

Pregunta 3b

...

“Una imagen vale mil palabras”, pero consume mil veces más memoria.

Codigo Con Sustitucion de Constantes

```
class Abra {  
    int a = 2, b = 1  
    fun cus(int x): int {  
        a = b + x  
        return pide(a)  
    }  
    fun pide(int y): int {  
        return a - y * b  
    }  
}
```

```
class Cadabra extends Abra {  
    Abra zo = new PataDeCabra()  
    fun pide(int y): int {  
        return zo.cus(a + b) - y  
    }  
}
```

```
class PataDeCabra extends Cadabra {  
    int b = 7, c = 6  
    fun cus(int x): int {  
        a = x - 3  
        c = a + b * c  
        return pide(a * b + x)  
    }  
    fun pide(int y): int {  
        return c - y * a  
    }  
}
```

Se consideran las variables a,b, c y zo como
Variables de instancia dado que no son statics (similar a java)

Asociación Dinámica de Métodos

```
Abra ho = new Cadabra()
```

```
Abra po = new PataDeCabra()
```

```
Cadabra cir = new PataDeCabra()
```

```
print(ho.cus(1) + po.cus(1) + cir.cus(1))
```

Inicialmente se crea una variable de tipo Cadabra y dos de tipo PataDeCabra.

Por emplearse asociación dinámica de métodos se tiene que cir y po tendrán los métodos de PataDeCabra. Y dado que Cadabra extiende a Abra y sobre-escribe la función pide entonces ho tiene dos métodos: cus (de Abra) y pide (de Cadabra).

Como el espacio para visualizar la ejecución es limitado se mostrará únicamente la línea print y se mostrará el método que se esté ejecutando.

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Abra {  
    int a = 2, b = 1  
0    fun cus(int x): int {  
1        a = b + x  
2        return pide(a)  
    }  
    fun pide(int y): int {  
3        return a - y * b  
    }  
}
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

cus@Abra	x	1
	b	1
	a	2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Abra {  
    int a = 2, b = 1  
0    fun cus(int x): int {  
1        a = b + x  
2        return pide(a)  
    }  
    fun pide(int y): int {  
3        return a - y * b  
    }  
}
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Abra {  
    int a = 2, b = 1  
0    fun cus(int x): int {  
1        a = b + x  
2        return pide(a)  
    }  
    fun pide(int y): int {  
3        return a - y * b  
    }  
}
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Cadabra extends Abra {
  Abra zo = new PataDeCabra()
0  fun pide(int y): int {
1    return zo.cus(a + b) - y
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Cadabra extends Abra {
  Abra zo = new PataDeCabra()
0  fun pide(int y): int {
1    return zo.cus(a + b) - y
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2


```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	3
	c (de zo)	6
	b (de zo)	7
	a (de zo)	2
pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	3
	c (de zo)	6
	b (de zo)	7
	a (de zo)	2 0
pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	3
	c (de zo)	6 42
	b (de zo)	7
	a (de zo)	2 0
pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)  a*b + x = 0*7 + 3 = 3
  }
4  fun pide(int y): int {
5    return c - y * a
  }
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	3
	c (de zo)	6 42
	b (de zo)	7
	a (de zo)	2 0
pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c
3       return pide(a * b + x)
    }
4   fun pide(int y): int {
5       return c - y * a
    }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	3
cus@PataDeCabra	x	3
	c (de zo)	6 42
	b (de zo)	7
	a (de zo)	2 0



```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
```

Retorna
42-3*0= 42

zo

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	3
cus@PataDeCabra	x	3
	c (de zo)	6 42
	b (de zo)	7
	a (de zo)	2 0

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x) Retorna 42
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	3
	c (de zo)	6 42
	b (de zo)	7
	a (de zo)	2 0
pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2

```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Cadabra extends Abra {
  Abra zo = new PataDeCabra()
0  fun pide(int y): int {
1    return zo.cus(a + b) - y
  }
}
```

Retorna 42 - 2 = 40
zo

cus	PataDeCabra
pide	PataDeCabra

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

pide@Cadabra	y	2
	zo	Tipo PataDCabra
	b	1
	a	2 2
cus@Abra	x	1
	b	1
	a	2 2


