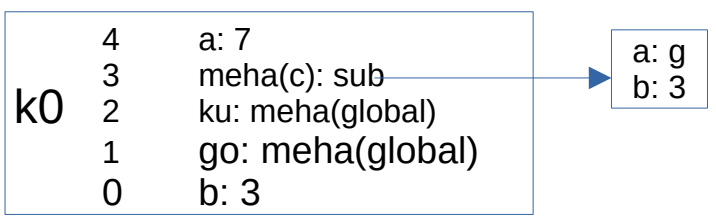
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



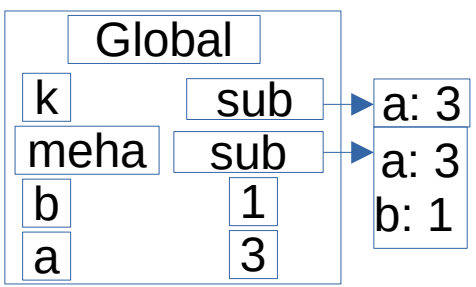


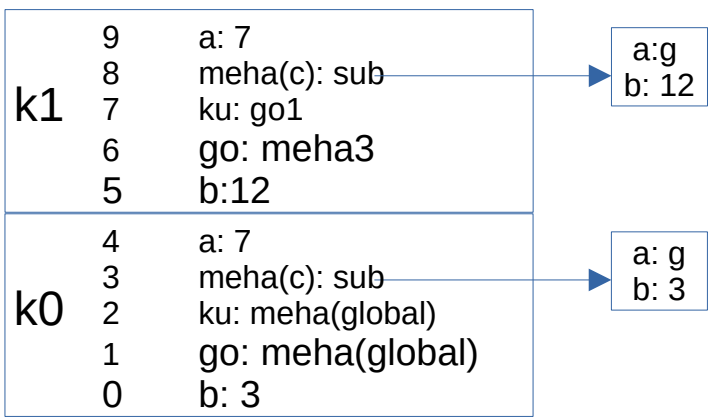
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



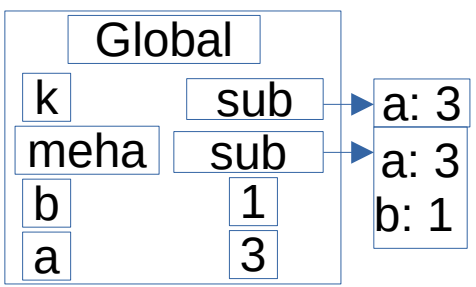


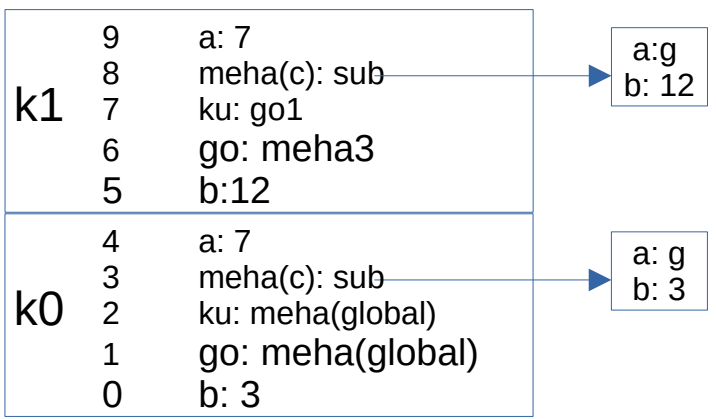
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



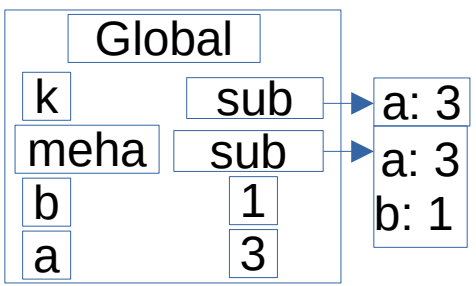


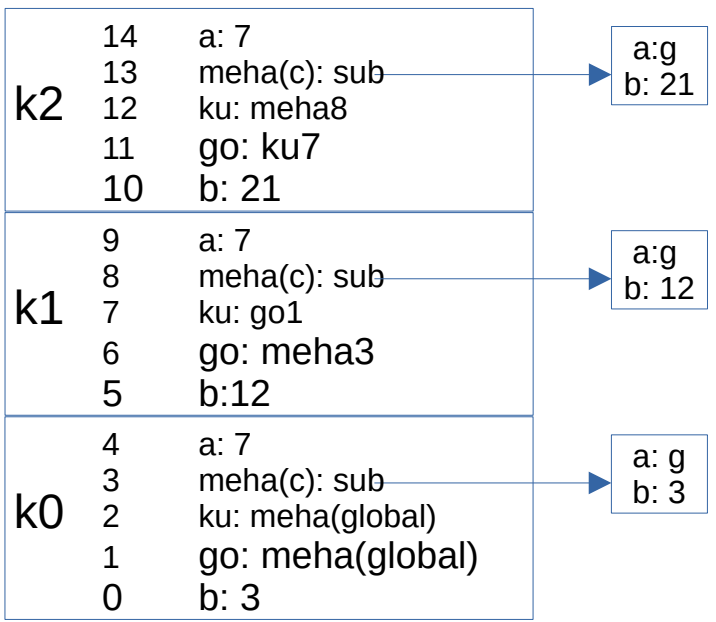
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



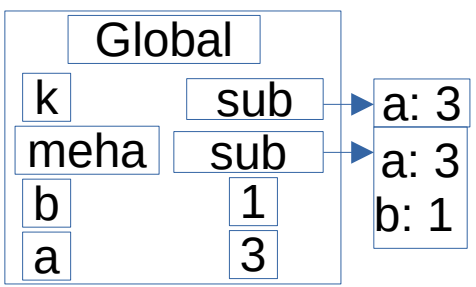


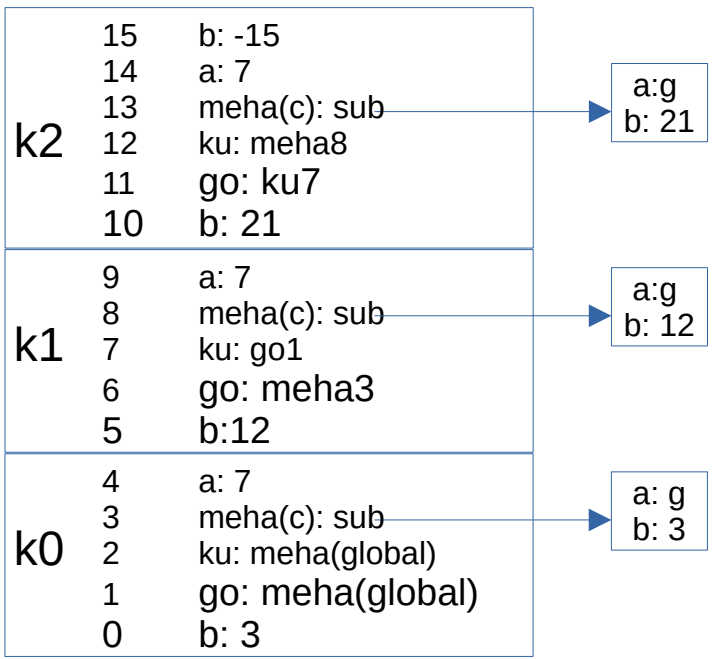
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



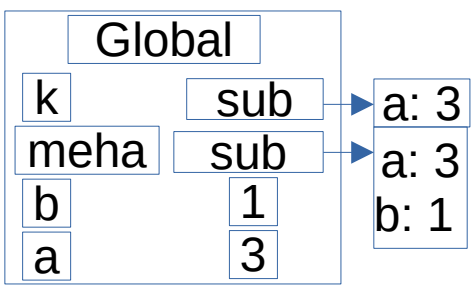


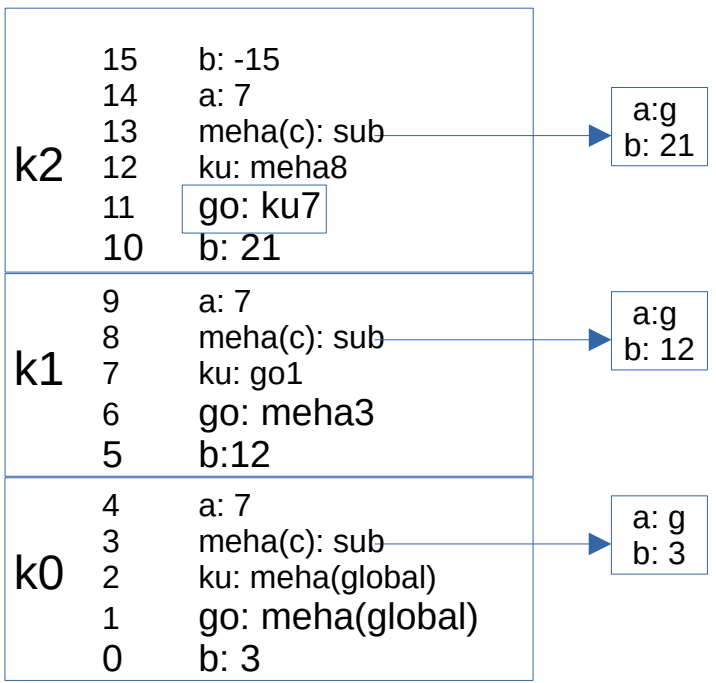
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



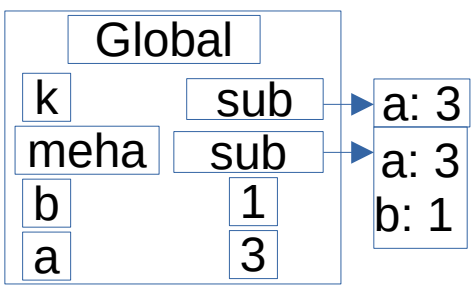


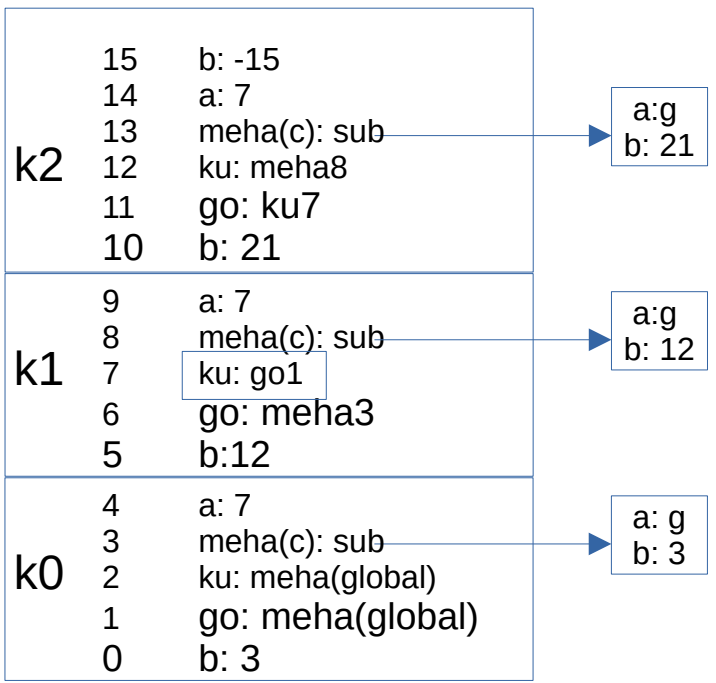
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



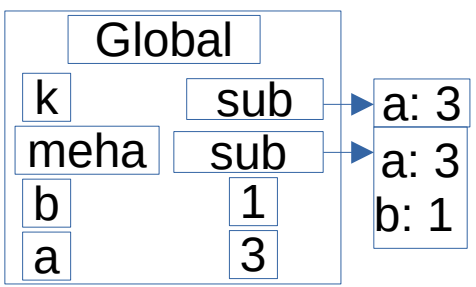


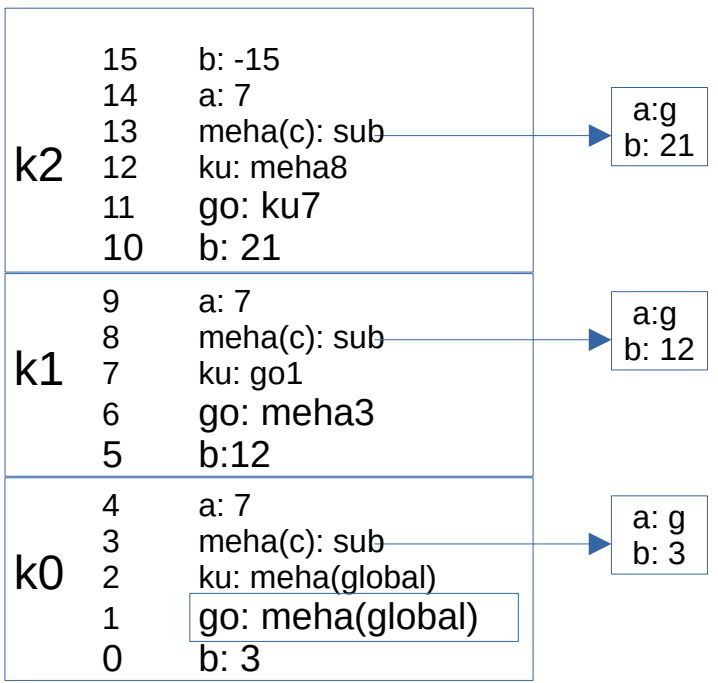
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



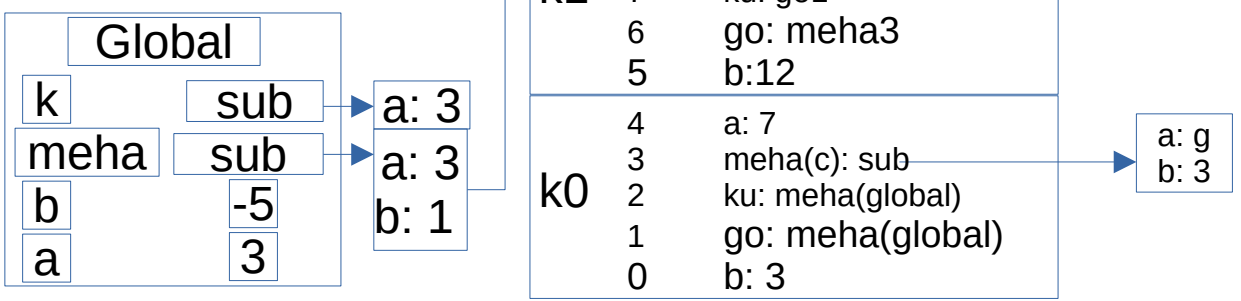


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

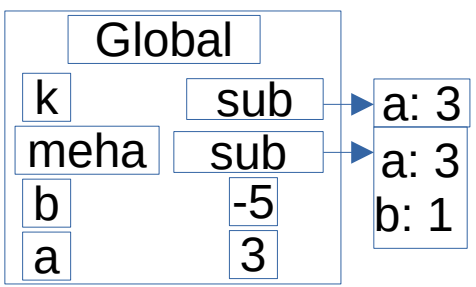


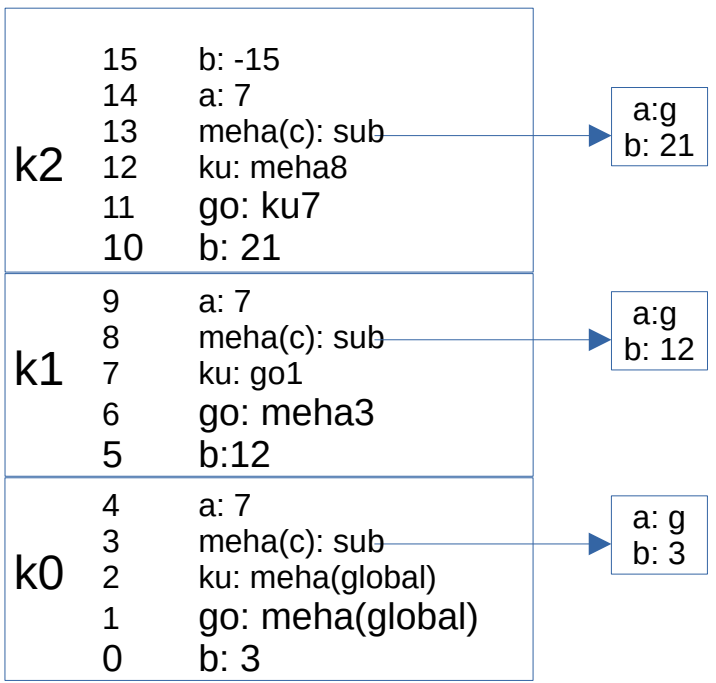


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

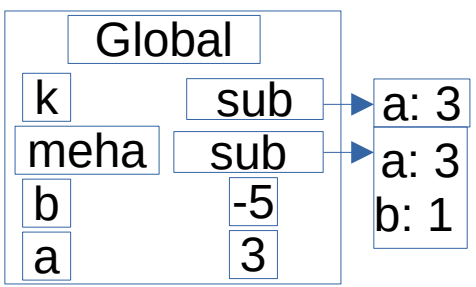


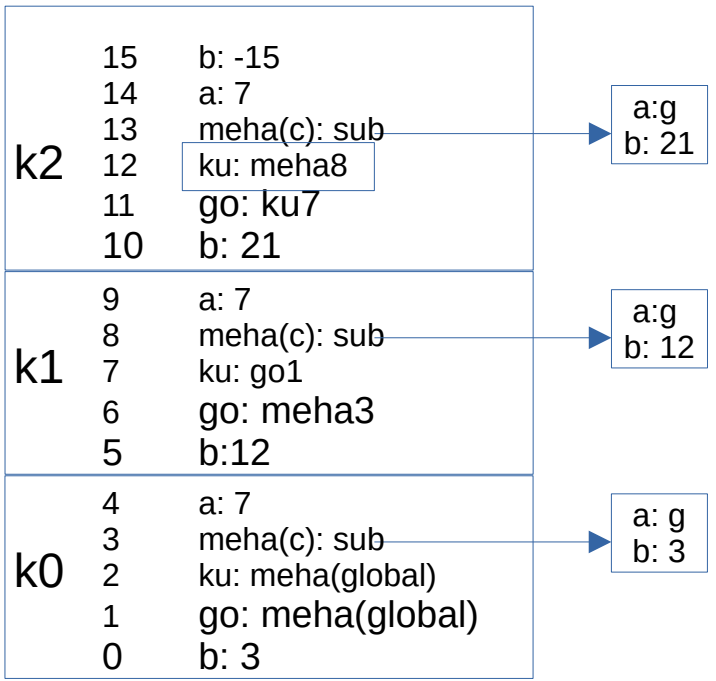
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



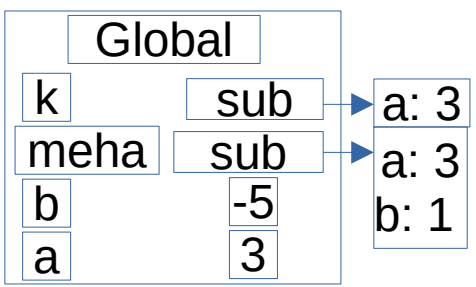


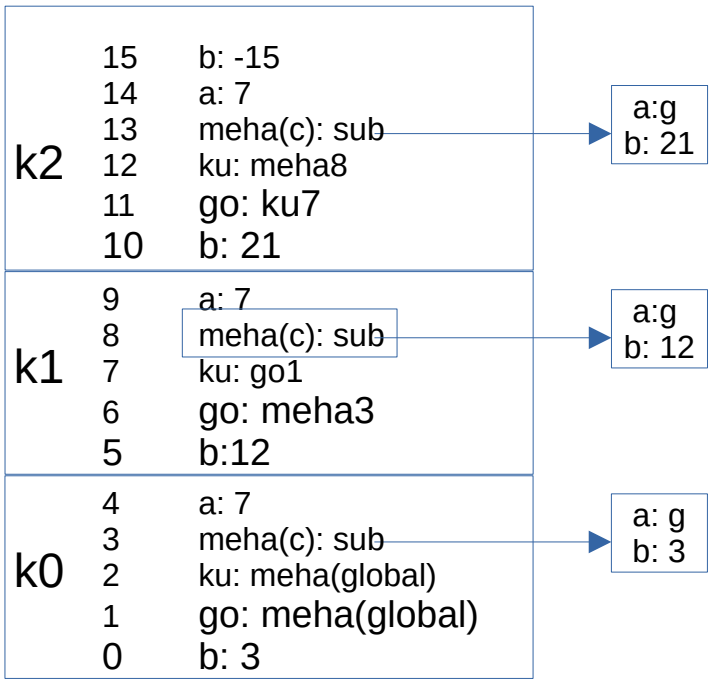
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



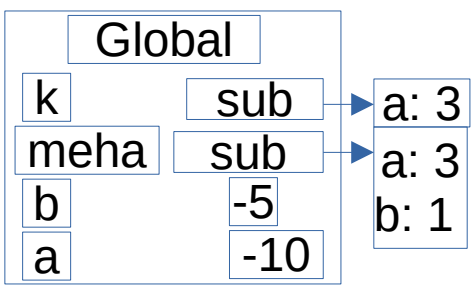


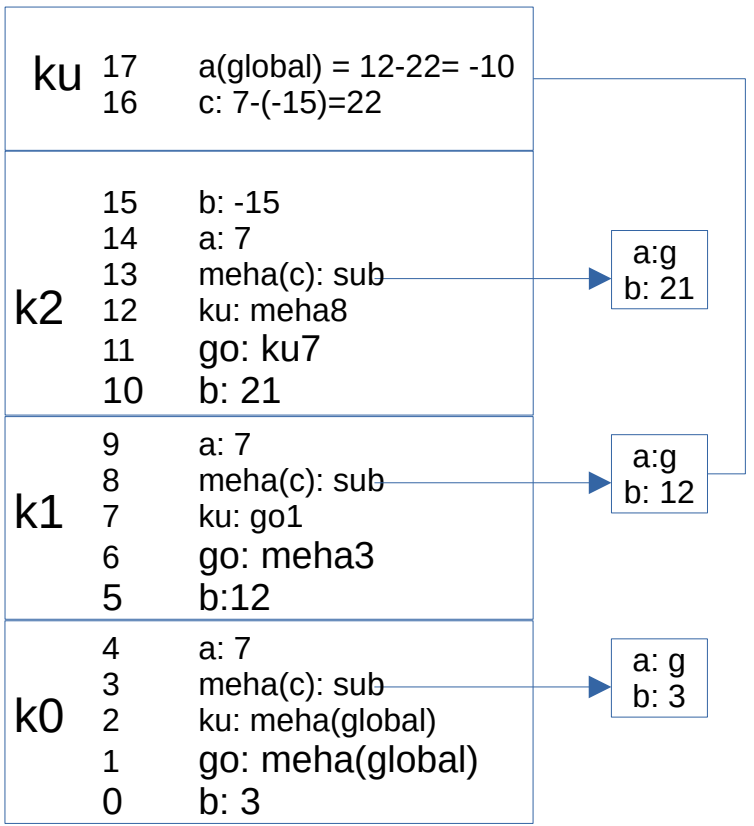
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