```
print( ho.cus(1) + po.cus(1) + cir.cus(1))
```

```
class Abra {  
    int a = 2, b = 1  
0    fun cus(int x): int {  
1        a = b + x  
2        return pide(a)  Retorna 40  
    }  
    fun pide(int y): int {  
3        return a - y * b  
    }  
}
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

cus@Abra	x	1
	b	1
	a	2 2

```
print( 40 + po.cus(1) + cir.cus(1))
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de po)	6
	b (de po)	7
	a (de po)	2

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de po)	6
	b (de po)	7
	a (de po)	2 -2

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c    c = -2 + 7*6=40
3       return pide(a * b + x)
    }
4   fun pide(int y): int {
5       return c - y * a
    }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de po)	6 40
	b (de po)	7
	a (de po)	2 -2

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c
3       return pide(a * b + x)    a*b + x = -2*7 + 1 = -13
    }
4   fun pide(int y): int {
5       return c - y * a
    }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de po)	6 40
	b (de po)	7
	a (de po)	2 -2

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c
3       return pide(a * b + x)
    }
4   fun pide(int y): int {
5       return c - y * a
    }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	-13
cus@PataDeCabra	x	1
	c (de po)	6 40
	b (de po)	7
	a (de po)	2 -2

```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c
3       return pide(a * b + x)
    }
4   fun pide(int y): int {
5       return c - y * a    c-y*a = 40-(-13)*(-2)=14
    }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	-13
cus@PataDeCabra	x	1
	c (de po)	6 40
	b (de po)	7
	a (de po)	2 -2


```
print( 40 + po.cus(1) + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
    int b = 7, c = 6
0   fun cus(int x): int {
1       a = x - 3
2       c = a + b * c
3       return pide(a * b + x)    pide(a*b+x) = 14
    }
4   fun pide(int y): int {
5       return c - y * a
    }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de po)	6 40
	b (de po)	7
	a (de po)	2 -2

```
print( 40 + 14 + cir.cus(1))
```

ho

cus	Abra
pide	Cad abra

po

cus	PataDe Cabra
pide	PataDe Cabra

cir

cus	PataDe Cabra
pide	PataDe Cabra

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {  
    int b = 7, c = 6  
0   fun cus(int x): int {  
1       a = x - 3  
2       c = a + b * c  
3       return pide(a * b + x)  
    }  
4   fun pide(int y): int {  
5       return c - y * a  
    }  
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de cir)	6
	b (de cir)	7
	a (de cir)	2

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {  
    int b = 7, c = 6  
0    fun cus(int x): int {  
1        a = x - 3  
2        c = a + b * c  
3        return pide(a * b + x)  
    }  
4    fun pide(int y): int {  
5        return c - y * a  
    }  
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de cir)	6
	b (de cir)	7
	a (de cir)	2 -2

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c    c = -2 + 7*6=40
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de cir)	6 40
	b (de cir)	7
	a (de cir)	2 -2

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)  a*b + x = -2*7 + 1 = -13
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de cir)	6 40
	b (de cir)	7
	a (de cir)	2 -2

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	-13
cus@PataDeCabra	x	1
	c (de cir)	6 40
	b (de cir)	7
	a (de cir)	2 -2

```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)
  }
4  fun pide(int y): int {
5    return c - y * a    c-y*a = 40-(-13)*(-2)=14
  }
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

pide@PataDeCabra	y	-13
cus@PataDeCabra	x	1
	c (de cir)	6 40
	b (de cir)	7
	a (de cir)	2 -2


```
print( 40 + 14 + cir.cus(1))
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
0  fun cus(int x): int {
1    a = x - 3
2    c = a + b * c
3    return pide(a * b + x)    pide(a*b+x) = 14
  }
4  fun pide(int y): int {
5    return c - y * a
  }
}
```

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

zo

cus	PataDeCabra
pide	PataDeCabra

cus@PataDeCabra	x	1
	c (de cir)	6 40
	b (de cir)	7
	a (de cir)	2 -2

```
Abra ho = new Cadabra()
```

```
Abra po = new PataDeCabra()
```

```
Cadabra cir = new PataDeCabra()
```

```
print(ho.cus(1) + po.cus(1) + cir.cus(1))
```

Imprime
68

ho

cus	Abra
pide	Cadabra

po

cus	PataDeCabra
pide	PataDeCabra

cir

cus	PataDeCabra
pide	PataDeCabra

**¡Un héroe es alguien que supera cada
obstáculo que la vida pone en su camino!**

-(All Might/My Hero Academia)