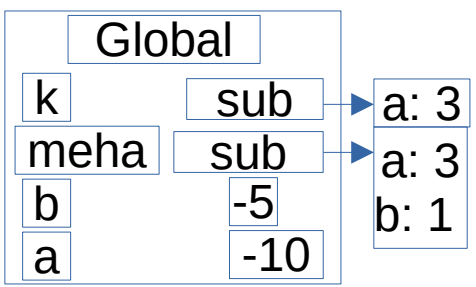


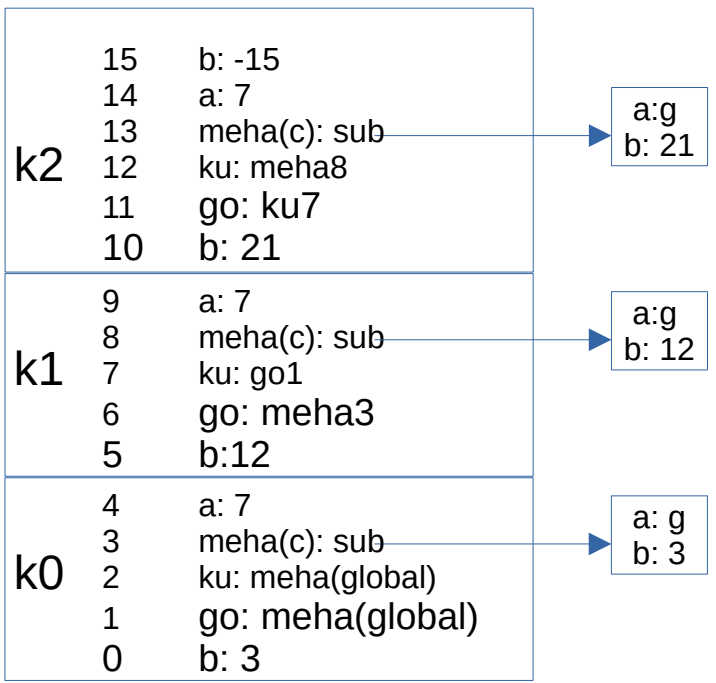
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```





```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

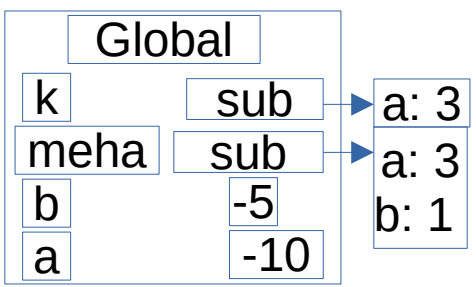


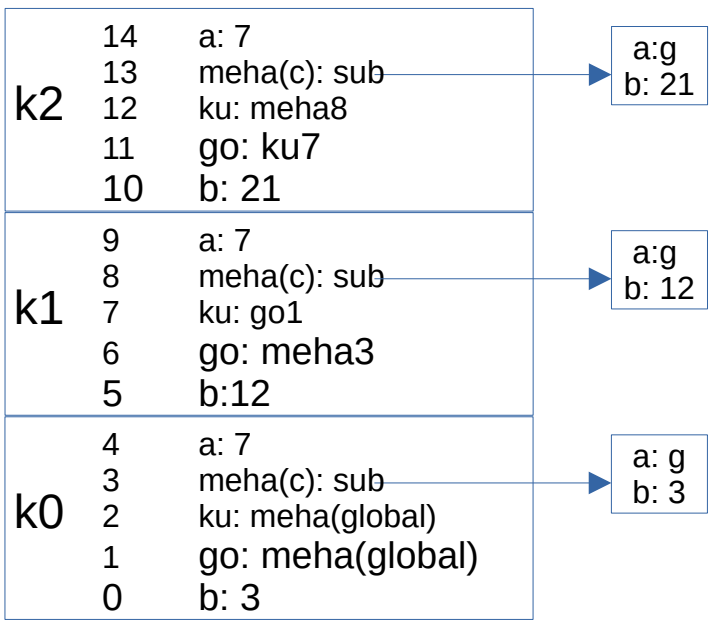


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

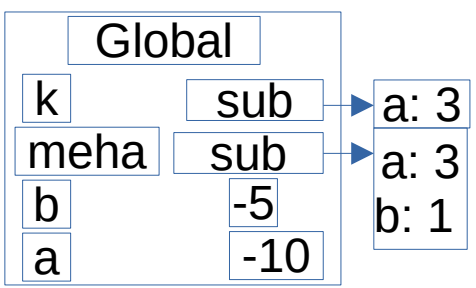


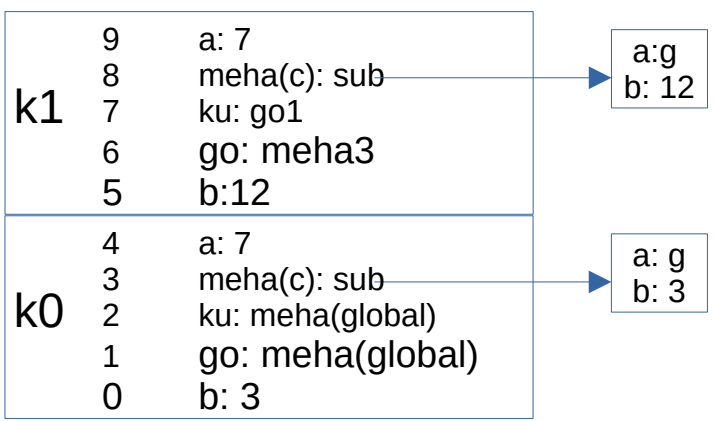


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

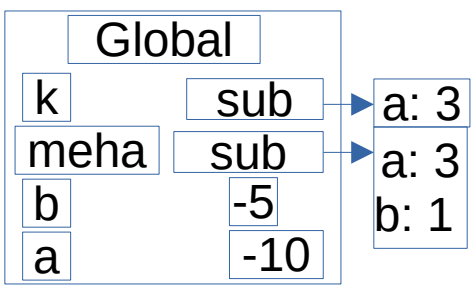


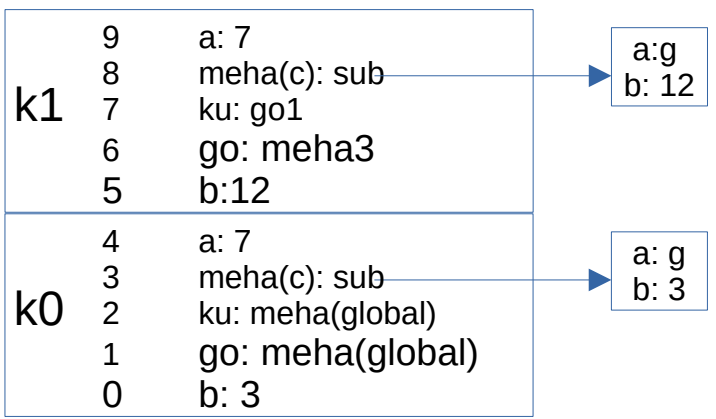


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21
7 12

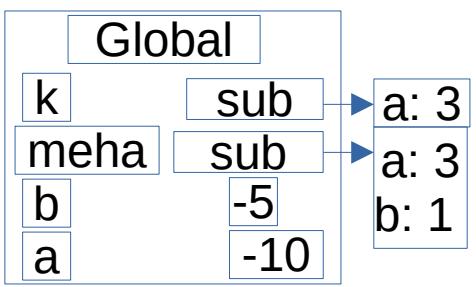


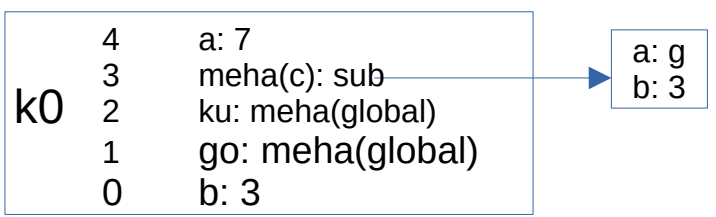


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21
7 12

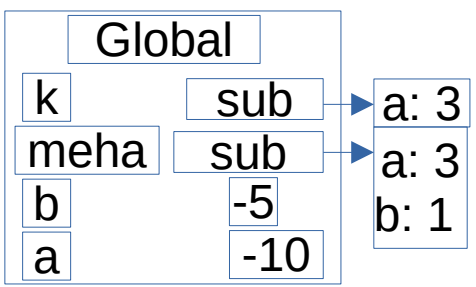


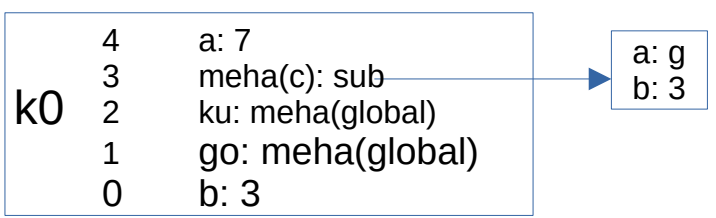


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7   21
7   12
7    3
```

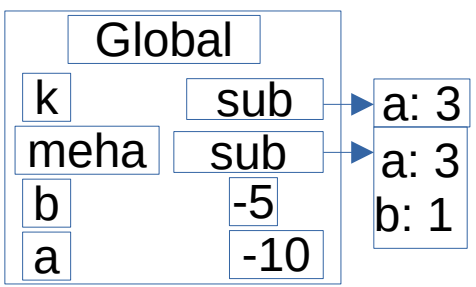




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

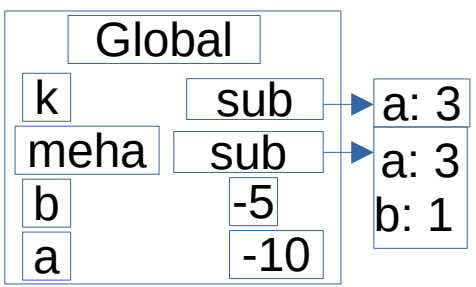
```
7   21
7   12
7    3
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7     21
7     12
7     3
-10   -5
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7      21
7      12
7       3
-10   -5
```

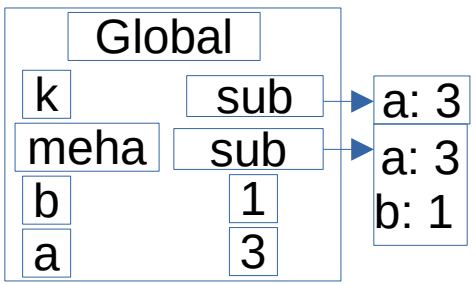
Finalizado

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

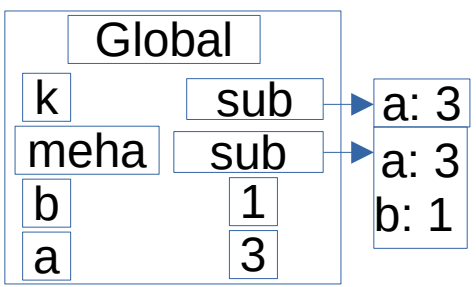
Alcance Dinamico y Clausura con Asociacion Profunda

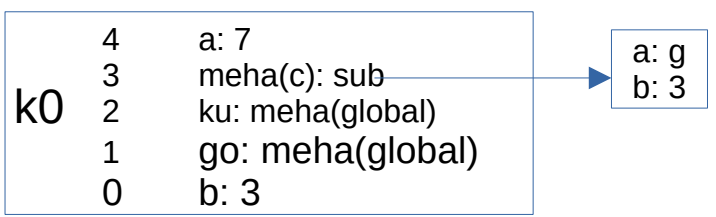
Inicialmente las variables Globales son declaradas.

Las clausuras se realizan Al definir las subrutinas

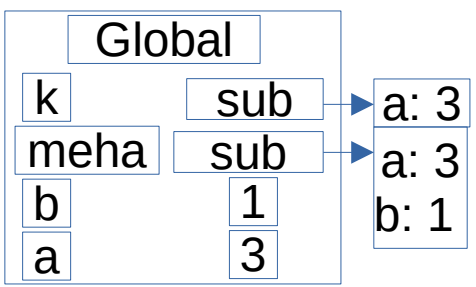


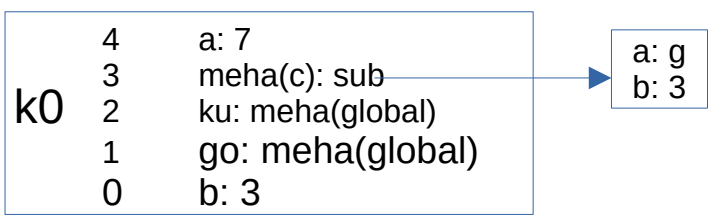
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



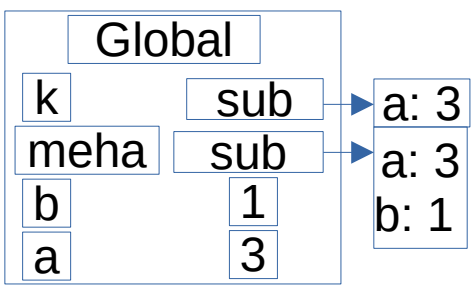


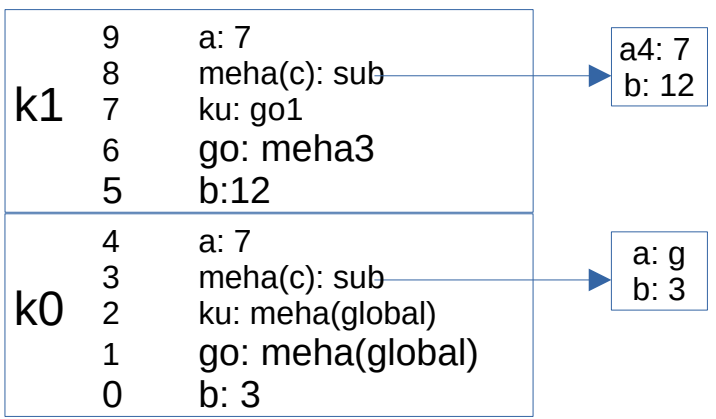
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



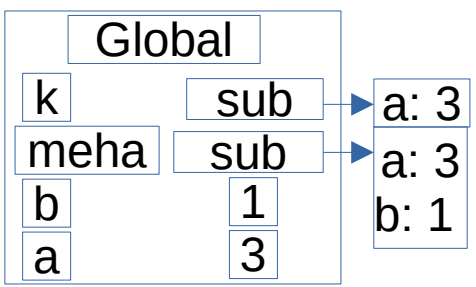


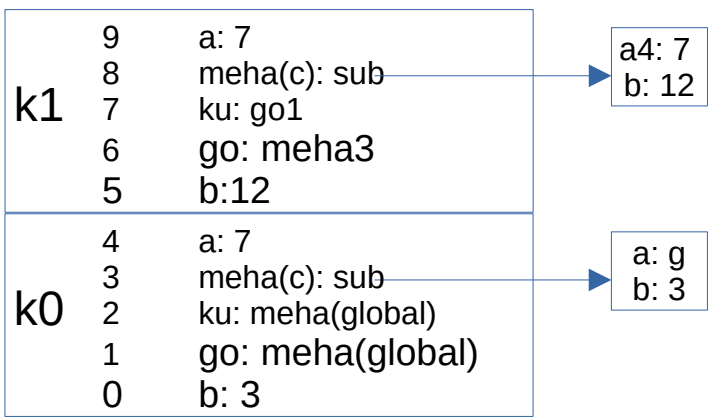
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



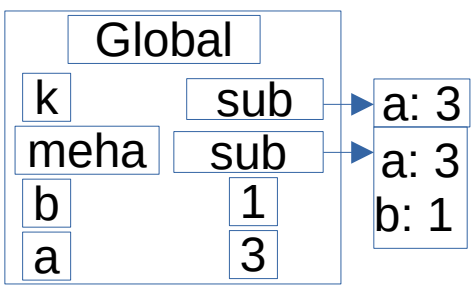


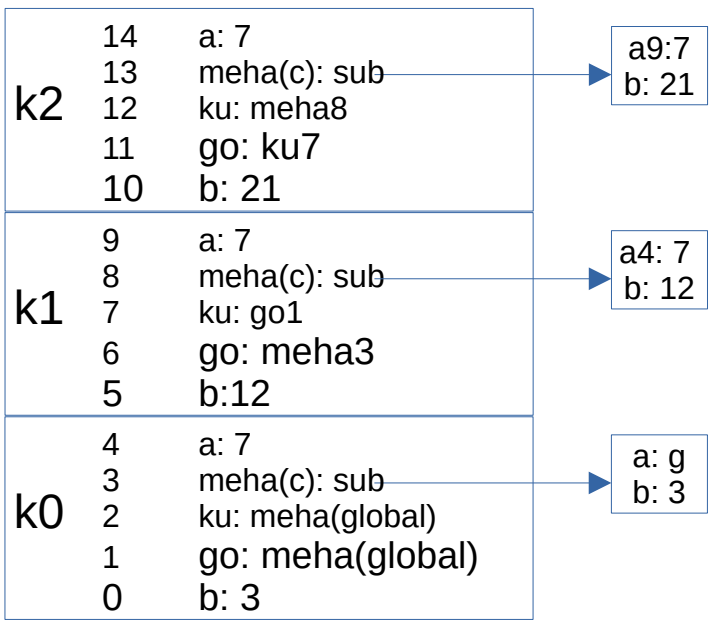
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



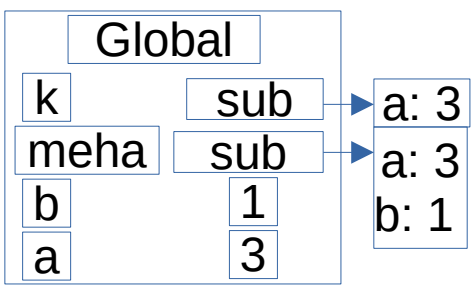


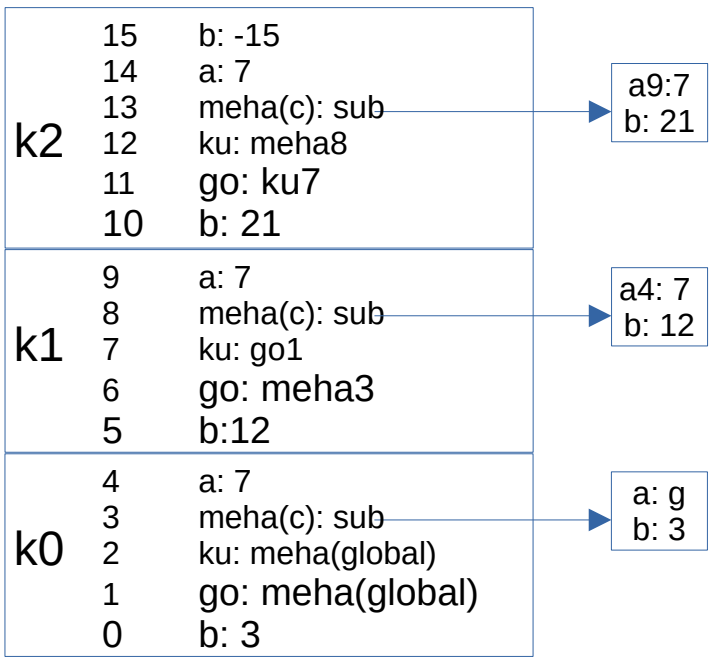
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



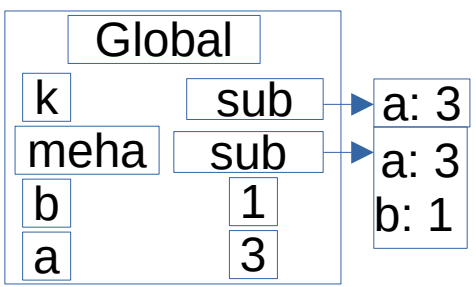


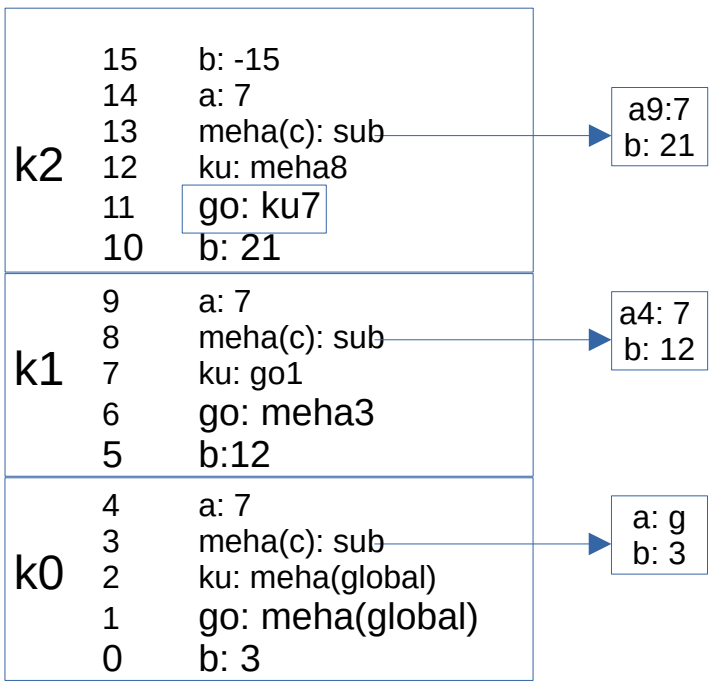
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



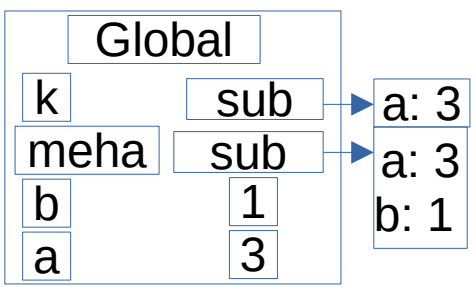


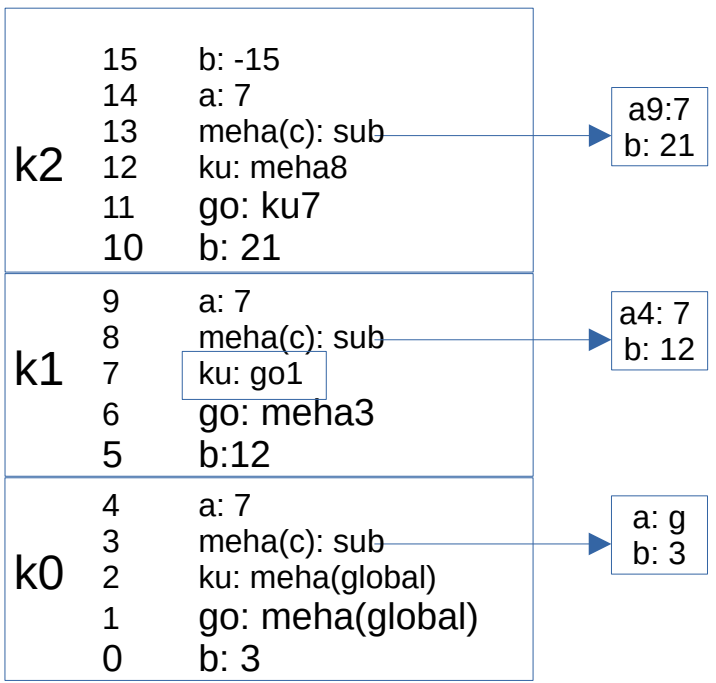
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



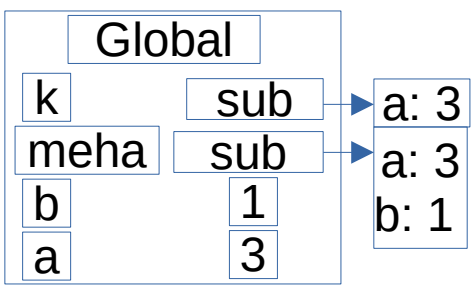


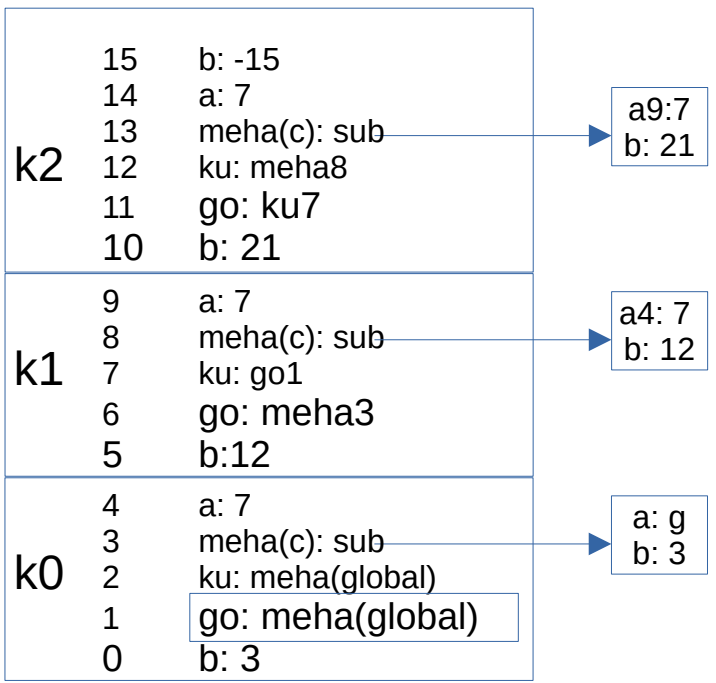
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```





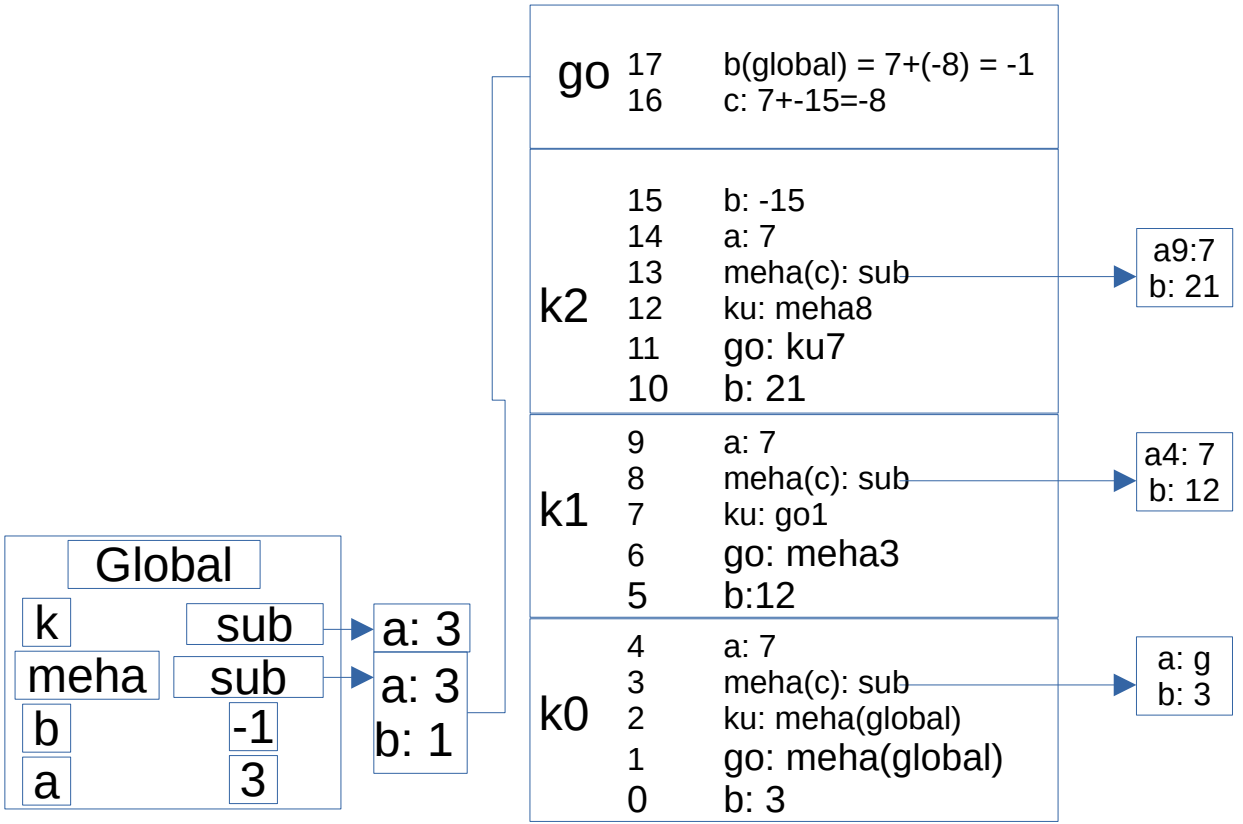
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



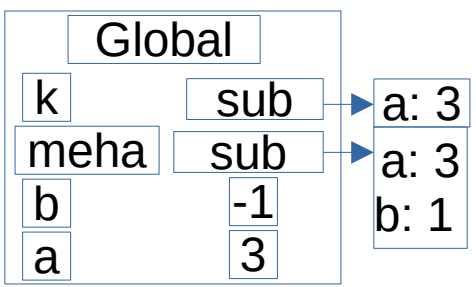


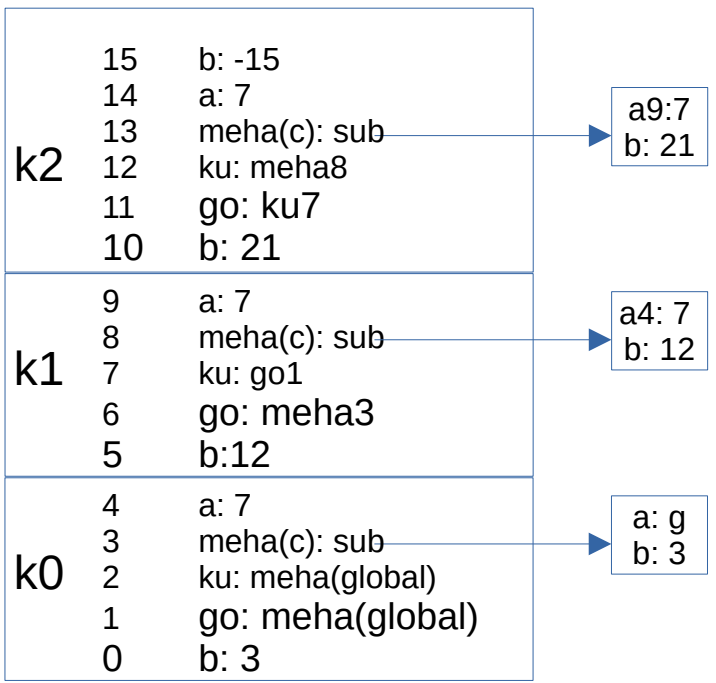
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

b(global) como estaba en la clausura
De meha entonces cambia su valor

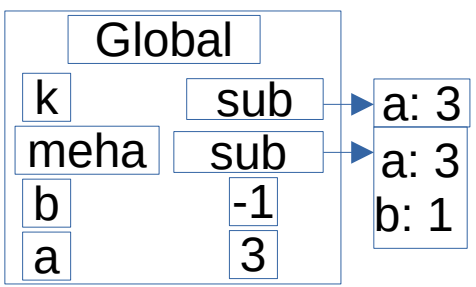


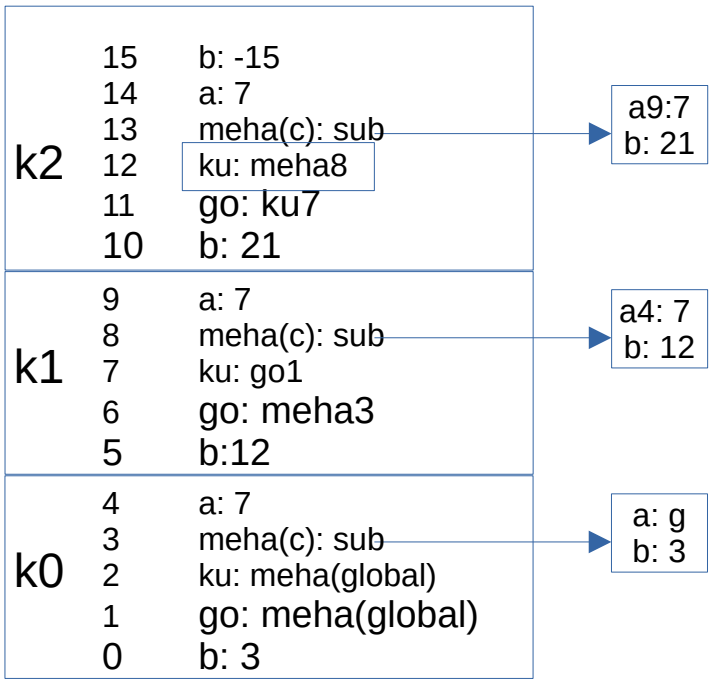
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



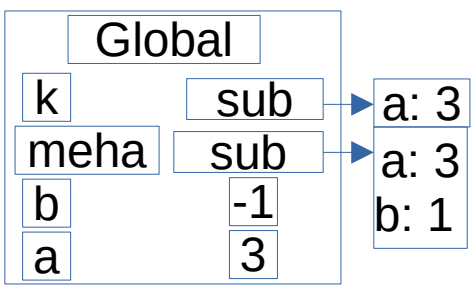


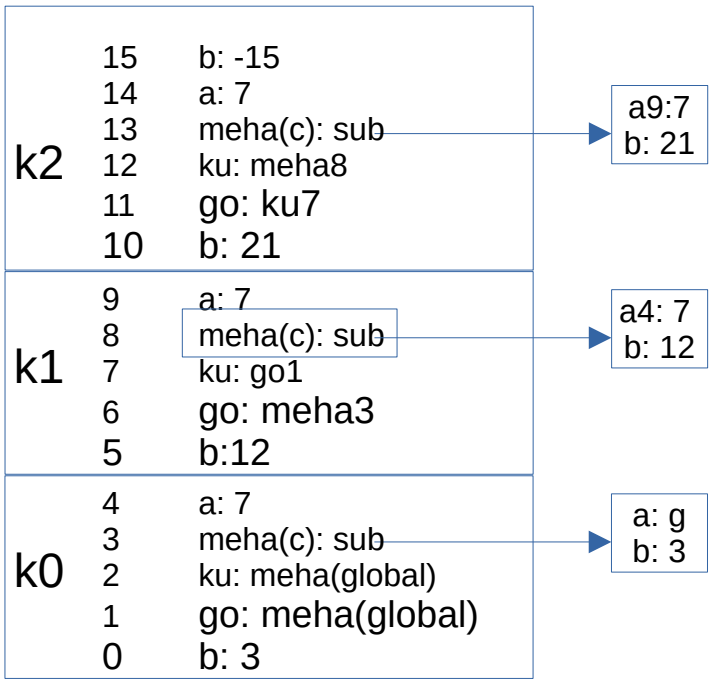
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



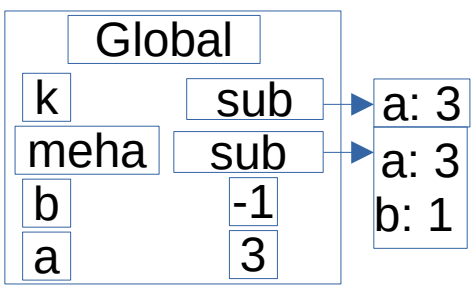


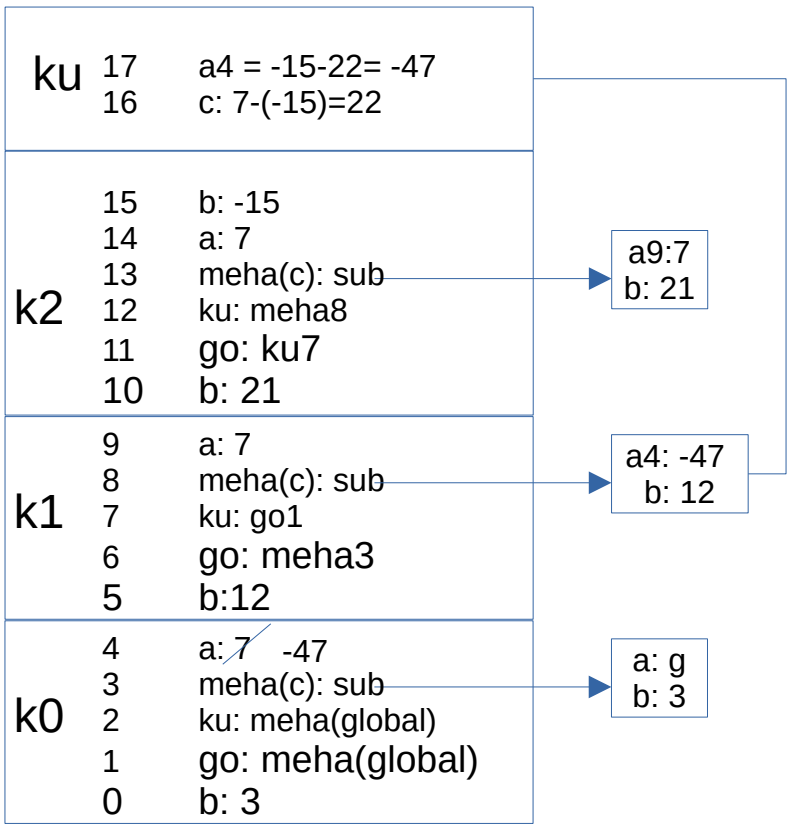
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



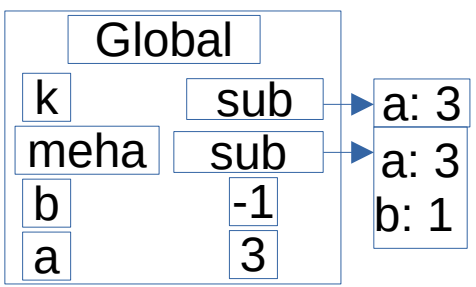


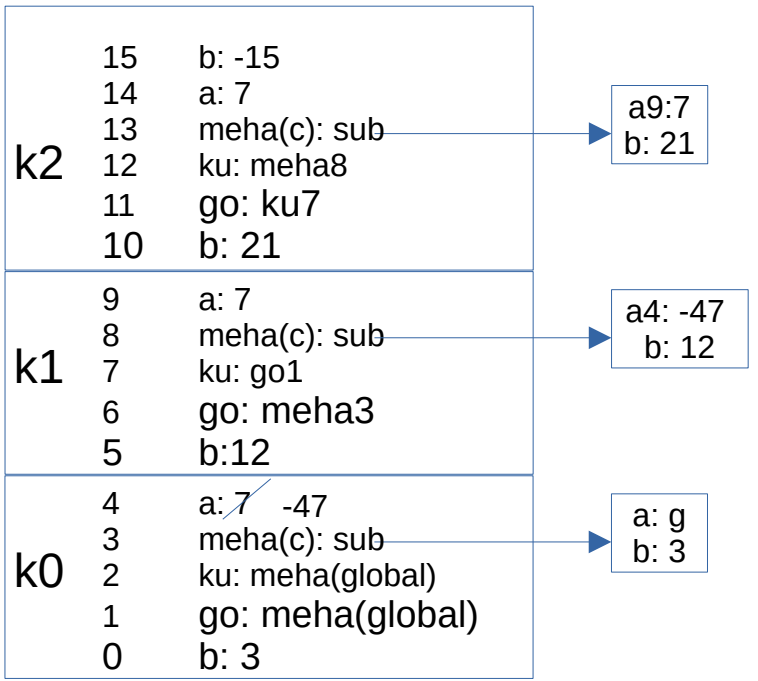
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```





```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

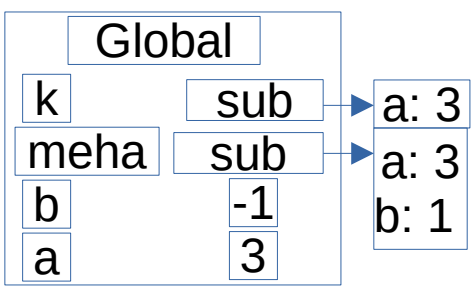


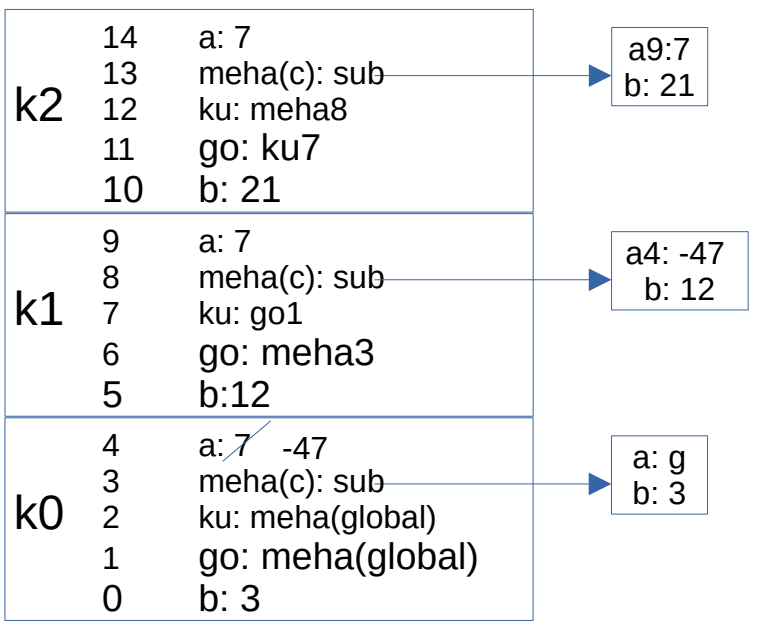


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

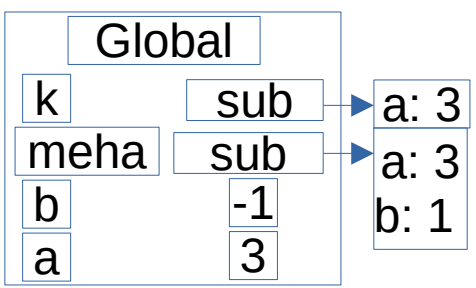


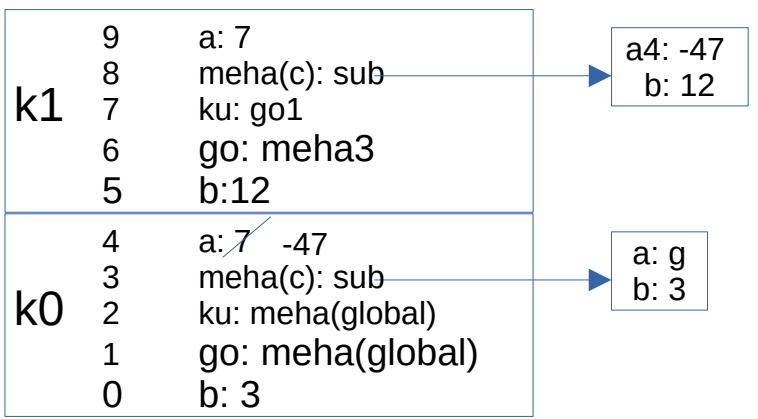


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21

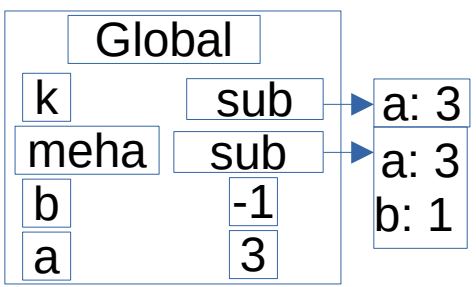


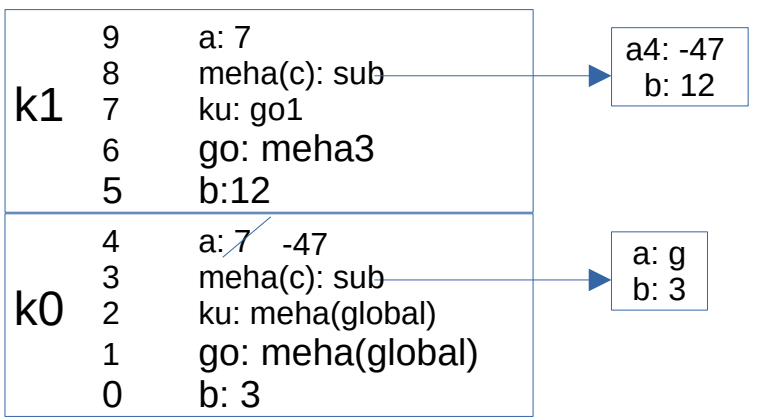


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21
7 12

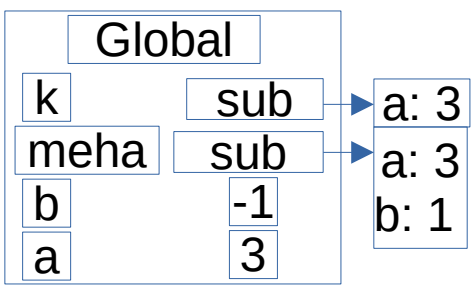


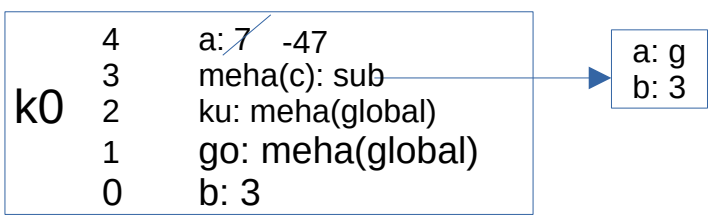


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

7 21
7 12

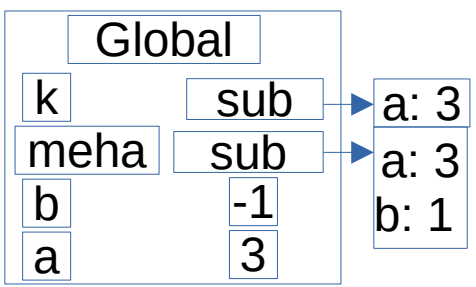


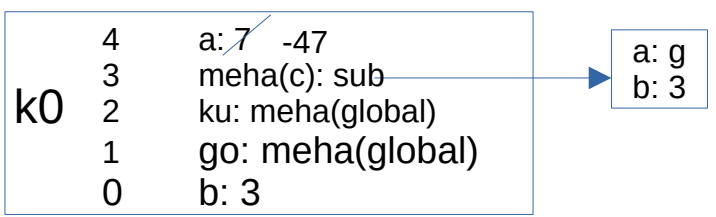


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7   21
7   12
-47    3
```

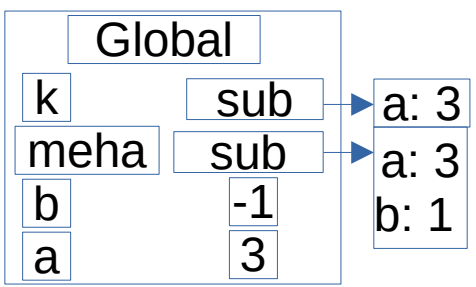




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

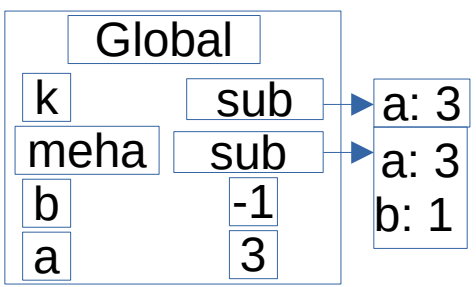
```
7   21
7   12
-47    3
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
7   21
7   12
-47    3
3    -1
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

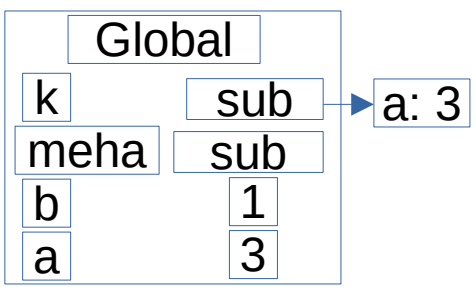
```
7   21
7   12
-47    3
3    -1
```

Finalizado

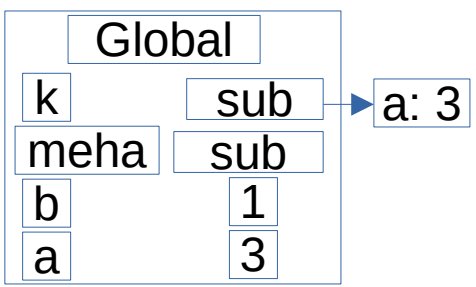
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Alcance Dinamico y Clausura con Asociacion Superficial

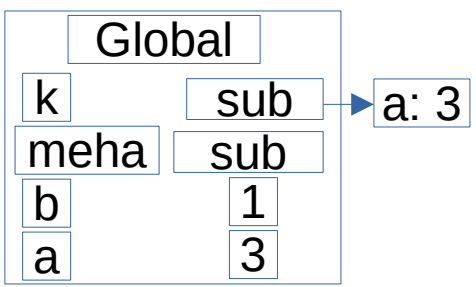
Las clausuras se realizan
Al Ejecutar las subrutinas



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

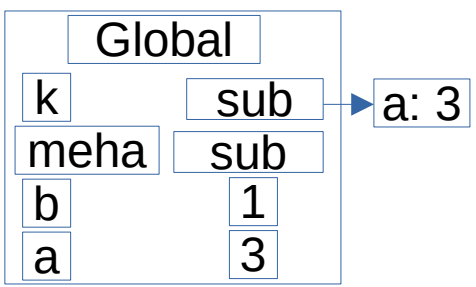


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



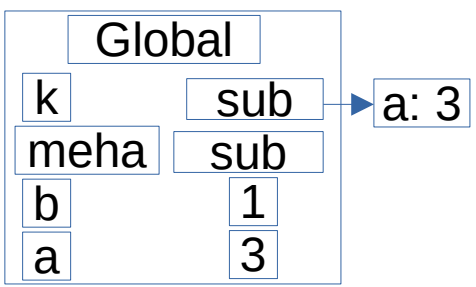
k0
4 a: 7
3 meha: sub
2 ku: meha(global)
1 go: meha(global)
0 b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k0
4 a: 7
3 meha: sub
2 ku: meha(global)
1 go: meha(global)
0 b: 3

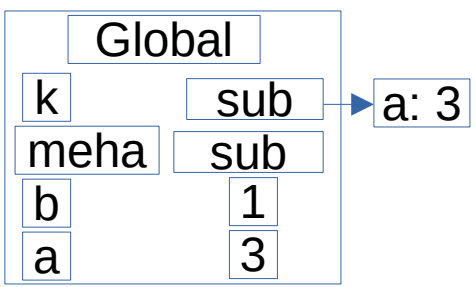
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12

k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

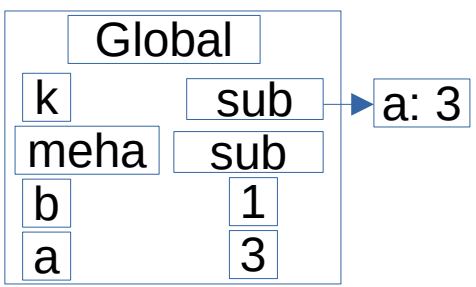
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12

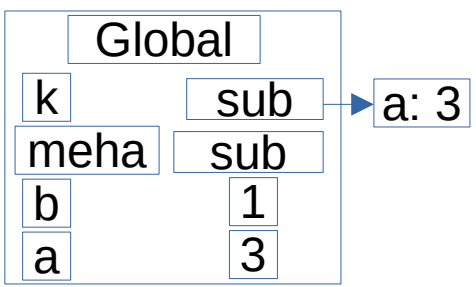
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k2	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

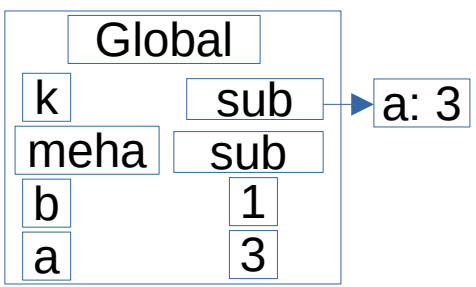
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```



k2	15	b: -15
	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

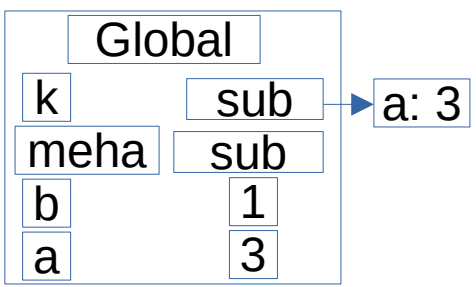
Como se esta ejecutando go se le crea la clausura
Recordando que go11 = ku7= meha(global)

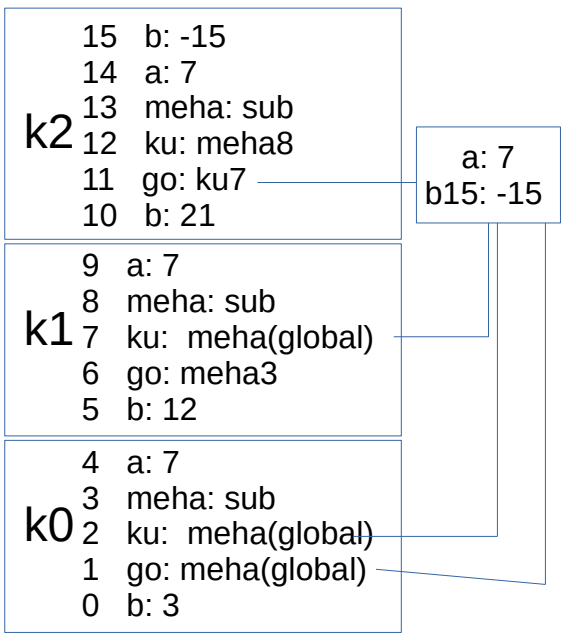


k2	15	b: -15
	14	a: 7
	13	meha: sub
	12	ku: meha8
	11	go: ku7
	10	b: 21
k1	9	a: 7
	8	meha: sub
	7	ku: meha(global)
	6	go: meha3
	5	b: 12
k0	4	a: 7
	3	meha: sub
	2	ku: meha(global)
	1	go: meha(global)
	0	b: 3

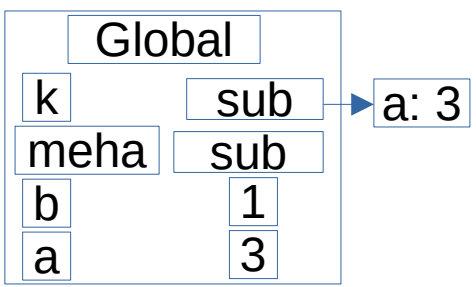
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

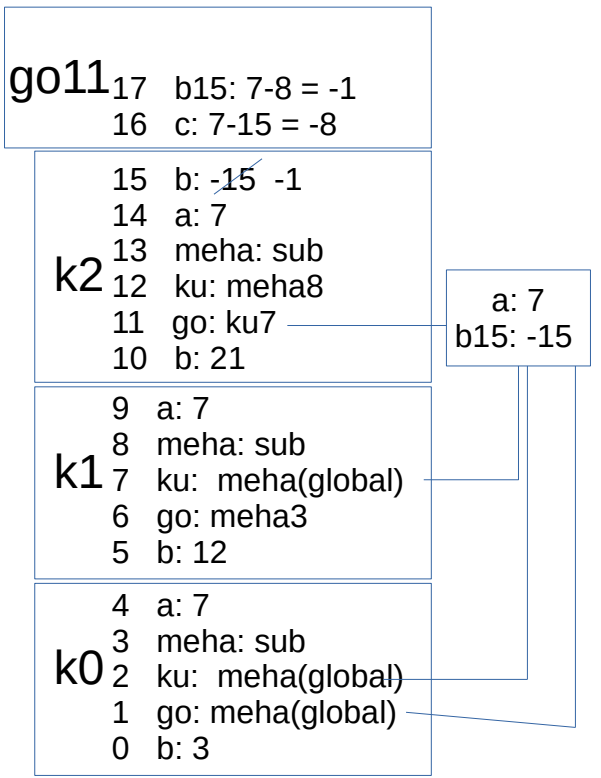
Como hay alcance estatico, en la clausura
Tenemos… a es 7 y b es global





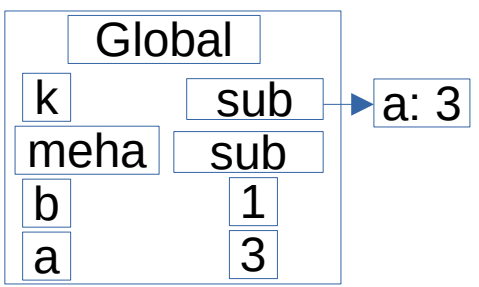
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

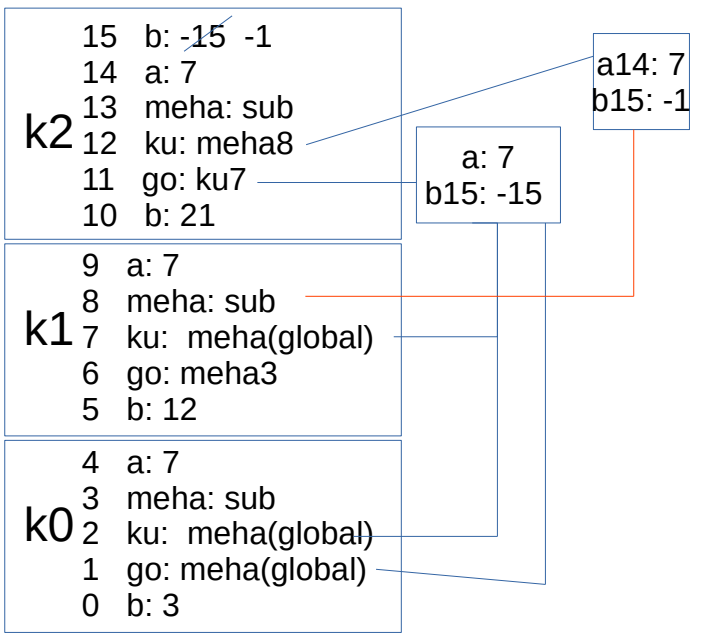




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

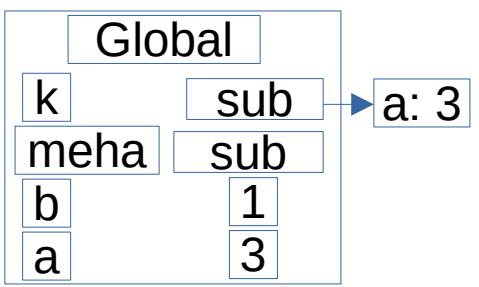
Hacemos la clausura de ku12

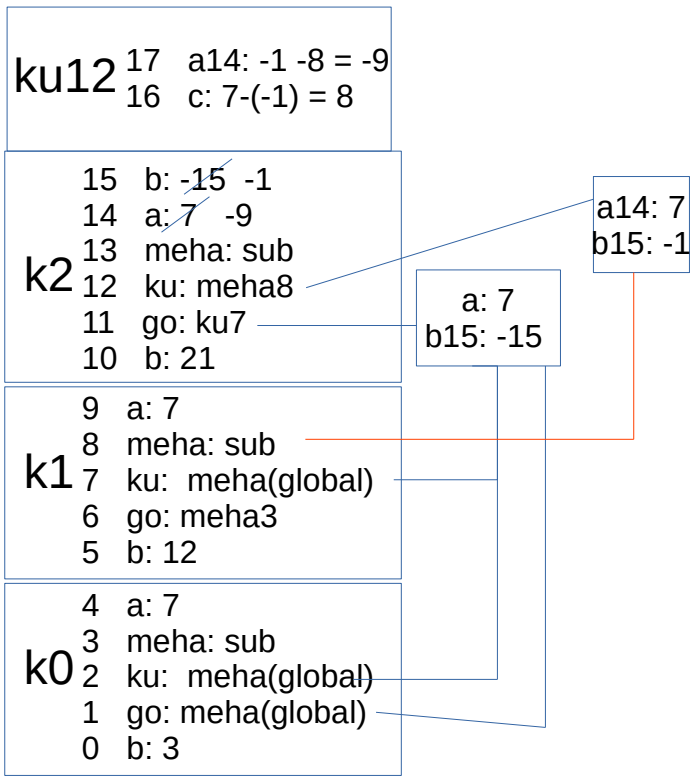




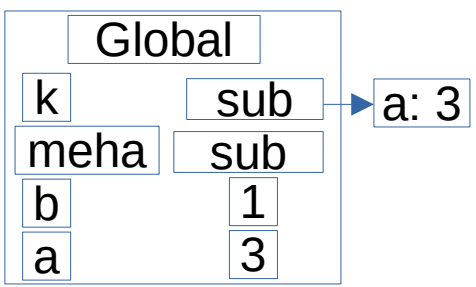
```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

    sub meha(int c) {
            a := b - c;
    }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

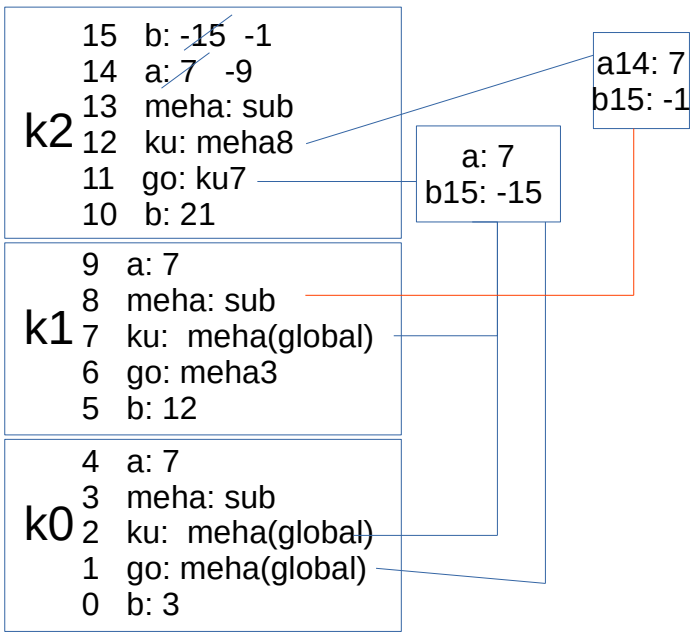
a14 se convierte en -9





```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

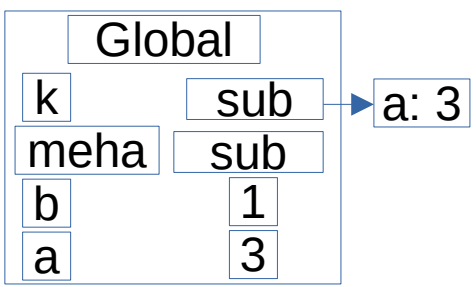


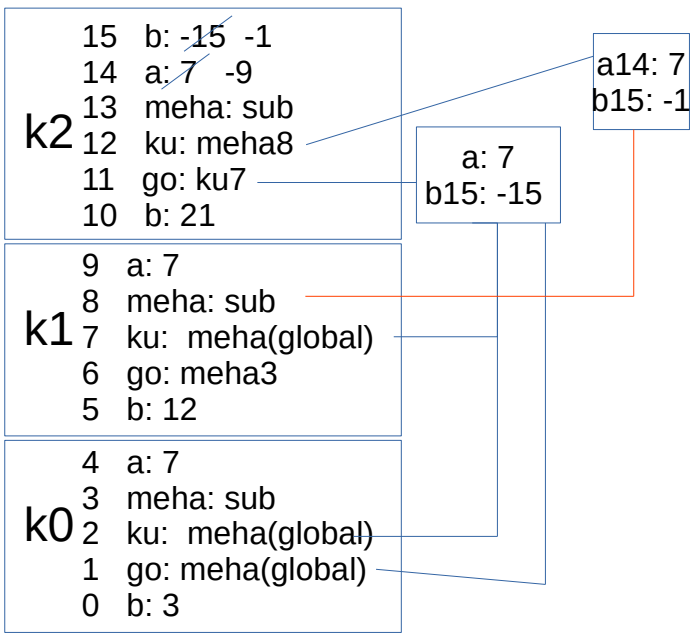


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

-1 -9

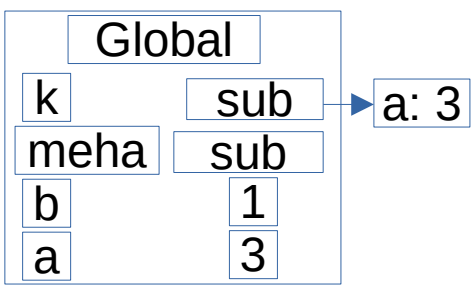


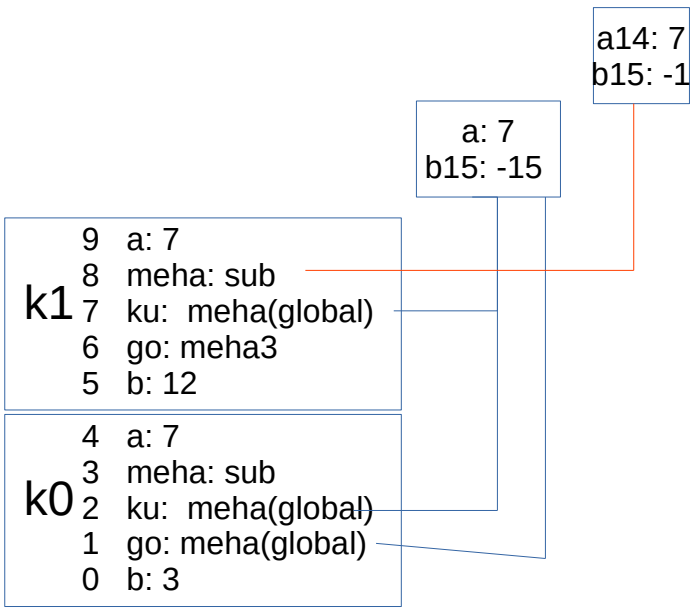


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

-1 -9

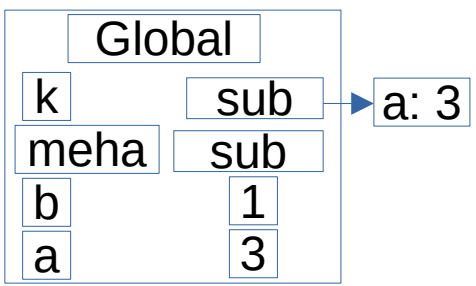


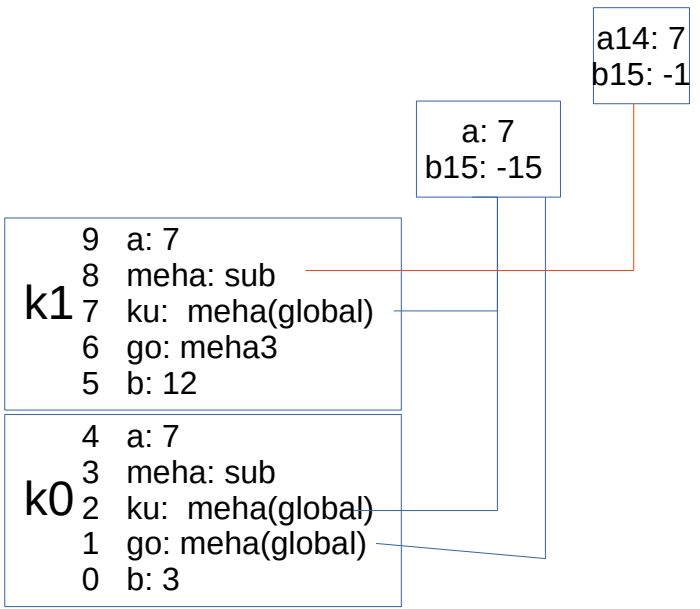


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

-1 -9
7 12

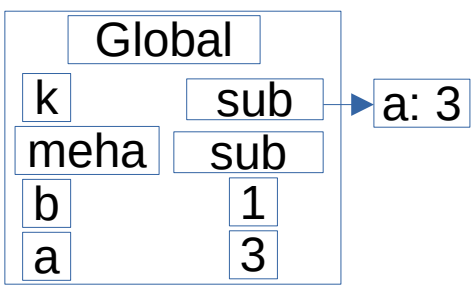


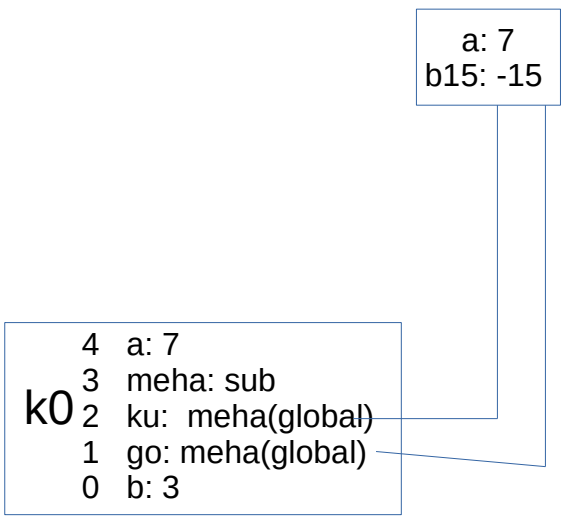


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

-1 -9
7 12

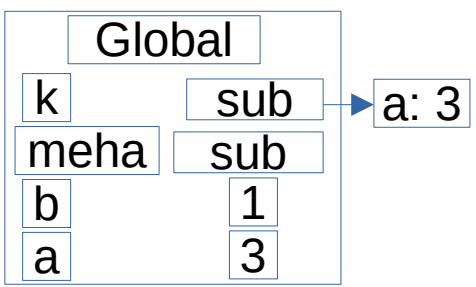


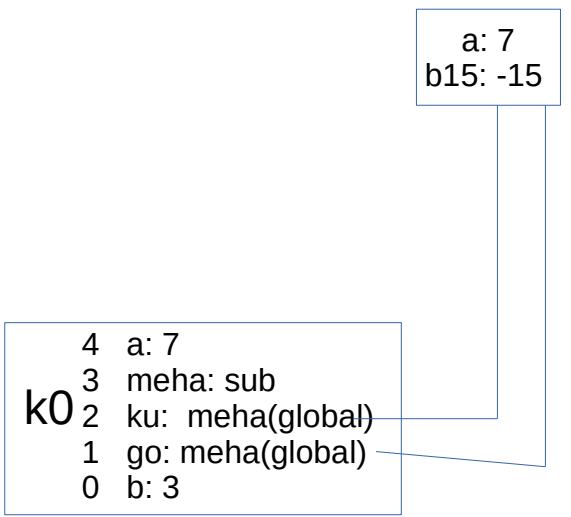


```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
-1  -9
7   12
7   3
```

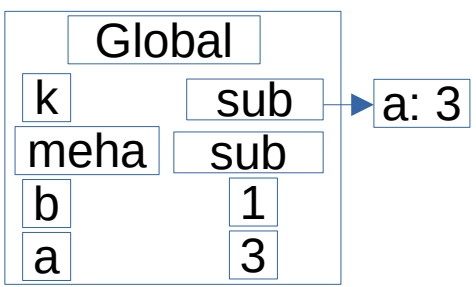




```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

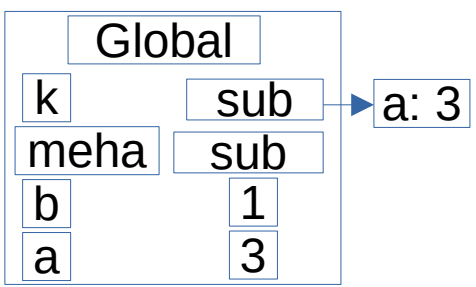
```
-1  -9
7    12
7    3
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
-1  -9
7    12
7    3
3    1
```



```
int a = 2 + 1, b = 1;

sub meha(int c)  {
        b := a + c;
}
sub k (int b, sub go, sub ku) {

        sub meha(int c) {
                a := b - c;
        }

        int a = 6 + 1;
        if (b < 3 * (2 + 1)) {
                k(b + 3 * (2 + 1), meha, go);
        } else if (b < 6 * (2 + 1)) {
                k(b + 3 * (2 + 1), ku, meha);
        } else {
                int b = 6 - b
                go(a + b);
                ku(a - b);
        }
        print(a, b)
}
k(a, meha, meha);
print(a, b)
```

Salida

```
-1  -9
7    12
7    3
3    1
```

Finaliza la ejecucion

Parte 3:

Repositorio github:
https://github.com/Eycer-usb/ci3641/tree/main/Pregunta_3

Parte 4:

Repositorio github:
https://github.com/Eycer-usb/ci3641/tree/main/Pregunta_4

Parte 5:

Repositorio github:
https://github.com/Eycer-usb/ci3641/tree/main/Pregunta_5